I hereby certify that this correspondence is being filed via EFS-Web with the United States Patent and Trademark Office on January 19, 2010.

By: /Larry E. Henneman, Jr./

Attorney Docket No.: 0081-011X1
Merged with: 0081-011X2
Merged with: 0081-011X3

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Ex Parte Reexamination:

Control No.: 90/009,034
Filed: September 21, 2006

merged with

Control No.: 90/009,389
Filed: January 16, 2009

merged with

Control No.: 90/010,520
Filed: April 30, 2009

For: HIGH PERFORMANCE, LOW COST MICROPROCESSOR ARCHITECTURE

U.S. Patent No. 5,440,749

Examiner: POKRZYWA, Joseph R.

Technology Center/Art Unit: 3992

TRANSMITTAL FOR AMENDMENT IN RESPONSE TO NON FINAL OFFICE ACTION IN MERGED EX PARTE REEXAMINATION PROCEEDINGS

Mail Stop ***Ex Parte Reexamination***
Central Reexamination Unit
P.O. Box 1450
Alexandria, VA 22313-1450

TRANSMITTAL FOR AMENDMENT

Sir:

Transmitted herewith for filing in the above-referenced merged reexamination proceedings are the following:

1. Amendment (58 pages); and
2. Certification of Service (2 pages).

The required fees are being paid via credit card in conjunction with the electronic filing of this paper.

Respectfully submitted,

Dated: January 19, 2010

/Larry E. Henneman, Jr./

Larry E. Henneman, Reg. No. 41,063
Henneman & Associates, PLC
70 N. Main Street
Three Rivers, MI 49093

Phone: (269) 279-8820
Fax: (269) 279-8830

I hereby certify that this correspondence is being filed via EFS-Web with the United States Patent and Trademark Office on January 19, 2009.

By: /Larry E. Henneman, Jr./

Attorney Docket No.: 0081-011X1
Merged with: 0081-011X2
Merged with: 0081-011X3

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Ex Parte Reexamination:

Control No.: 90/009,034
Filed: September 21, 2006

merged with

Control No.: 90/009,389
Filed: January 16, 2009

merged with

Control No.: 90/010,520
Filed: April 30, 2009

For: HIGH PERFORMANCE, LOW COST MICROPROCESSOR ARCHITECTURE

U.S. Patent No. 5,440,749

Examiner: POKRZYWA, Joseph R.

Technology Center/Art Unit: 3992

Customer No. 40972

AMENDMENT IN RESPONSE TO NON FINAL OFFICE ACTION IN EX PARTE REEXAMINATION PROCEEDINGS

Mail Stop ***Ex Parte Reexamination***
Central Reexamination Unit
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

In response to the non final office action mailed November 19, 2009, in the merged reexamination proceedings, please enter the following amendments.

A listing of the claims begins on page 2.

Remarks begin on page 21.

1. (Previously presented) A microprocessor system, comprising a central processing unit integrated circuit, a memory external of said central processing unit integrated circuit, a bus connecting said central processing unit integrated circuit to said memory, and means connected to said bus for fetching instructions for said central processing unit integrated circuit on said bus from said memory, said means for fetching instructions being configured and connected to fetch multiple sequential instructions from said memory in parallel and supply the multiple sequential instructions to said central processing unit integrated circuit during a single memory cycle, said bus having a width at least equal to a number of bits in each of the instructions times a number of the instructions fetched in parallel, said central processing unit integrated circuit including an arithmetic logic unit and a first push down stack connected to said arithmetic logic unit, said first push down stack including means for storing a top item connected to a first input of said arithmetic logic unit to provide the top item to the first input and means for storing a next item connected to a second input of said arithmetic logic unit to provide the next item to the second input, a remainder of said first push down stack being connected to said means for storing a next item to receive the next item from said means for storing a next item when pushed down in said push down stack, said arithmetic logic unit having an output connected to said means for storing a top item.

2. (Original) The microprocessor system of claim 1 additionally comprising means connected to said means for fetching multiple instructions for determining by decoding the multiple instructions if multiple instructions fetched by said means for fetching multiple instructions require a memory access, said means for fetching multiple instructions fetching additional multiple instructions if decoding the multiple instructions shows that the multiple instructions do not require a memory access.

3. (Original) The microprocessor system of claim 2 in which the decoding determines if the multiple instructions do not require a memory access by a state of a bit of each of the multiple instructions.

4. (Original) The microprocessor system of claim 3 in which the bit is a most significant bit of the multiple instructions.

5. (Original) The microprocessor system of claim 1 additionally comprising an instruction register for the multiple instructions connected to said means for fetching instructions, means connected to said instruction register for supplying the multiple instructions in succession from said instruction register, a counter connected to control said means for supplying the multiple instructions to supply the multiple instructions in succession, means for decoding the multiple instructions connected to receive the multiple instructions in succession from the means for supplying the multiple instructions, said counter being connected to said means for decoding to receive incrementing and reset control signals from said means for decoding, said means for decoding being configured to supply the reset control signal to said counter and to supply a control signal to said means for fetching instructions in response to a SKIP instruction in the multiple instructions.

6. (Original) The microprocessor system of claim 5 additionally comprising a loop counter connected to receive a decrement control signal from said means for decoding, said means for decoding being configured to supply the reset control signal to said counter and the decrement control signal to said loop counter in response to a MICROLOOP instruction in the multiple instructions to provide a microloop within the multiple instructions in said instruction register for a number of repetitions controlled by said loop counter.

7. (Original) The microprocessor system of claim 1 additionally comprising an instruction register for the multiple instructions and a variable width operand to be used with one of the multiple instructions connected to said means for fetching instructions, means connected to said instruction register for supplying the multiple instructions in succession from said instruction register, a counter connected to control said means for supplying the multiple instructions to supply the multiple instructions in succession,

means for decoding the multiple instructions connected to receive the multiple instructions in succession from the means for supplying the multiple instructions, said counter being connected to said means for decoding to receive incrementing and reset control signals from said means for decoding, said means for decoding being configured to control said counter in response to an instruction utilizing the variable width operand stored in said instruction register, and means connected to said counter to select the variable width operand for use with the instruction utilizing the variable width operand in response to said counter.

8. (Original) A microprocessor system, comprising a central processing unit, a memory, a bus connecting said central processing unit to said memory, and means connected to said bus for fetching instructions for said central processing unit on said bus from said memory, said means for fetching instructions being configured and connected to fetch multiple sequential instructions from said memory in parallel and supply the multiple sequential instructions to said central processing unit during a single memory cycle, said central processing unit including an arithmetic logic unit and a first push down stack connected to said arithmetic logic unit, said first push down stack further including means for storing a top item connected to a first input of said arithmetic logic unit to provide the top item to the first input, means for storing a next item connected to a second input of said

arithmetic logic unit to provide the next item to the second input, said arithmetic logic unit having an output connected to said means for storing a top item, a second push down stack, said means for storing a top item being connected to provide an input to said second push down stack and a control means connected between said means for storing a top item and said second push down stack for controlling provision of the input to said second push down stack, said second push down stack additionally being configured as a register file and said means for storing a top item and said second push down stack additionally configured as the register file being bidirectionally connected.

9. (Original) A microprocessor system, comprising a central processing unit, a dynamic random access memory, a bus connecting said central processing unit to said dynamic random access memory, and multiplexing means on said bus between said central processing unit and said dynamic random access memory, said multiplexing means being connected and configured to provide multiplexed row addresses, column addresses and data on said bus from said central processing unit to said dynamic random access memory and to provide data from said dynamic random access memory to said central processing unit, and

means connected to said bus for fetching instructions for said central processing unit on said bus from said dynamic random access memory, said means for fetching instructions being configured to fetch multiple sequential instructions from said dynamic random access memory in parallel and supply the multiple instructions to said central processing unit during a single memory cycle,

said central processing unit including an arithmetic logic unit and a first push down stack connected to said arithmetic logic unit, said first push down stack including means for storing a top item connected to a first input of said arithmetic logic unit to provide the top item to the first input, and means for storing a next

item connected to a second input of said arithmetic logic unit to provide the next item to the second input, a remainder of said first push down stack being connected to said means for storing a next item to receive the next item from said means for storing a next item when pushed down in said push down stack, said arithmetic logic unit having an output connected to said means for storing a top item.

10. (Original) The microprocessor system of claim 9 additionally comprising a second push down stack, said means for storing a top item being connected to provide an input to said second push down stack and a control means connected between said means for storing a top item and said second push down stack for controlling provision of the input to said second push down stack.

11. (Original) The microprocessor system of claim 10 in which said second push down stack is additionally configured as a register file and said means for storing a top item and said second push down stack additionally configured as the register file are bidirectionally connected.

12. (Original) The microprocessor system of claim 11 additionally comprising means connected to said means for fetching multiple instructions for determining by decoding the multiple instructions if multiple instructions fetched by said means for fetching multiple instructions require a memory access, said means for fetching multiple instructions fetching additional multiple instructions if decoding the multiple instructions shows that the multiple instructions do not require a memory access.

13. (Original) The microprocessor system of claim 12 additionally comprising an instruction register for the multiple instructions connected to said means for fetching instructions, means connected to said instruction register for supplying the

multiple instructions in succession from said instruction register, a counter connected to control said means for supplying the multiple instructions to supply the multiple instructions in succession, means for decoding the multiple instructions connected to receive the multiple instructions in succession from the means for supplying the multiple instructions, said counter being connected to said means for decoding to receive incrementing and reset control signals from said means for decoding, said means for decoding being configured to supply the reset control signal to said counter and to supply a control signal to said means for fetching instructions in response to a SKIP instruction in the multiple instructions.

