A digital pipelined computer includes a memory (10) arranged to store high level language instructions and a plurality of microprogrammed digital computers (11, 12, 13, 14) coupled in parallel to the memory (10) and intercoupled by an interface (19) to form a pipeline for executing the high level language instructions. The interface (19) includes buffer register means (20-25) connected between selected pairs of the microprogrammed digital computers. When one of the microprogrammed digital computers attempts to load a full buffer register means or to read an empty buffer register means the computer initiating such attempt is temporarily halted.
FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT

<table>
<thead>
<tr>
<th>Code</th>
<th>Country</th>
<th>Code</th>
<th>Country</th>
</tr>
</thead>
<tbody>
<tr>
<td>AT</td>
<td>Austria</td>
<td>LU</td>
<td>Luxembourg</td>
</tr>
<tr>
<td>BR</td>
<td>Brazil</td>
<td>MC</td>
<td>Monaco</td>
</tr>
<tr>
<td>CF</td>
<td>Central African Republic</td>
<td>MG</td>
<td>Madagascar</td>
</tr>
<tr>
<td>CG</td>
<td>Congo</td>
<td>MW</td>
<td>Malawi</td>
</tr>
<tr>
<td>CH</td>
<td>Switzerland</td>
<td>NL</td>
<td>Netherlands</td>
</tr>
<tr>
<td>CM</td>
<td>Cameroon</td>
<td>RO</td>
<td>Romania</td>
</tr>
<tr>
<td>DE</td>
<td>Germany, Federal Republic of</td>
<td>SE</td>
<td>Sweden</td>
</tr>
<tr>
<td>DK</td>
<td>Denmark</td>
<td>SN</td>
<td>Senegal</td>
</tr>
<tr>
<td>FR</td>
<td>France</td>
<td>SU</td>
<td>Soviet Union</td>
</tr>
<tr>
<td>GA</td>
<td>Gabon</td>
<td>TD</td>
<td>Chad</td>
</tr>
<tr>
<td>GB</td>
<td>United Kingdom</td>
<td>TG</td>
<td>Togo</td>
</tr>
<tr>
<td>JP</td>
<td>Japan</td>
<td>US</td>
<td>United States of America</td>
</tr>
</tbody>
</table>
A Digital Pipelined Computer

Technical field

This invention relates to pipelined computers of the kind including memory means arranged to store programs of high level language instructions, a plurality of microprogrammed digital processor means each coupled to said memory means, and interface means intercoupling said processor means to form a pipeline and including at least one buffer register means coupled between first and second selected ones of said processor means.

It should be understood that a pipelined computer is one which has a plurality of processing stages in series forming a "pipeline". The processing stages are operative in turn to execute successive phases of an instruction and different phases of different instructions are executable concurrently by different ones of the processing stages. Such a computer can have a high operating speed.

Background Art

A general pipelined computer structure is known from British Patent Specification No. 1,506,972. The known computer includes a plurality of microprogrammed processing means forming stages in a pipeline. Shifting of information along the pipeline occurs when all the processor means have completed operation. This shifting arrangement is relatively slow. An alternative faster shifting arrangement includes a "handshaking" procedure wherein each processing unit contains a handshaking microprogram routine and communicates with adjacent processing units by means of "ready" and "accepted" signals. Under this arrangement, whenever a given stage completes its current operation it sends a "ready" signal to the next stage. It then waits for the next stage to indicate its acceptance of the trans-
ferred information by issuing an "accepted" signal, and then further waits for a "ready" signal from the preceding stage. When this is received, information is transferred into the buffer of the given stage from the preceding stage. The given stage transmits an "accepted" signal to the preceding stage and is now ready to start operation. This shifting arrangement has the disadvantage of being relatively slow because of the need to exchange "ready" and "accepted" signals before further operations can take place.

It is an object of the present invention to provide a pipelined computer of the kind specified wherein the aforementioned disadvantage is alleviated.