14. (Original) The microprocessor system of claim 13 additionally comprising a loop counter connected to receive a decrement control signal from said means for decoding, said means for decoding being configured to supply the reset control signal to said counter and the decrement control signal to said loop counter in response to a MICROLOOP instruction in the multiple instructions within the multiple instructions in said instruction register for a number of repetitions controlled by said loop counter.

15. (Original) The microprocessor system of claim 13 in which said means for decoding is configured to control said counter in response to one of the multiple instructions utilizing a variable width operand stored in said instruction register with the multiple instructions, said microprocessor system additionally comprising means connected to said counter to select the variable width operand for use with the instruction utilizing the variable width operand in response to a state of said counter resulting from control of said counter by said means for decoding.

16. (Original) The microprocessor system of claim 12 in which the decoding determines if the multiple instructions do not require a memory access by a state of a bit of each of the multiple instructions.

17. (Original) The microprocessor system of claim 16 in which the bit is a most significant bit of the multiple instructions.

18. (Original) The microprocessor system of claim 9 additionally comprising a programmable read only memory containing instructions connected to said bus, means connected to said bus for fetching instructions for said central processing unit on said bus, said means for fetching instructions including means for assembling a plurality of instructions from said programmable read only memory, storing the plurality of instructions in said dynamic random access memory and subsequently supplying the plurality of instructions from said dynamic random access memory to said central processing unit on said bus.

19. (Original) The microprocessor system of claim 9 additionally comprising a direct memory access processing unit having the capacity to request and execute instructions, said bus connecting said direct memory access processing unit to said dynamic random access memory, said dynamic random access memory containing instructions for said central processing unit and said direct memory access processing unit, said direct memory access processing unit being connected to means for fetching instructions for said central processing unit on said bus and for fetching instructions for said direct memory access processing unit on said bus.

20. (Original) The microprocessor system of claim 19 additionally comprising a variable speed system clock connected to said central processing unit and a fixed speed system clock connected to control said means for fetching instructions for said

central processing unit and for fetching instructions for said direct memory access processing unit.

21. (Original) The microprocessor system of claim 9 in which said microprocessor system is configured to provide different memory access timing for different storing capacity sizes of said dynamic random access memory by including a sensing circuit and a driver circuit, and an output enable line connected between said dynamic random access memory, said sensing circuit and said driver circuit, said sensing circuit being configured to provide a ready signal when said output enable line reaches a predetermined electrical level after a memory read operation as a function of different capacitance on said bus as a result of the different storing capacity sizes of said dynamic random access memory, said microprocessor system being configured so that said driver circuit provides an enabling signal on said output enable line responsive to the ready signal.

22. (Original) The microprocessor system of claim 21 in which the predetermined electrical level is a predetermined voltage.

23. (Original) The microprocessor system of claim 9 in which said microprocessor system is configured to operate at a variable clock speed; said microprocessor system additionally comprising a ring counter variable speed system clock connected to said central processing unit, said central processing unit and said ring counter variable speed system clock being provided in a single integrated circuit, said ring counter variable speed system clock being configured to provide different clock speed to said central processing unit as a result of transistor propagation delays, depending on at least one of temperature of said single integrated circuit, voltage and microprocessor fabrication process for said single integrated circuit.

24. (Original) The microprocessor system of claim 23 additionally comprising an input/output interface connected between said microprocessor system and an external memory bus to exchange coupling control signals, addresses and data between said central processing unit and said input/output interface, and a second clock independent of said ring counter variable speed system clock connected t said input/output interface to provide clock signals for operation of said input/output interface asynchronously from said central processing unit.

25. (Original) The microprocessor system of claim 24 in which said second clock is a fixed frequency clock.

26. (Original) The microprocessor system of claim 9 in which said first push down stack has a first plurality of stack registers having stack memory elements configured as latches, a second plurality of stack registers having stack memory elements configured as a random access memory, said first and second plurality of stack registers and said central processing unit being provided in a single integrated circuit with a top one of said second plurality of stack registers being connected to said a bottom one of said first plurality of stack registers, and a third plurality of stack registers having stack memory elements configured as a random access memory external to said single integrated circuit, with a top one of said third plurality of stack registers being connected to a bottom one of said second plurality of stack registers, said microprocessor system being configured to operate said first, second and third plurality of stack registers hierarchically as interconnected stacks.

27. (Original) The microprocessor system of claim 26 additionally comprising a first pointer connected to said first plurality of stack registers, a second pointer connected to said second plurality of stack registers, and a third pointer connected

to said third plurality of stack registers, said microprocessor system being configured to operate said first, second and third plurality of stack registers hierarchically as interconnected stacks by having said central processing unit being connected to pop items from said first plurality of stack registers, said first stack pointer being connected to said second stack pointer to pop a first plurality of items from said second plurality of stack registers when said first plurality of stack registers are empty from successive pop operations by said central processing unit, said second stack pointer being connected to said third stack pointer to pop a second plurality of items from said third plurality of stack registers when said second plurality of stack registers are empty from successive pop operations by said central processing unit.

28. (Original) The microprocessor system of claim 9 additionally comprising a first register connected to supply a first input to said arithmetic logic unit, a first shifter connected between an output of said arithmetic logic unit and said first register, a second register connected to receive a starting polynomial value, an output of said second register being connected to a second shifter, a least significant bit of said second register being connected to said arithmetic logic unit, a third register connected to supply feedback terms of a polynomial to said arithmetic logic unit, a down counter, for counting down a number corresponding to digits of a polynomial to be generated, connected to said arithmetic logic unit, said arithmetic logic unit being responsive to a polynomial instruction to carry out an exclusive OR of the contents of said first register with the contents of said third register if the least significant bit of said second register is a "ONE" and to pass the contents of said first register unaltered if the least significant bit of said second register is a "ZERO" until said down counter completes a count, the polynomial to be generated resulting in said first register.

29. (Original) The microprocessor system of claim 28 in which said first register is a result register, said first shifter is a left shifting shifter, said second register is a multiplier register connected to receive a multiplier in bit reversed form, said second shifter is a right shifting shifter, said third register is connected to supply a multiplicand to said arithmetic logic unit, said down counter is configured for counting down a number corresponding to one less than the number of digits of the multiplier, said arithmetic logic unit being responsive to a multiply instruction to add the contents of said result register with the contents of said third register, if the least significant bit of said second register is a "ONE" and to pass the contents of said first register unaltered if the least significant bit of said second register is a "ZERO" until said down counter completes a count, the product resulting in said first register.

30. (Previously Presented) The microprocessor system of claim 1 wherein said central processing unit integrated circuit includes means for executing said multiple sequential instructions prior to said fetching means fetching next multiple sequential instructions.

31. (Previously Presented) The microprocessor system of claim 30 wherein said means for executing are configured to determine whether to execute said multiple sequential instructions prior to said fetching means fetching said next multiple sequential instructions based on said multiple sequential instructions.

32. (Previously Presented) The microprocessor system of claim 1 wherein said means for fetching are configured to selectively fetch next multiple sequential instructions in response to execution of said multiple sequential instructions.

33. (Previously Presented) The microprocessor system of claim 32 wherein said means for fetching are configured to fetch said next multiple sequential

instructions prior to completion of execution of said multiple sequential instructions.

34. (Previously Presented) The microprocessor system of claim 1 wherein said central processing unit integrated circuit includes an instruction register configured to store said multiple sequential instructions, said central processing unit integrated circuit being configured to access an operand located in a first instruction location of the instruction register in response to an instruction of the multiple sequential instructions in a second instruction location of the instruction register distinct from the first instruction location.

35. (Previously Presented) The microprocessor system of claim 34 wherein said central processing unit integrated circuit is configured to access the operand in response to an op-code of the instruction in the second instruction location.

36. (Previously Presented) The microprocessor system of claim 1 further comprising an instruction register configured to store the multiple sequential instructions in corresponding instruction locations including a particular location for storing an instruction to be executed, the central processing unit integrated circuit being configured to respond to content of an instruction of the multiple sequential instructions by accessing the particular location of the instruction register.

37. (Previously Presented) The microprocessor system of claim 36 wherein the central processing unit integrated circuit is configured to respond to content of the instruction of the multiple sequential instructions by accessing the particular location of the instruction register after the means for fetching fetches next multiple sequential instructions.

38. (Previously Presented) The microprocessor system of claim 36 wherein the central processing unit integrated circuit is configured to respond to content of the instruction of the multiple sequential instructions by accessing the first-execution location of the instruction register without the fetching means fetching next multiple sequential instructions.

39. (Previously Presented) The microprocessor system of claim 36 wherein the content is an op-code.

40. (Previously Presented) The microprocessor system of claim 1 wherein the multiple sequential instructions comprise a first plurality of sequential instructions arranged from beginning to ending positions of the first plurality of sequential instructions, the central processing unit integrated circuit being configured to respond to content of a first instruction of the first plurality of sequential

instructions by accessing a second instruction in a second plurality of sequential instructions arranged from beginning to ending positions of the second plurality of sequential instructions, the second instruction being in the beginning position of the second plurality of sequential instructions.

41. (Previously Presented) The microprocessor system of claim 40 wherein the second plurality of sequential instructions is distinct from the first plurality of sequential instructions.

42. (Previously Presented) The microprocessor system of claim 40 wherein the second plurality of sequential instructions is the first plurality of sequential instructions and the first instruction is disposed in a position other than the beginning position of the first plurality of instructions.

43. (Previously Presented) The microprocessor system of claim 40 wherein the content is an op-code.

44. (Previously Presented) The microprocessor system of claim 1 wherein the central processing unit integrated circuit includes an instruction register having a plurality of instruction locations for storing the multiple sequential instructions according to an order, the plurality of instruction locations including a first location to be accessed before any other of the plurality of instruction locations, the central processing unit integrated circuit further including means for accessing a next instruction out of the order, the next instruction being located at the first location.

45. (Previously Presented) The microprocessor system of claim 1 wherein the central processing unit integrated circuit includes an instruction register having a plurality of instruction locations for storing the multiple sequential instructions, the

plurality of instruction locations including a first location to be accessed before any other of the plurality of instruction locations, the central processing unit integrated circuit further including means, responsive to content of an instruction of the multiple sequential instructions in a location other than the first location, for accessing a next instruction at the first location.

46. (Previously Presented) The microprocessor system of claim 1 wherein said central processing unit integrated circuit includes a program counter comprising address bits, said fetching means configured to locate the multiple sequential instructions using the address bits from the program counter.

47. (Previously Presented) The microprocessor system of claim 46 wherein the address bits are a most significant bit portion from the program counter.

48. (Previously Presented) The microprocessor system of claim 47 wherein the central processing unit integrated circuit is configured to increment the address bits of the program counter after said means for fetching multiple sequential instructions fetches the multiple sequential instructions.

49. (Previously Presented) The microprocessor of claim 47 wherein the most significant bit portion is 30 of 32 bits of the program counter.

50. (Previously Presented) The central processing unit integrated circuit of claim 47 further comprising an instruction register having a plurality of instruction locations for storing the multiple sequential instructions, and multiplexer means connected to said instruction register for selectively supplying multiple instructions from said instruction register.

51. (Previously Presented) The microprocessor system of claim 47 wherein the multiple sequential instructions comprise a first plurality of sequential instructions, the central processing unit integrated circuit being configured to respond to content of a first instruction of the first plurality of sequential instructions by accessing a second plurality of sequential instructions using an address specified by the address bits.

52. (Previously Presented) The microprocessor system of claim 51 wherein the second plurality of sequential instructions is distinct from the first plurality of sequential instructions.

53. (Previously Presented) The microprocessor system of claim 51 wherein the content is an op-code.

54. (Previously Presented) The microprocessor system of claim 47 further comprising an instruction register having a plurality of instruction locations ordered from a beginning instruction location to an ending instruction location, wherein the central processing unit integrated circuit is configured to respond to content in an instruction location other than the beginning instruction location by accessing the beginning instruction location.

55. (New) The microprocessor system of claim 1 in which said microprocessor system is configured to provide different memory access timing for different storing capacity sizes of said external memory by including a sensing circuit and a driver circuit, and an output enable line connected between said external access memory, said sensing circuit and said driver circuit, said sensing circuit being configured to provide a ready signal when said output enable line reaches a predetermined electrical level after a memory read operation as a function of different capacitance

on said bus as a result of the different storing capacity sizes of said external memory, said microprocessor system being configured so that said driver circuit provides an enabling signal on said output enable line responsive to the ready signal.

56. (New) The microprocessor system of claim 55 in which the predetermined electrical level is a predetermined voltage.

57. (New) The microprocessor system of claim 1 in which said microprocessor system is configured to operate at a variable clock speed; said microprocessor system additionally comprising a ring counter variable speed system clock connected to said central processing unit integrated circuit, said central processing unit integrated circuit and said ring counter variable speed system clock being provided in a single integrated circuit, said ring counter variable speed system clock being configured to provide different clock speed to said central processing unit integrated circuit as a result of transistor propagation delays, depending on at least one of temperature of said single integrated circuit, voltage and microprocessor fabrication process for said single integrated circuit.

58. (New) The microprocessor system of claim 57 additionally comprising an input/output interface connected between said microprocessor system and an external memory bus to exchange coupling control signals, addresses and data between said central processing unit integrated circuit and said input/output interface, and a second clock independent of said ring counter variable speed system clock connected to said input/output interface to provide clock signals for operation of said input/output interface asynchronously from said central processing unit.

59. (New) The microprocessor system of claim 58 in which said second clock is a fixed frequency clock.

60. (New) The microprocessor system of claim 1 in which said first push down stack has a first plurality of stack registers having stack memory elements configured as latches, a second plurality of stack registers having stack memory elements configured as a random access memory, said first and second plurality of stack registers and said central processing unit integrated circuit being provided in a single integrated circuit with a top one of said second plurality of stack registers being connected to said a bottom one of said first plurality of stack registers, and a third plurality of stack registers having stack memory elements configured as a random access memory external to said single integrated circuit, with a top one of said third plurality of stack registers being connected to a bottom one of said second plurality of stack registers, said microprocessor system being configured to operate said first, second and third plurality of stack registers hierarchically as interconnected stacks.

61. (New) The microprocessor system of claim 60 additionally comprising a first pointer connected to said first plurality of stack registers, a second pointer connected to said second plurality of stack registers, and a third pointer connected to said third plurality of stack registers, said microprocessor system being configured to operate said first, second and third plurality of stack registers hierarchically as interconnected stacks by having said central processing unit integrated circuit being connected to pop items from said first plurality of stack registers, said first stack pointer being connected to said second stack pointer to pop a first plurality of items from said second plurality of stack registers when said first plurality of stack registers are empty from successive pop operations by said central processing unit, said second stack pointer being connected to said third stack

pointer to pop a second plurality of items from said third plurality of stack registers when said second plurality of stack registers are empty from successive pop operations by said central processing unit.

REMARKS

This amendment is filed responsive to the office action mailed November 19, 2009 in the merged Reexamination Proceedings. Claims 1-54 were previously presented or original. Claims 55-61 are new. Claims 1-27 and 30-61 are subject to re-examination. Claims 28 and 29 are not subject to reexamination. Claims 21-27, which depend from claim independent claim 9, are confirmed. Claims 1-20 and 30-54 are rejected.

Claims 55-61 are new, but correspond to confirmed claims 21-27. The new claims are substantially similar to the confirmed claims except for the change in dependency from claim 9 to claim 1, and appropriate wording changes to "memory" and "central processing unit." The new claims are supported in the specification at least by original claims 21-27.

In the action of November 19, 2009, the examiner confirmed claims 21-27 and stated his reasons for patentability. Those reasons are equally applicable to new claims 55-61. Patent owner submits that the new claims 55-61 are in condition for allowance.

The rejections

Claims 30 and 31 are rejected under 35 U.S.C. § 112, first paragraph, as failing to comply with the written description requirement.

Claims 1, 9 and 18-20, are rejected under 35 U.S.C. §102(b) as being anticipated by IMS T414 transputer datasheet.

Claims 1-2, 7, 8, 30-53, are rejected under 35 U.S.C. §102(b) as anticipated by US patent 4,758,948 to May et al., a patent on an improved transputer.

Claims 1-6 and 8, are rejected under 35 U.S.C. §102(b) as anticipated by McGregor, an article describing the Motorola MC68020 microprocessor.

Claims 1 and 8 are rejected under 35 U.S.C. §102(e) as being anticipated by US patent 4,980,821 to Koopman et al.

Claims 10-12 are rejected under 35 U.S.C. §103(a) as obvious over the T414 data sheet in view of May '948.

Claims 9-17 are rejected under 35 U.S.C. §103(a) as being obvious over McGregor in view of the MC68020 32-bit microprocessor user's manual and further in view of US patent 4,985,848 to Pfeiffer et al.

Rejections Under 35 U.S.C. § 112

New **claims 30 and 31** currently read as follows:

> 30. (New) The microprocessor system of claim 1 wherein said central processing unit integrated circuit includes means for executing said multiple sequential instructions prior to said fetching means fetching next multiple sequential instructions.
>
> 31. (New) The microprocessor system of claim 30 wherein said means for executing are configured to determine whether to execute said multiple sequential instructions prior to said fetching means fetching said next multiple sequential instructions based on said multiple sequential instructions.

Claims 30 and 31 stand rejected under 35 USC section 112, paragraph 1, as failing to comply with the written description requirement. The Office Action states:

13. Particularly, the newly presented ***claims 30 and 31*** require that a means for executing multiple sequential instructions prior to fetching the next multiple sequential instructions. This feature is not believed to be expressly described in the specification of the '749 Patent. The Patent Owner states that the features are taught on col. 7, line 63- col. 8, line 16 of the '749 Patent. But in this section, the '749 Patent states "While the current instructions in instruction register 108 are executing, the memory controller obtains the address of the next set of four instructions...and obtains that set of instructions." Thus, with this, the executing appears to be at the same time as the fetching, and not "prior to fetching the next multiple sequential instructions."

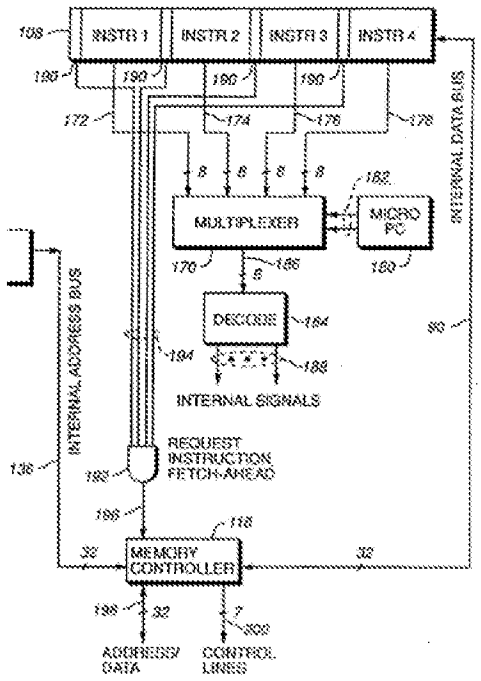


Fig. 4, US'749

Having certain instructions executed before a next fetch depends on specific conditions

As the Examiner correctly points out, in some circumstances, execution occurs at the same time as the fetching of the next multiple sequential instructions. However, claims 30 and 31 are directed to a different feature of preventing the fetching of the next multiple sequential instructions if any of the instructions currently in the instruction register requires a memory reference. This feature is also supported by the quoted passage of the '749 patent.