Disclosure of the invention

According to the present invention, there is provided a pipelined computer of the kind specified characterized in that said interface means includes synchronizing means coupled to said selected ones of said processor means and to said buffer register means and arranged to provide respective first and second control signals for stopping operation of one of said selected ones of said processor means which attempts to load or read said buffer register means when said buffer register means is full or empty respectively.

It will be appreciated that a pipelined computer according to the invention has a high speed of operation because of the provision of the first and second control signals when an attempt is made to load or read a full or empty buffer means respectively.

Loading of an empty buffer and unloading of a full buffer can take place without occupying waiting time.

Brief description of the drawings

One embodiment of the invention will now be described by way of example with reference to the accompanying drawings, in which:-
Figure 1 is a block diagram of a pipelined computer constructed according to the invention.

Figure 2 is a diagram illustrating a format of one high level language instruction set which is executed by the pipelined computer of Figure 1.

Figure 3 is a set of tables listing tasks which the various computers perform within the pipelined computer of Figure 1.

Figure 4 is a timing diagram illustrating the overlap operation of the pipeline of the pipelined computer of Figure 1.

Figure 5 is a block diagram of a hardwired three stage pipeline computer which comprises one of the microprogrammable stages of the pipelined computer of Figure 1.

Figure 6 is a timing diagram illustrating the operation of the hardwired pipeline of the computer of Figure 5 in conjunction with the microprogrammed stages within the pipelined computer of Figure 1.

Figure 7 is a detailed logic diagram of one preferred embodiment of the interlocking interface in the pipelined computer of Figure 1.

Best mode for carrying out the invention

A preferred embodiment of a pipelined computer constructed according to the invention will now be described in conjunction with Figure 1. This embodiment includes a digital memory 10 for storing therein programs comprised of high level language instructions. Also included are a plurality of microprogrammed digital computers 11-14 having respective control store memories 11a-14a for their microprograms. Computers 11-14 also respectively have interfaces 15-18 to memory 10. An interlocking interface 19 intercouples computers 11-14 to form a pipeline for executing the high level language instructions in memory 10. Each of the computers 11-14 is a separately microprogrammable stage of this pipeline.
Basically, this pipeline operates as follows. Computer 11 executes its microprogram in control store 11a, which operates to fetch the high level language instructions from memory 10. These fetched instructions also are aligned in a predetermined format and loaded into registers 20 in interface 19 under microprogram control. Computer 12 executes its microprogram in control store 12a, which operates to receive the aligned instructions from register 20 and calculate the memory addresses of the operands for these instructions. The calculated addresses are also loaded into register 21 under microcode control.

Computer 13 executes its microcode in control store 13a to receive the addresses from register 21, and fetches from memory 10 a portion of the operands which are indicated by the addresses. These operands and corresponding addresses are loaded into a register 22 within interface 19. Computer 14 executes microcode in control store 14a to receive the operand and addresses in register 22, and performs operations on the operand as specified by the high level language instruction. Computer 14 also fetches from memory 10 any remaining portions of the operand and performs thereon the operation specified by the high level language instruction.

Also included in the illustrated preferred embodiment are registers 23, 24 and 25. Register 23 is included within interface 19 to provide a means for computer 12 to transfer the calculated operand addresses of the high level language instructions directly to computer 14. In operation, computer 12 sends the results of its processing of the high level language either to computer 13 or computer 14 depending on the particular high level instruction that is being processed. Register 24 and 25 provide a means whereby computers 14 and 12 respectively can indicate the results of their operation to computers lying upstream in the pipeline. These results are indicated by various flags and branch control
information for example. The precise tasks performed by computers 11-14 and the information which they transfer via registers 20-25 will be explained in greater detail in conjunction with Figure 3.

In one preferred version of the Figure 1 embodiment, computers 11-14 each consist of the same identical hardware. That is, computers 11-14 differ only in their respective microprograms. Accordingly, this version has an extremely regular architecture. It also increases the volume of the single computer that is used for each of the stages by a factor of four over the volume of the pipelined computer. This increased volume translates to a reduced cost for the pipelined computer. Also, the regularized structure decreases the complexity of the pipelined computer, which makes for ease of design and maintenance.

Also preferably, each of the computers 11-14 execute their respective microprograms in their own internal pipeline. That is, computers 11-14 preferably each include a plurality of hardwired execution stages which are interconnected in a pipeline. The pipeline within computer 11 executes the microprograms in control store 11a of computer 11; the pipeline in computer 12 executes the microprograms in control store 12a of computer 12; etc. Any number of stages may comprise the respective pipelines within computers 11-14. For example, computers 11 and 14 may each include a three stage pipeline while computers 12 and 13 may each include a two stage pipeline. At the same time of course, computers 11-14 each form one stage of the computerized computer having higher level language instructions in memory 10.

In addition, control stores 11a-14a preferably are each writable control stores. That is, modifiable microprograms can be stored therein. This provides a means whereby one set of microprograms can be stored in control stores 11a-14a during one time interval to execute one set of high level language instructions; while
a second set of microprograms can be stored in control stores 11a-14a during a second time interval to execute a corresponding second set of high level language instructions.

A format for one set of high level language instructions, and their associated operands, which is suitable for being processed by the disclosed pipelined computer is illustrated in Figure 2. These instructions are the machine language of a Century 100 processor as described in the publication entitled "615-100 Functional Specification" from NCR Corporation. Basically, these high level language instructions have a double stage format 30 and a single stage format 31. Double stage commands 30 include eight bytes of information which are indicated in Figure 2 by the squares labeled OP through B1. Single stage command 31 includes four bytes of information as indicated in Figure 2 by the squares labelled OP through A1.

In formats 30 and 31, the symbol OP represents one of the following operation codes: Pack, Unpack, Add Unsigned, Subtract Unsigned, Add Binary, Subtract Binary, Branch Overflow, Branch Less, Branch On Equal, Branch Less Or Equal, Branch Greater, Branch Less Or Greater, Branch Greater Or Equal, Branch Unconditionally, Move A Right To Left, Compare Binary, Repeat, and Wait. All of these instructions pass through the stages of the pipeline formed by computers 11 and 12. The Add Binary, Subtract Binary, Move A Right To Left, and Compare Binary instructions also pass through computers 13 and 14.

In comparison, the Pack, Unpack, Add Unsigned, Subtract Unsigned instructions pass through computer 14 but bypass computer 13.

Each of the above described operation codes can be incorporated into the double stage format 30 or the single stage format 31. Single stage format commands have a B address equal to the B address which results from execution of the preceding command. For
example, the B address which results from execution of a Binary Compare command is the same as its initial B
address. Accordingly, one task which computer 12 per-
forms is to save fields T-B1 of a double stage command
and use them to form the B address for subsequent single
stage commands.

Each of the above described operation codes may
also be aligned in memory 10 on a double word boundary,
or a word boundary. This is indicated at 32 in Figure
2. Computer 11 has the task of fetching the commands
from memory 10, and shifting them so that they are all
aligned in register 20 in standard format regardless of
how they are aligned in memory 10.

The operands 33 and 34 on which instructions
30 and 31 operate are also illustrated in Figure 2. The
T field indicates the length of operands 33 and 34.
Operand 33 is addressed by the fields RA, A2, and A1;
while operand 34 is addressed by the fields RB, B2, and
B1. RA and RB indicate index registers. Basically,
operand 33 is addressed by the contents of index regis-
ter RA plus the quantity A2A1 as indicated at 35, while
operand 34 is addressed by the contents of index regis-
ter RB plus B2B1 as indicated at 36. Various modifica-
tions to this addressing are described in the above
cited publication on the Century 100.