The quoted passage states that the four instructions present in the instruction register are analyzed by decoder 192 to determine whether any of the instructions require a memory reference (such as for example when an instruction requires an operand to be fetched from memory, or when the instruction is a branch instruction). *If* none of the instructions require a memory access, then the decoder 192 requests an instruction fetch ahead. *If any,* of the instructions require a memory access, then decoder 192 does not request the instruction fetch ahead. Thus overlapping of execution and instruction fetch occurs only when there are no instructions in the instruction register that require a memory reference. In contrast, execution is not overlapped if any of the instructions requires a memory reference. In essence, the decoder 192 prevents instruction fetch ahead if any of the instructions in the instruction register requires a memory reference. This provides clear written description support for claims 30 and 31.

Rejections Under 35 U.S.C. §102

The T414 datasheet does not anticipate claims 1, 9 or 18-20

The office action rejected **claims 1, 9 and 18-20** as being anticipated by "IMS T414 Transputer Datasheet" published by INMOS, Ltd., February 1987 (hereinafter the "T414 datasheet").

Regarding **claim 1**, rejection in part states that the claim element "said means for fetching instructions being configured and connected to fetch multiple sequential instructions from said memory in parallel and supply the multiple sequential instructions to said central processing unit integrated circuit during a single memory cycle" is disclosed in the T414 datasheet at page 5, wherein the reference states "as memory is word accessed the processor will receive several instructions for every fetch", also wherein "the process is a sequence of instructions. A transputer can run several process in parallel (concurrently)."; also see page 3, wherein "... six registers are used in the execution of a sequential process.", and also wherein "Each instruction consists of a single byte divided into two 4 bit parts."

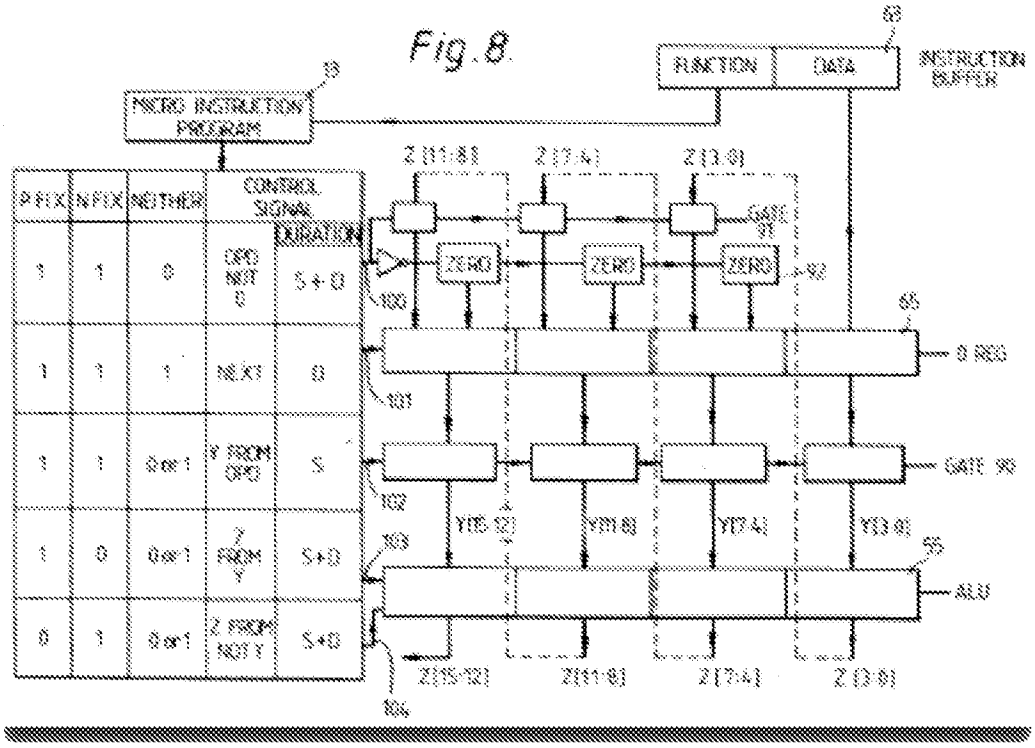
Regarding claim nine, the rejection in part states that "said means for fetching instructions being configured to fetch multiple sequential instructions from said dynamic random access memory in parallel and supply the multiple instructions to said central processing unit during a single memory cycle [see page 5, wherein "as memory is word accessed the processor will receive several instructions for every fetch", also wherein "a process is a sequence of instructions. A transputer can run several processes in parallel (concurrently)."; also see page 3, wherein "... six registers are used in the execution of a sequential process.", and also wherein "Each instruction consists of a single byte divided into two 4 bit parts."]

The T414 data sheet does not teach or disclose the fetching of multiple sequential instructions in parallel and supplying them to a CPU during a single memory cycle

Both claims 1 and 9 require the **fetching** in parallel **and** the **supplying** of multiple sequential instructions from a memory to a "CPU" over a bus connecting the "memory" and "CPU" **during a single memory cycle**. The office action cites the T414 data sheet for showing that, "the processor will receive several instructions for every fetch." The cited passage also states that the instructions are first supplied to a prefetch buffer. See, page 5 of the T414 datasheet, which, "There is an extra word of pre-fetch buffer so the processor rarely has to wait for an instruction fetch before proceeding. Since this buffer is short, there is little time penalty when a jump instruction causes the buffer contents to be discarded."

The data sheet has no clear discussion of how the T414 supplies the instructions to the CPU. Patent owners argue that the Transputer T414 appears to operate in the same way as May. Indeed, May states at col. 5, lines 19-21 that the group of patent applications listed at col. 5, lines 7-17, which include Edwards, describes the Transputer, and May describes an improved Transputer. It goes on to list the improvements, none of which pertain to how instructions are obtained from an instruction buffer and then decoded. Such details are said to be described in the referenced applications.

Edwards describes the way the Transputer decodes and executes instructions. As described in Edwards, see, e.g., Fig. 8, below, instructions are supplied to a one-instruction-wide instruction buffer, one at a time, and are there decoded. Fetching multiple instructions into a prefetch buffer and then supplying them one at a time is not sufficient to meet the claim limitation - the supplying of "multiple sequential instructions to a CPU during a single memory cycle."



Edwards, US'698 fig. 8

Because the cited prior art does not disclose each and every element of claim 1 or claim 9, no prima facie case of anticipation is established with respect to claim 1 or claim 9. Therefore, Patent Owners respectfully request reconsideration and withdrawal of the rejections of claim 1 and claim 9 under 35 U.S.C. §102(b) over the IMS T414 Transputer Data Sheet.

Claims 18 and 19, both depend from a patentable claim 9 and therefore are themselves patentable for at least the same reasons as claim 9.

May does not anticipate claims 1-2, 7, 8, and 30-53.

Claims **1-2, 7, 8, and 30-53 are rejected under 35 U.S.C. §102(b)** as anticipated by US patent number 4,758,948 issued to May et al. on July 19, 1988. The office action notes that May incorporates by reference US patent number 4, 680, 698 issued Edward et al. on July 14, 1987.

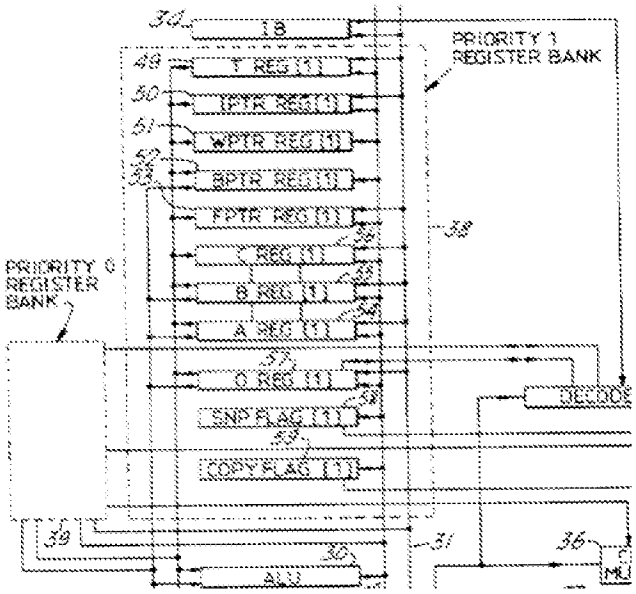
Regarding **claims 1 and 8**, the rejection in part states,

> said means for fetching instructions being configured and connected to fetch multiple sequential instructions from said memory in parallel and supply the multiple sequential instructions to said central processing unit integrated circuit during a single memory cycle [col. 7, lines 26-27, wherein "each instruction consists of 8 bits..."; also see col. 6, lines 10-13, wherein "The processor accesses the memory in words...(in this particular example 16 bits).";
>
> also see col. 7, lines 22-25, wherein "Communications between the CPU and the memory is effected via ...data bus 31."; also see Fig. 2, whereby the bus 31 is split so that instructions are transferred in parallel to another priority register bank;

This rejection is substantially identical to the rejection of claim 1 set forth in the office action mailed March 9, 2009 in ex parte reexamination control number 90/009,034. Patent owner responded on May 4, 2009, traversing the rejection, particularly at pages 26 and 27 of that response. The patent owner hereby incorporates that response and its included traverse in its entirety.