Figure 3 illustrates how the multiple tasks
which constitute execution of the high level language
instructions of Figure 2 are partitioned among computers
11-14. As indicated in Table 1, computer 11 pre-fetches
up to four sequential high level language instructions
from memory 10. If one of these is a branch instruc-
tion, computer 11 pre-fetches both alternative instruc-
tions that may be executed depending on whether or not
the branch is taken. These pre-fetched instructions are
aligned into a single predetermined format, and loaded
sequentially into register 20. Computer 11 also moni-
tors the state of register 25 to determine if a branch
is or is not taken. In response thereto, computer 11 loads the correct alternative instruction into register 20.

Computer 12 receives the instructions which are loaded into register 20. Subsequently, it computes the addresses of the operands for these instructions. That is, it computes the contents of index register RA plus A2A1; and computes the contents of index register B plus B2B1. If register 20 holds a single stage command, then computer 12 calculates the address that results from execution of the preceding command and tacks that on as the B address for the single stage command.

In addition, computer 12 directly executes all branch instructions. It does this by testing the flags in register 24 in accordance with the operation code of the particular branch instruction, and by setting the state of register 25 to indicate that the branch condition either was or was not met. For OP codes which are not a branch, computer 12 passes the calculated operand addresses into either register 21 or 23 in accordance with the particular OP code as previously described in conjunction with Figure 2.

The function performed by computer 13 during the processing of the Add Binary, Subtract Binary, Move A Right To Left, and Compare Binary instructions is listed in Table 3. First, computer 13 receives the information loaded by computer 12 into the register 21. Subsequently, computer 13 pre-fetches one word of the A operand. If the bytes therein are not aligned on a word boundary, computer 13 performs no shifting of the alignment. Computer 13 also pre-fetches either one or two words of the B operand as is required to have at least as many bytes of the B operand pre-fetched as there are bytes of the A operand pre-fetched. The pre-fetched operands, and the A and B addresses, are then loaded into register 22 along with several pointers indicating the byte alignment of the A and B operands within the pre-fetched words.
As per TABLE 4, computer 14 receives the addresses and operands from register 22 or 23 depending upon the particular high level language instruction being performed. Any remaining portion of the operands are fetched by computer 14. The operands are also operated on by computer 14 in accordance with the operation code. For some instructions, computer 14 sets flags in register 24 to indicate the results of the operation. These flags are utilized by computer 12 during branch commands as described above.

A timing diagram illustrating how computers 11-14 perform each of their assigned tasks in a pipeline fashion is contained in Figure 4. This figure illustrates the execution of the commands Compare Binary, Move A Right To Left, Unpack, and Branch On Equal as an example. Specifically, the sequence 40a, 40b, and 40c illustrates how computers 11-14 process the Compare Binary instruction. Similarly, the sequence 41a, 41b, and 41c illustrates how computers 11-14 process the Move A Right To Left instruction. Both of these instructions of the high level language utilize all four stages of the pipeline.

In comparison, the sequence 42a and 42b illustrates how the Unpack command is performed by computers 11-14. As illustrated, only three stages of the pipeline are required to perform this command. Also in comparison, the sequence 43a and 43b illustrates how the Branch On Equal command is performed by computers 11-14. As illustrated, this command utilizes only computers 11 and 12. Computer 11 fetches and aligns the Branch On Equal command, determines that a Branch command has been fetched, and in response thereto fetches two alternate commands. One of the alternate commands will be performed if the Branch condition fails, while the other alternative commands will be performed if the Branch condition is met. Computer 12 tests the flag in register 24 and selects one of the alternate commands based
on the result of the test. This is indicated at 44a. After one of the alternate commands has been selected, it proceeds through the pipeline in the normal fashion as indicated at 44b and 44c.

Due to the pipelined interconnection of computers 11-14, the execution of the high level language instruction is overlapped in time. This overlapping is clearly illustrated in Figure 4. For example, at time instant 45, computers 11-14 respectively are performing portions of the Branch On Equal instruction, the Unpack instruction, the Move A Right To Left instructions, and the Compare Binary instruction. At the same time, each of the computers 11-14 preferably have their own internal hardwired pipeline which is operating to perform their respective microprograms. This point is described in greater detail in conjunction with Figures 5 and 6.