In summary, May, as further explained by Edwards, supplies instructions one by one, even if the access to memory is by word. Therefore, May does not disclose means for fetching configured to supply multiple sequential instructions to a central processing unit during a single memory cycle, as required by claim 1 and claim 8.

Further, even though May provides two sets of certain registers, a first set for priority 0 and a second set for priority 1, as illustrated in figure 2, a portion of which is set forth below, it provides only one instruction buffer, IB 34, common to both priority processes. Instructions are fed to the instruction buffer, not to the other registers. Therefore, the presence of priority 0 registers is irrelevant to the claim limitation concerning in the fetching of multiple sequential instructions … and supplying them to the CPU integrated circuit.



From Fig. 2, May'948

Claims 2, 7 and 30-53 all depend from a patentable claim 1 and are, therefore, patentable for at least the same reasons as claim 1.

Additional reasons for the patentability of **claims 2, 7, 8, 30-39, 44 and 45**:

Claim 2: May has no means to decode multiple instructions to determine whether they require a memory access.

With regard to the **claim 2**, the action states,

means connected to said means for fetching multiple instructions for determining by decoding the multiple instructions if multiple instructions fetched by said means for fetching multiple instructions require a memory access [see col. 7, lines 26-39, wherein "each instruction consists of 8 bits having the format shown in Fig. 7. 4 bits represent the required function of the instruction and 4 bits are allocated for data. Each instruction derived from the program sequence for the process is fed into an instruction buffer 34 and the instruction is decoded by the decoder 35."],

said means for fetching multiple instructions fetching additional multiple instructions if decoding the multiple instructions shows that the multiple instructions do not require a memory access [see col. 9, lines 7-9 and 31-33].

The cited portions of May'948 read in addition to the portions included in the rejection, the following:

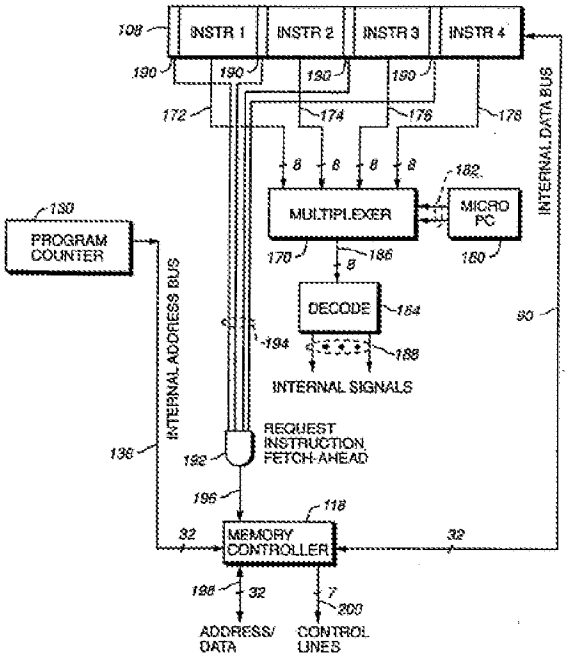
tion buffer 34, decoder 35, condition multiplexor 36, MIR 37, microinstruction ROM 13 and switches 32 are generally as described in the above mentioned patent applications.

The other applications include Edwards, which as noted above, states that the CPU executes one instruction and then fetches the next from memory. May and Edwards have no disclosure of any apparatus that decodes multiple instructions to determine whether any of the instructions require a memory access, and then selectively fetching the next multiple instructions depending on the answer.

Regarding **claim 7**, the office action states in part,

a counter connected to control said means for supplying the multiple instructions to supply the multiple instructions in succession [see col. 35, lines 44-46, wherein "The IPTR REG (0) contains a pointer 180 to the next instruction in the program sequence 181 which is stored in memory", whereby the "instruction pointer IPTR" is considered as a "counter"],

The claim requires that a "counter" be connected "to control" the means for supplying the multiple instructions and so that it may "to supply" multiple instructions, "in succession", to a means for decoding defined in the next claim element. The "means for supplying" is defined in claim 7 to be connected to the instruction register and whose function is "for supplying the multiple instructions in succession from the instruction register." The corresponding structure for the means for supplying is multiplexer 170.

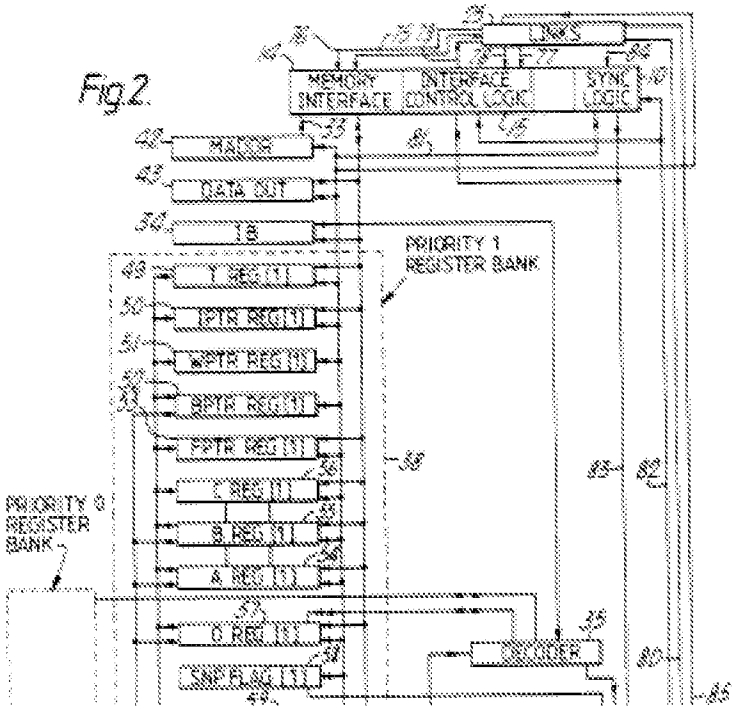


US '749, Fig. 16

The corresponding structure for the counter function is the 2-bit micro PC counter 180 illustrated in Figs. 4, 16 and 20, as described at column 7 lines 58-59, column 16, lines 42-46, col. 18, lines 48-49.

The only counter connected to control the supplying of instructions to a means for decoding is the two bit micro program counter 180. While the program counter 130 is also connected to the memory controller 118, which is part of the means for supplying multiple instructions to the CPU, it has no control relationship with the means so as to cause it to provide the instructions in succession to the decoding means. That functionality is instead provided by the micro program counter 180. The micro program counter 180 controls the multiplexer 170 to provide instructions, one at a time from the instruction register 108 to the decoding means 184.

The office action cites IPTR register (0) and quotes column 35, lines 44-46 as corresponding structure in May because it contains a pointer to the next instruction in the program sequence "stored in memory." But the claim requires the instructions be “supplied” from an instruction register, not memory.



US'948, Fig. 2

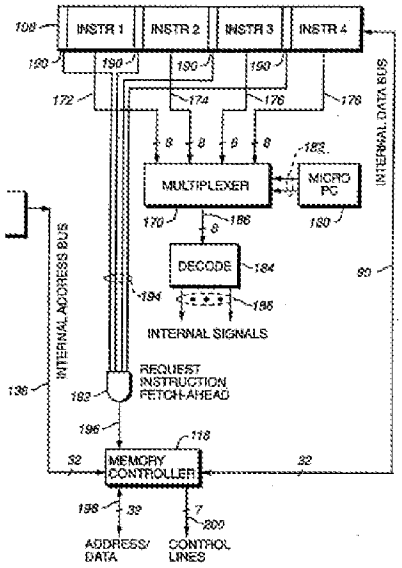
May '948 discusses that instructions are provided sequentially to the instruction buffer 34, i.e., one at a time. Col. 7, 1ines 63-64. May states that its execution structure is described in Edwards. Edwards illustrates in figure 8 that instructions are decoded from the instruction buffer, which is only one instruction wide. Edwards further states, at col. 24, 11. 12-18, that after it decodes an instruction it requests a next instruction from memory. Edwards has no disclosure of any structure connected to an instruction buffer that would provide instructions in succession to a means for decoding. May therefore has no structure corresponding to the claimed "counter."

Regarding **claims 30 and 33**, (each claim contains the same claim element and the same basis for rejection) the office action states:

Regarding ***claim 30***, May'948 discloses the system discussed above in claim 1, and further teaches that said central processing unit integrated circuit includes means for executing said multiple sequential instructions prior to said fetching means fetching next multiple sequential instructions [see col. 21, line 33-col. 22, line 2, wherein "The actions may be performed on behalf of the current process are to perform "StartNextProcess", to perform "BlockCopyStep" or to fetch, decode and execute an instruction.", and also wherein "Otherwise the processor will fetch, decode and execute an instruction if there is a current process."].

Response:

The corresponding structure was discussed above in connection with the Section 112, p.1, rejection of claim 30. It comprises apparatus (AND gate 192 connected to the first bits 190 of each instruction in the instruction register 108)



US'749, Fig. 4

that analyses the instructions in the instruction register to determine whether any of them require a memory reference, and if such an instruction exists, prevents a

request for instruction fetch ahead. This results in all instructions in the instruction register being executed prior to the memory controller fetching the next multiple sequential instructions.

Claim 30 depends from claim 1. The multiple sequential instructions both analyzed and executed are the instructions fetched from external memory in parallel and provided to the CPU integrated circuit in a single memory cycle.

The cited portion of May states that one of three alternative actions is to "fetch, decode and execute an instruction." There is nothing in this passage that suggests executing the "multiple sequential instructions" in the CPU prior to requesting a next multiple sequential instructions from memory. May clearly does not disclose the claimed limitations.

Regarding **claim 31**,

> Regarding *claim 31*, May'948 discloses the system discussed above in claim 30, and further teaches that said means for executing are configured to determine whether to execute said multiple sequential instructions prior to said fetching means fetching said next multiple sequential instructions based on said multiple sequential instructions [see col. 21, line 33-col. 22, line 2].

May makes no determination to fetch a next group of instructions based upon current multiple sequential instructions.

Referring to the discussion above regarding claim 30, claim 31 requires the analysis of the multiple sequential instructions to make the determination. In contrast, the decision whether to fetch or not to fetch in May is determined from a "Condition Multiplexor" that is connected to flags SNPFlag and CopyFlag, which

are bits in a one-bit register 58, see eg, Fig. 21, and col. 8, lines 28-34. Such one-bit registers are not instructions let alone the multiple sequential instructions required by the claim.

Regarding **claim 32**,

> Regarding ***claim 32***, May'948 discloses the system discussed above in claim 1, and further teaches that said means for fetching are configured to selectively fetch next multiple sequential instructions in response to execution of said multiple sequential instructions {see col. 21, line 33-col. 22, line 2, wherein "The actions may be performed on behalf of the current process are to perform "StartNextProcess", to perform "BlockCopyStep" or to fetch, decode and execute an instruction.", and also wherein "Otherwise the processor will fetch, decode and execute an instruction if there is a current process."}.

<u>May does not base a decision to fetch a group of instruction upon execution of a current multiple sequential instructions</u>

The cited passage in May does not disclose that the condition for selectively fetching is based upon the execution of anything but one instruction, nor that anything other than one instruction is fetched and executed. May does not disclose the claim limitation that the decision whether to is in "response to [the] execution of said multiple sequential instructions."

Regarding **claim 34**,

> Regarding ***claim 34***, May'948 discloses the system discussed above in claim 1, and further teaches that said central processing unit integrated circuit includes an instruction register configured to store said multiple sequential instructions [see Fig. 2; also see col. 7, line 41-col. 8, line 34; also see Fig. 3, whereby IPTR 50 is used for the current instruction and IPTR S 65 is used for a process that is not the current process, see col. 9, lines 59-67],
>
> said central processing unit integrated circuit being configured to access an operand located in a first instruction location of the instruction register in response to an instruction of the multiple sequential instructions in a second instruction location of the instruction register distinct from the first instruction location [see col. 7, lines 40-col. 8, line 62; also see col. 9, lines 59-67, wherein "Location 65 is used when a process is not the current process to hold a pointer (IPTR) to the next instruction to be executed by the process when it becomes the current process."].

May's instruction pointers are not instruction registers

IPTR 50 and IPTR S 65 are "instruction pointers" not "instruction registers." They hold the memory address of an instruction, not the instruction. See, e.g., col. 8 at line 10.

10	**IPTR REG**	**A register 50 which holds the instruction pointer (IPTR) of any process indicated by register 51**

The closest analogous structure in May is instruction buffer 34. However, as described above, that buffer holds only one instruction at a time, not multiple instructions at a time.

Regarding **claim 35**,

> Regarding ***claim 35***, May'948 discloses the system discussed above in claim 34, and further teaches that said central processing unit integrated circuit is configured to access the operand in response to an op-code of the instruction in the second instruction location [operand register OREG 57, see col. 8, line 63-col. 9, line 4; also see the Edwards'698 Patent, whereby the various "operation codes" are described in col. 23, lines 2-20, as well as in col. 29, line 28-col. 32, line 24].

Response:

Claim 35 depends from claim 34, a claim patentable for the reasons set forth above, and is therefore patentable for at least the same reasons.

Regarding claim 36,

> Regarding ***claim 36***, May'948 discloses the system discussed above in claim 1, and further teaches of an instruction register configured to store the multiple sequential instructions in corresponding instruction locations including a particular location for storing an instruction to be executed [col. 7, lines 40-col. 8, line 62; also see col. 9, lines 59-67, wherein "Location 65 is used when a process is not the current process to hold a pointer (IPTR) to the next instruction to be executed by the process when it becomes the current process."], the central processing unit integrated circuit being configured to respond to content of an instruction of the multiple sequential instructions by accessing the particular location of the instruction register [see col. 7, line 56-col. 8, line 34; also see col. 9, lines 59-67].

May's instruction pointers are not instruction registers

Claim 36 requires an instruction "register" storing multiple sequential instructions in locations such that when certain instructions are executed (such as a skip, microloop or branch instruction) the next instruction is accessible by accessing a particular location in the instruction register. As described for example at col. 16, lines 40-53, skip and micro loop instructions both include resetting the 2-bit microinstruction (AKA, micro PC) counter 180 to zero, which causes the next instruction access to access a next instruction located at the INSTR 1 location of the instruction register.

As described above in connection with claim 34, IPTR 50 and IPTR S 65 are not instruction registers. They are instruction pointers and hold the address of instructions, not the instructions themselves. May does not disclose an instruction register that holds multiple sequential instructions, let alone an instruction and means that will access a next instruction at a particular location of the instruction register.

Regarding **claim 37**,

Regarding ***claim 37***, May'948 discloses the system discussed above in claim 36, and further teaches that the central processing unit integrated circuit is configured to respond to content of the instruction of the multiple sequential instructions by accessing the particular location of the instruction register after the means for fetching fetches next multiple sequential instructions [see col. 7, lines 40-col. 8, line 62; also see col. 9, lines 59-67, wherein "Location 65 is used when a process is not the current process to hold a pointer (IPTR) to the next instruction to be executed by the process when it becomes the current process."].

Response:

Claim 37 depends from claim patentable claim 36 and is, therefore, patentable for at least the same reasons as claim 36.

Regarding **claim 38**,

Regarding ***claim 38***, May'948 discloses the system discussed above in claim 36, and further teaches that the central processing unit integrated circuit is configured to respond to content of the instruction of the multiple sequential instructions by accessing the first-execution location of the instruction register without the fetching means fetching next multiple sequential instructions [see col. 7, lines 40-col. 8, line 62; also see col. 9, lines 59-67, wherein "Location 65 is used when a process is not the current process to hold a pointer (IPTR) to the next instruction to be executed by the process when it becomes the current process."].

Response:

Claim 38 depends from claim patentable claim 36 and is, therefore, patentable for at least the same reasons as claim 36.

Regarding **claim 39**,

> Regarding ***claim 39***, May'948 discloses the system discussed above in claim 36, and further teaches that the content is an op-code [see the Edwards '698 Patent, whereby the various "operation codes" are described in col. 23, lines 2-20, as well as in col. 29, line 28-col. 32, line 24].

Response:

Claim 39 depends from claim patentable claim 36 and is, therefore, patentable for at least the same reasons as claim 36.

Claims 40-43 depend, either directly or indirectly, from patentable claim 1 and are, therefore, patentable for at lease the same reasons as claim 1.

Regarding **claim 44**,

Regarding ***claim 44***, May'948 discloses the system discussed above in claim 1, and further teaches that the central processing unit integrated circuit includes an instruction register having a plurality of instruction locations for storing the multiple sequential instructions according to an order [see col. 7, line 40-col. 8, line 67; also see Fig. 3; also see col. 35, lines 12-53, wherein "...an instruction pointer (IPTR) in the IPTR REG 50 indicates the next instruction to be executed from the sequence of instructions in the program relating to that particular process....When a process is scheduled it either becomes the current process or is added to a list or queue of processes waiting execution. Such a list is formed as a linked list with each process on the list having a pointer in the link location 66 of its workspace to the workspace of the next process on that list."],

May's instruction pointers are not instruction registers

As noted above, IPTR 50 is an instruction pointer that contains the address of an instruction located in memory. It is not a register that holds instructions.

May does not have "an instruction register having a plurality of instruction locations for storing the multiple sequential instructions....", as recited in claim 44.

Regarding **claim 45**,

> Regarding ***claim 45***, May'948 discloses the system discussed above in claim 1, and further teaches that
>
> the central processing unit integrated circuit includes an instruction register having a plurality of instruction locations for storing the multiple sequential instructions, the plurality of instruction locations including a first location to be accessed before any other of the plurality of instruction locations [see col. 35, lines 12-53, wherein "...an instruction pointer (IPTR) in the
>
> IPTR REG 50 indicates the next instruction to be executed from the sequence of instructions in the program relating to that particular process.... When a process is scheduled it either becomes the current process or is added to a list or queue of processes waiting execution. Such a list is formed as a linked list with each process on the list having a pointer in the link location 66 of its workspace to the workspace of the next process on that list.", also wherein "the processor maintains two lists of processes which are waiting to be executed, one for each priority level."; also see the various examples in Figs. 16A-21C],

May's instruction pointers are not instruction registers

As noted above, IPTR 50 is an instruction pointer that contains the address of an instruction located in memory. It is not a register that holds instructions.

May does not have "an instruction register having a plurality of instruction locations for storing the multiple sequential instructions....", as recited in claim 45.

Regarding **claim 46**,

> Regarding ***claim 46***, May'948 discloses the system discussed above in claim 1, and further teaches that said central processing unit integrated circuit includes a program counter comprising address bits [whereby the "instruction pointer IPTR" is considered as a counter], said fetching means configured to locate the multiple sequential instructions using the address bits from the program counter [see col. 35, line 12-col. 36, line 2].

Claims 46-54 depend from patentable claim 1 and are, therefore, patentable for at least the same reasons as claim 1.