When one stage of the pipeline formed by computers 11-14 needs data from the preceding stage and that data is not available, the stage which needs the data enters into a freeze state until the data is available. This is indicated at 46 and 47 in Figure 4. Specifically, the freeze state at 46 illustrates the condition where computer 13 has finished processing its portion of the Move A Right To Left command and is waiting for the next command for which it has tasks to perform. In this illustrated example, this command is the second alternative of the branch command.

Similarly, the freeze state at 47 illustrates the case where computer 11 waits for computer 12 to determine which alternative of a branch condition is to be executed. Subsequent to that determination, computer 11 continues in its normal fashion to pre-fetch up to four sequential commands along the branch path which was taken. One preferred embodiment of interface 19 which implements this freeze function is illustrated and described in conjunction with Figure 7.

Referring now to Figure 5, there is illustrated a block diagram of a computer which suitably may
be used as each of the computers 11-14. That is, four of the computers of Figure 5 are included within the computerized computer of Figure 1. The computer of Figure 5 has its own three stage pipeline. These are termed the Fetch stage, the Interpret stage, and the Execute stage. Each of these stages are implemented by hardwired logic. This logic is defined in great detail in the publication "56-101 Processor Sub-system Functional Specification" from NCR Corporation.

Basically, the Fetch stage includes a control store address register and branch control logic 50, a control store memory 51, and control store data registers 52a-52e. These registers are partitioned to receive various portions of the microcommands which are read from control store memory 51. For example, register 52a receives and decodes a portion of the microcommands which addresses a register storage unit 60 within the InterPre stage.

The Interpret stage of the pipeline also includes a multiplexer 61, and buffer registers 62a-62e. Registers 62a and 62b form inputs to byte and bit shifting logic 70 which lies within the Execute stage. Registers 62c and 62d form inputs to an arithmetic logic unit 71 within the Execute stage. And register 62e forms an input to jump test logic and modify logic 72 within the Execute stage. The Execute stage also includes a buffer register 73 for the register storage unit 60. It further includes memory interface registers 74. Registers 74 couple via leads 15 to memory 10 as was previously described in conjunction with Figure 1.

A bus 75 interconnects units 71-74 and registers 20-25 of Figure 1. For example, bus 75 of computer 12 couples to registers 20, 21, 23, 24, and 25. Similarly, bus 75 of computer 13 couples to registers 21 and 22. Additional details of the lines and signals within bus 75 are described in conjunction with Figure 7.

Referring now to Figure 6, there is illustrated the detailed timing of how the multiple pipelines
within the pipelined computer of Figure 1 interact with each other. Transition 41b in Figure 6 corresponds to transition 41b and the timing diagram of Figure 4.

Near this transition, computer 13 operates on the Compare Binary command and the Move A Right To Left command. Simultaneously, computer 12 operates on the Move A Right To Left command and the Unpack command.

In Figure 6, reference numerals 80-82 respectively refer to the microinstructions in the Fetch stage, the Interpret stage, and the Execute stage of computer 13. Similarly, reference numerals 83-85 respectively refer to the microcommands in the Fetch stage, the Interpret stage, and the Execute stage of computer 12. Also, the mnemonics M1, M2, etc. refer to the first, second etc. microcommand of the program, in the corresponding stage, for the Move A Right To Left command; the mnemonics U1, U2, etc. refer to the first, second, etc. microcommand of the programs for the Unpack command; and the mnemonics C8, C9 and C10 refer to the eighth, ninth and tenth microcommand of the programs for the Compare Binary command.

At the previously described time instant 45 for example, the Fetch stage, the Interpret stage, and the Execute stage of computer 13 respectively operate on the fourth, third, and second microcommands in control store 13a which performs a portion of the Move A Right To Left command. Simultaneously, the Fetch stage, the Interpret stage, and the Execute stage of computer 12 respectively operate on the sixth, fifth, and fourth microcommand in control store 12a which performs a portion of the Unpack command. Similarly, at a time instant 86, the Fetch stage of computer 13 operates on the second microcommand of the Move A Right To Left command; the Execute stage of computer 13 operates on the tenth microcommand of the Compare Binary instruction; and the Execute stage of computer 12 operates on the second microcommand of the Unpack instruction. Also at time
instants 45 and 86, the pipelines within computers 11 and 14 are operating in a similar overlapped fashion.

The details of the interlocking interface 19 which interconnect computer 11-14 will now be described in conjunction with Figure 7. This figure is a detailed logic diagram of buffer 20. The other buffer registers 21-25 are of similar structure - the only difference being the number of information bits which is contained in the particular buffer register.

These bits are held in a register 90. Register 90 is a triggerable D-type register having a plurality of inputs which are loaded from bus 75a in computer 11 and unloaded to bus 75a in computer 12. ADDR signals on bus 75b are utilized to load register 90 and to test register 90 to determine if it is full or empty. To this end, signals ADDR couple to a decoder 91. Decoder 91 operates to select a particular binary code and indicates the presence of that code by a logic level on lead 92. Lead 92 couples to a logical AND gate 93 which operates to set a flip-flop 94 when it is empty, and to load register 90 through an AND gate 95.

A control signal Xout on a lead within bus 75b is an additional enabling signal which is connected to AND gate 93. Signal Xout indicates when computer 11 is executing a microcommand which loads the data on bus 75a into a register as specified by the ADDR code on bus 75b.

An AND gate 96 generates a signal FREEZE on another lead within bus 75b. This FREEZE signal is the logical AND of the FULL signal from flip-flop 94, the address detect signal on lead 92, and the X out signal. It operates to halt computer 11 when it attempts to load a full register 90.

Similarly, an AND gate 97 provides a FREEZE signal for computer 12 which enables the pipeline in computer 12 to stop when it requests data from an empty register 90. This FREEZE signal is generated on a lead within bus 75b in computer 12. The inputs to gate 97
are generated by a decoder 98 which decodes ADDR signals on bus 75b in computer 12, the empty side of flip-flop 94, and an Xin signal on a lead within bus 75b in computer 12. The Xin signal indicates that computer 12 is requesting data to be put onto bus 75 by the register indicated by the ADDR signals. In comparison, when computer 12 requests data from a full register 90, the pipeline in computer 12 does not stop. Instead, it proceeds to take data from register 90. An AND gate 99 is provided to force flip-flop 94 to the EMPTY state after this data transfer occurs.

Timing signals are also provided for synchronizing all of the above described operations. These signals are called CLOCK and CLOCK'. An inverter 100 generates signal CLOCK' from signal CLOCK. Suitably, these signals are square waves with a 50 NS period. Also, a RESET signal and a NAND gate 101 are included to initialize flip-flop 94 to an EMPTY state whenever the entire system is reset.

This completes the disclosure of the embodiment of Figure 1. It should be pointed out however, that many modifications and changes may be made to the Figure 1 embodiment. For example, the pipelined computer of Figure 1 may be scaled up to include additional ones of the computers 11-14 within the interlocking interface 19, or it may be scaled down to include as few as two of the computers 11-14. Basically, as the number of computers is increased, the amount of overlap in the pipeline increases. Conversely, as the number of computers is decreased, the amount of overlap in the pipeline decreases. The tasks which each stage of the pipeline must perform may be assigned various ways as was described in conjunction with Figure 3.

As another alternative, the pipelined computer of Figure 1 may be expanded to include one or more hardware assist units. For example, a special hard-wired assist unit which does high speed multiples and
high speed divides may couple to computer 14. In operation, computer 14 would pass the operands to the hardware assist unit and receive the result therefrom. As another example, the hardware assist unit could couple directly between computers 12 and 13 to perform the extended set-up. In this configuration, only computers 11, 12 and 14 would be microprogrammed. This configuration would be attractive if the logic required to do the assigned tasks were relatively simple.