MacGregor

Claims 1-6, and 8 are rejected under 35 U.S.C. §102(b) as being anticipated by MacGregor. The Examiner writes (in part):

> 17. **Claims 1-6 and 8** are rejected under 35 U.S.C. 102(b) as being anticipated by "The Motorola MC68020", written by Doug MacGregor *et al*., IEEE Micro, August 1984, pages 101-118 (hereafter "MacGregor") in view of the reference "MC68020 32-Bit Microprocessor User's Manual", published by Motorola, having a copyright dated of 1984 (hereafter the "MC68020 User's Manual").

Patent Owners respectfully traverse.

Regarding **claims 1 and 8**, the office action states:

> said means for fetching instructions being configured and connected to fetch multiple sequential instructions from said memory in parallel and supply the multiple sequential instructions to said central processing unit integrated circuit during a single memory cycle [see page 107, wherein "The cache interface to the processor data paths allows complete overlap of instruction fetches with data operand accesses, and thus provides a significant increase in performance. If simultaneous instruction and data operand requests are generated by the micromachine, a hit in the instruction cache allows concurrent fetches to take place."; also see page 111, wherein "Figure 10 illustrates that data and instruction addresses are calculated in parallel and have separate paths to the address pads. This allows a simultaneous instruction and data access if there is a hit in the cache while a data access is taking place."],

MacGregor supplies the CPU only one instruction at a time

The passage of MacGregor cited in the office action relates to the parallel access of both data and instructions, but not to the parallel fetching of instructions. The reference does not suggest the parallel fetching of multiple instructions from memory.

Elsewhere, MacGregor discusses fetching of instructions. However, it does not disclose fetching "multiple sequential instructions from said memory in parallel and supply the multiple sequential instructions to said central processing unit integrated circuit during a single memory cycle". MacGregor might imply that it fetches two instructions from memory at a time, but the instructions are supplied to the CPU one at a time. Such non-parallel supplying of instructions to the CPU is not supplying them to the CPU during a single memory cycle as required by the claim. MacGregor, at page 111, states,

> However, there is an appreciable gain from the 32 bit bus for instruction accesses. Although instructions are of

> word length and are fetched a word at a time in the M68000, there is a benefit in fetching two words at a time, because the instruction stream is sequential in nature. The MC68020 stores the second word in a **temporary register**. This implies that even with the cache disabled or while executing in-line code, the MC68020 will realize a reduction in the number of bus cycles approaching 50 percent. (emphasis supplied)

What MacGregor discloses is a "temporary register" for the second of two instructions in a "longword," the term used by MacGregor to describe 32-bits. The first is supplied to the CPU. When it is finished executing, the second is supplied to the CPU. Both words are not supplied to the CPU in parallel, during a single memory cycle as required by claims 1 and 8.

Claims 2-4 depend from patentable claim 1 and are, therefore distinguished over MacGregor for at least the same reasons as claim 1.

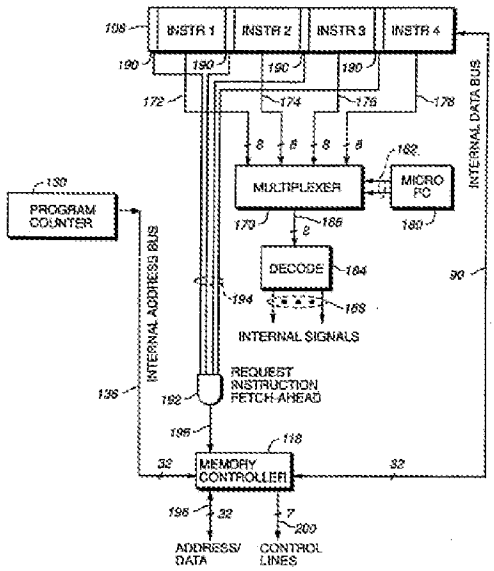
Regarding claim 5, the office action states:

> Regarding ***claim 5***, MacGregor discloses the system discussed above in claim 1, and further teaches of an instruction register for the multiple instructions connected to said means for fetching instructions [see page 112, wherein "The depth of the instruction pipe on the MC68020 is three words."; also see Fig. 12 on page 114],
>
> means connected to said instruction register for supplying the multiple instructions in succession from said instruction register, a counter connected to control said means for supplying the multiple instructions to supply the multiple instructions in succession [see Figure 1 on page 104, having a "program counter"; also see page 102, wherein "The MC68000 family supports 14 addressing modes including ...program-counter-relative"],

MacGregor's program counter does not supply instructions to a decoder from an instruction register

The claim requires that a "counter" be connected "to control" the means for supplying the multiple instructions "to supply" multiple instructions, "in succession", to a means for decoding defined in the next claim element. The "means for supplying" is defined in claim 5 to be connected to the instruction register and whose function is "for supplying the multiple instructions in succession from the instruction register."

The corresponding structure for the means for supplying is multiplexer 170.



US '749, Fig. 16

The corresponding structure for the counter function is the 2-bit micro PC counter 180 illustrated in Figs. 4, 16 and 20, as described at column 7, lines 58-59, column 16, lines 42-46, col. 18, lines 48-49. The only counter connected to control the supplying of instructions to a means for decoding is the two bit micro program counter 180. While the program counter 130 is also connected to the memory controller 118, which is part of the means for supplying multiple instructions to the CPU, it has no control relationship with the means so as to cause it to provide the instructions in succession to the decoding means. That functionality is instead provided by the micro program counter 180. The micro program counter 180 controls the multiplexer 170 to provide instructions, one at a time from the instruction register 108 to the decoding means 184.

The office action cites the three stage pipeline of the Motorola MC 68020 because page 112 of MacGregor describes the pipelines as being "three words

deep." However, the Motorola MC 68020 pipeline comprises three, serial-linked stages, each stage of which holds one 16 bit instruction. See the Motorola user's manual at figure 1-5 and the description of the pipeline at page 1-8.

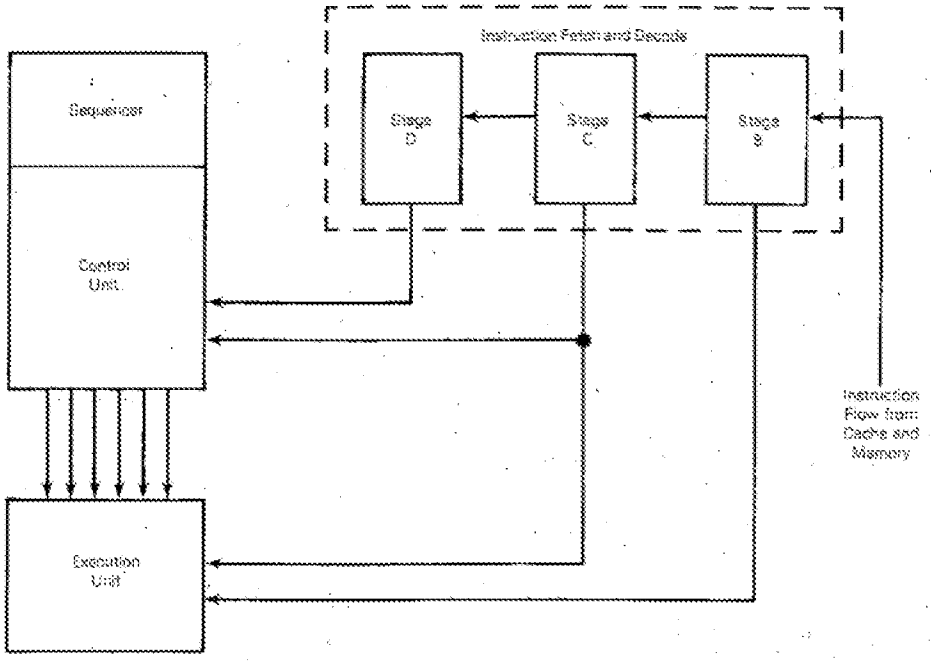


Figure 1-5. MC68020 Pipeline

Instructions are loaded from the on-chip cache or from external memory during instruction prefetch into stage B. The instructions are sequenced from stage B through stage C to D. Stage D presents a fully decoded and validated instruction to the control unit for execution. Instructions with immediate data and extension words find these words already loaded in stage C and ready for use by the control and execution units.

Motorola MC 68020 user's manual at 1-8

From this, it is apparent that MacGregor does not disclose a multiple-instruction instruction register. Furthermore, MacGregor does not disclose anything corresponding to the means for supplying the multiple instructions in

succession from the instruction register, nor a counter connected to control the means for supplying to supply the multiple instructions in succession to a decoding means, all as required by claim 5.

Claim 6 depends from claim 5, which is patentable over MacGregor, and is therefore patentable over MacGregor for at least the same reasons as claim 5.

Koopman

Koopman does not anticipate claims 1 and 8. Koopman fetches and supplies instructions one at time.

Claims 1 and 8 are rejected under 35 U.S.C. § 102(e). The office action states:

20. **Claims 1 and 8** are rejected under 35 U.S.C. 102(e) as being anticipated by U.S. Patent Number 4,980,821, issued to Koopman *et al.* on December 25, 1990, being filed on March 24, 1987 (hereafter "Koopman'821").

Regarding claim 1, the office action specifically rejects claim 1 as follows:

a central processing unit integrated circuit [ALU 60, see Fig. 1B],

a memory external of said central processing unit integrated circuit [microprogram memory 78, see Fig. 1B],

a bus connecting said central processing unit integrated circuit to said memory [bidirectional system bus 32, see Fig. 1A], and

means connected to said bus for fetching instructions for said central processing unit integrated circuit on said bus from said memory [host PC bus interface 36],

said means for fetching instructions being configured and connected to fetch multiple sequential instructions from said memory in parallel and supply the multiple sequential instructions to said central processing unit integrated circuit during a single memory cycle [see col. 8, lines 11-48, wherein "...a microinstruction pre-fetch is used. This means that the next microinstruction is being read from micromemory at the same time the current instruction is being executed."], said bus having a width at least equal to a number of bits in each of the instructions times a number of the instructions fetched in parallel [see col. 4, lines 20-44],

Regarding claim 8, the office action specifically rejects claim 8 as follows:

> a central processing unit [see Fig. 1B],
>
> a memory [microprogram memory 78, see Fig. 1B],
>
> a bus connecting said central processing unit to said memory [bidirectional system bus 32, see Fig. 1A], and
>
> means connected to said bus for fetching instructions for said central processing unit on said bus from said memory, said means for fetching instructions being configured and connected to fetch multiple sequential instructions from said memory in parallel and supply the multiple sequential instructions to said central processing unit during a single memory cycle [see col. 8, lines 11-48, wherein "...a microinstruction pre-fetch is used. This means that the next microinstruction is being read from micromemory at the same time the current instruction is being executed."],

Both claims 1 and claim 8 require similar claim elements, and the rejections are substantially similar. Furthermore, the rejection of claim 1 is virtually identical to the rejection of claim 1 entered in the office action mailed March 4, 2009 in Control Number 90/009,034.

The patent owner traverses the rejection of claims 1 and 8 for the same reasons set forth in their response to the March 4, 2009 office action filed by the patent owner on May 4, 2009, particularly at pages 27-35 thereof. Patent owner hereby incorporates by reference that response and in particular pages 27-35 thereof.

As described at page 34 of the prior response, Koopman does not have a "means to supply sequential instructions in a single memory cycle." Rather it fetches single 8-bit instructions, one at a time from RAM, along a "direct path" separate from the bidirectional system bus 32, and provides them one at a time to the microinstruction execution unit. This is particularly described in connection with the operation of the decoder illustrated in Fig. 14.

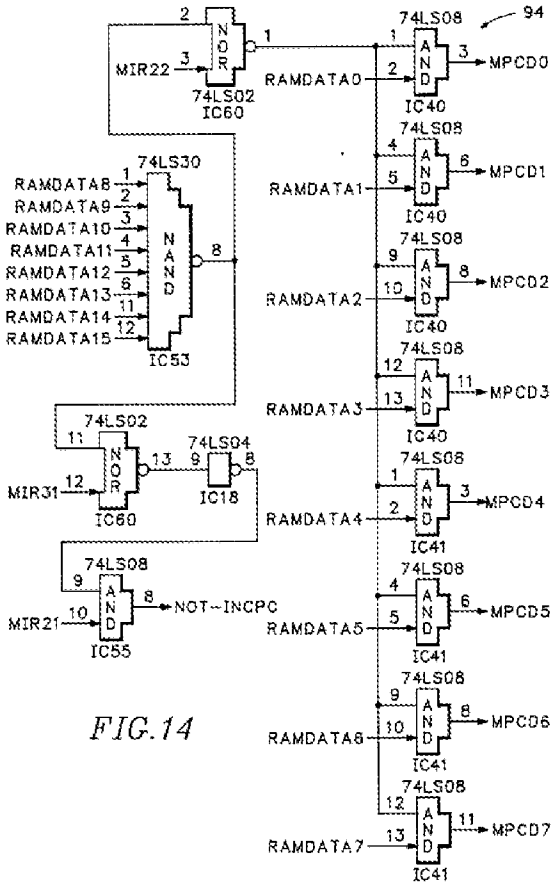


FIG. 14

US'821, Fig. 14

Here the 16 bits from RAM, RAMDATA 0-15, are provided as inputs to the decoder. If bits RAMDATA8-15 are all 1's, bits RAMDATA0-7 are gated to the MPC,

and are used to index the microinstruction memory. Col. 7, line 55- col. 8, line 10. Koopman at col. 7, lines 67 describes the entire 16 bits as the "op-code." Col. 7, line 67. An op-code is an instruction in Koopman. Koopman fetches only one op-code, not multiple op-codes, at a time "**from said memory**". Koopman has no disclosure of fetching multiple sequential instructions, in parallel, from a memory and supplying them to a CPU during a single memory cycle as required by claims 1 and 8.

The Office Action cites the overlapping fetch of microinstructions from microinstruction memory while a current microinstruction is being executed for this claim element. The overlapping fetching and execution is not "fetching **in parallel** and supplying multiple sequential instructions to a CPU during a single memory cycle."

Rejections Under 35 U.S.C. §103(a)

The office action provides:

Claim Rejections - 35 USC § 103

21. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

22. **Claims 10-12** are rejected under 35 U.S.C. 103(a) as being unpatentable over the T414 Data Sheet in view of the May'948 Patent.

Regarding ***claim 10***, the T414 Data Sheet discloses the system discussed above in claim 9, and but does not expressly disclose of a second push down stack, said means for storing a top item being connected to provide an input to said second push down stack and a control means connected between said means for storing a top item and said second push down stack for controlling provision of the input to said second push down stack.

Claims 10-12 depend from claim 9 and therefore, include all of the limitations of claim 9. For at least the reasons set forth above with respect to the rejections of claims 1 and 9 under 35 U.S.C. §102(b), neither the T414 Data Sheet nor May disclose all of the elements of claim 9. Therefore, the combination of these references do not disclose all of the elements of any of claims 10-12 and cannot establish a prima facie case of obviousness with respect to any of claims 10-12.

Regarding claims 9-17, the office action states:

> 23. **Claims 9-17** are rejected under 35 U.S.C. 103(a) as being unpatentable over "The Motorola MC68020", written by Doug MacGregor *et al.*, IEEE Micro, August 1984, pages 101-118 (hereafter "MacGregor") in view of the reference "MC68020 32-Bit Microprocessor User's Manual", published by Motorola, having a copyright dated of 1984 (hereafter the "MC68020 User's Manual"), and further in view of U.S. Patent Number 4,985,848, issued to Pfeiffer *et al.* (hereafter "Pfeiffer").
>
> *an external memory* [see page 107, wherein "The enable bit controls the MC68020's use of cache...if it is clear, the cache is disabled and the processor always fetches from external memory."],

Note, the claim requires a DRAM and does not require the DRAM be external.

MacGregor does not provide multiplexed address and data on the same bus
MacGregor supplies the CPU instructions one at a time

Regarding **claim 9**, MacGregor has neither a "multiplexing means... configured to provide multiplexed row addresses, column addresses and data" on a bus connected to dynamic random access memory, nor does it have "means for fetching ... multiple sequential instructions from said dynamic random access memory in parallel and supply the multiple instructions to said central processing unit during a single memory cycle."

MacGregor supplies the CPU instructions one at a time

The latter claim element is essentially the same as the corresponding claim element in claim 1, which was discussed above in connection with the rejection of

claim 1 over MacGregor. The patent owners refer to and incorporate that discussion here. What MacGregor actually discloses is supplying the instructions to the execution pipeline, instruction by instruction, by storing the second half of the two-word instruction group fetched from memory in a "temporary register".

MacGregor does not provide multiplexed address and data on the same bus

With regard to the first of the above-referenced claim elements, MacGregor does not disclose, "multiplexing means... configured to provide multiplexed row addresses, column addresses and data" on a bus connected to dynamic random access memory. To the contrary, MacGregor actually states that its external addressing on its external bus is "non-multiplexed." See, e.g., MacGregor at page 107, which reads:

> **Hlgh-speed asynchronous bus**
> The MC68020 maintains the fast, easy-to-use, nonmultiplexed asynchronous bus interface of the M68000 family. However, several significant new features have been designed into the MC68020 bus interface. Figure 6 shows the MC68020's bus interface structure. The bus interface consists of full 32-bit address and data buses, sequence control lines, and miscellaneous lines for interrupt and arbitration control. The 32-bit address and data buses are nonmultiplexed for maximum performance and simple interface design. This and other improvements allow the interface to support a 180-11s bus cycle time at 16.67 MHz, with an address-valid-to-data-valid specification of 120 ns. A typical bus cycle is shown in Figure 7.

While MacGregor does have a "data multiplexer," as cited by the examiner, to support memories that have different data bus widths, as described at page 107 in connection with its discussion of "dynamic bus sizing," there is no discussion

anywhere in MacGregor that addresses are multiplexed. In fact, MacGregor states the exact opposite, that the busses are not multiplexed.

US patent number 4,985,848 (Pfeiffer *et al.*) was cited to show that the external memory can be DRAM, and was not cited to show that the MacGregor bus could be reconfigured as a multiplexed bus.

For the above reasons, the combination of MacGregor, the MC 68020 user's manual and Pfeiffer *et al.* fail to teach or suggest all of the claim limitations of claim 9 and do not, therefore, establish a prima facie case of obviousness with respect to claim 9. Claims 10-17 depend, either directly or indirectly, from claim 9 and are, therefore, patentable over the cited references for at least the same reasons as claim 9.

Patent owner respectfully submits that all claims under reexamination are confirmable as patentable.

If the Examiner believes a telephone conference would expedite prosecution of this application, the Examiner is invited to telephone Patent Owner's undersigned attorney at 269-279-8820.

Respectfully submitted,

Dated: January 19, 2010

/Larry E. Henneman, Jr./
Larry E. Henneman
Reg. No. 41,063

Phone: (269) 279-8820
Fax: (269) 279-8830

Henneman & Associates, PLC
70 N. Main Street
Three Rivers, MI 49093