As still another variation to the pipelined computer of Figure 1, additional levels of buffering may be included in the interlocking interface 19. That is, multiple registers 90 may be interconnected in stacks between computers 11-14. In this embodiment, control logic similar to that illustrated in Figure 7 would generate a suitable FREEZE signal to indicate when the stack was full or empty. The stack could suitably be constructed out of commercially available stack chips, such as an 8X03 chip from Signetics for example.
Claims

1. A pipelined computer, including memory means arranged to store programs of high level language instructions, a plurality of microprogrammed digital processor means each coupled to said memory means, and interface means intercoupling said processor means to form a pipeline and including at least one buffer register means coupled between first and second selected ones of said processor means, characterized in that said interface means including synchronizing means coupled to said selected ones (11, 12) of said processor means and to said buffer register means (20) and arranged to provide respective first and second control signals for stopping operation of one of said selected ones (11, 12) of said processor means which attempts to load or read said buffer register means (20) when said buffer register means (20) is full or empty respectively.

2. A pipelined computer according to claim 1, characterized in that said buffer register means includes a buffer register (90) providing a single level of buffer storage and in that said synchronizing means includes a bistable circuit (94) arranged to be controlled to first state upon said buffer register (90) being loaded and to be controlled to a second state upon said buffer register (90) being read.

3. A pipelined computer according to claim 2, characterized in that said synchronizing means includes first gating means (96) coupled to said bistable circuit (94) and to said first selected one (11) of said processor means and adapted to provide said first control signal (FREEZE) in response to an output request signal (X OUT) from said first selected one (11) of said processor means when said bistable circuit (94) is
3. (concluded)
in said first state (FULL), and second gating means
10 (97) coupled to said bistable circuit (94) and to said
second selected one (12) of said processor means and
adapted to provide said second control signal (FREEZE)
in response to an input request signal (X IN) from said
second selected one (12) of said processor means when
15 said bistable circuit (94) is in said second state
(EMPTY).

4. A pipelined computer according to claim 1, charac-
terized in that said first and second selected ones
(11, 12) of said processor means are respectively up-
stream and downstream in said pipeline said first and
second selected ones (11, 12) of said processor means
being adapted to respectively load and read said buffer
register means (20), and in that there is provided fur-
ther buffer register means (e.g. 24) coupled between
third and fourth selected ones (14, 12) of said pro-
cessor means lying respectively downstream and upstream
in said pipeline, and adapted to respectively load and
read said further buffer register means (e.g. 24).

5. A pipelined computer according to claim 1, charac-
terized in that each microprogrammed processor means
of said plurality includes a plurality of hardwired
execution stages (FETCH, INTERPRET, EXECUTE, Fig. 5)
connected to form a hardwired pipeline for executing
the microprogram of that processor means.

6. A pipelined computer according to claim 1, charac-
terized in that each of said microprogrammed procес-
sor means includes the same identical hardware.

7. A pipelined computer according to claim 1, charac-
terized in that selected ones of said microprogrammed
processor means includes modifiable control store means.
8. A pipelined computer according to claim 1, characterized in that selected ones of said microprogrammed processor means have respective hardware assist means coupled thereto.

9. A pipelined computer according to claim 1, characterized in that said pipeline includes four microprogrammed digital computer means forming respective first (11), second (12), third (13) and fourth (14) stages of said pipeline, adapted respectively to fetch said high level language instructions from said memory means (10) and align them in a predetermined alignment; to calculate memory addresses of the operands of the aligned instructions; to pre-fetch at least selected portions of said operands; and to perform operations on said operands.

10. A pipelined computer according to claim 9, characterized in that the first stage (11) of said pipeline is further adapted to pre-fetch alternative instructions when the fetched high level language instruction is a branch instruction and in that said second stage (12) of said pipeline is adapted to directly execute said branch instruction and provide information concerning the result of the execution to said first stage (11) of said pipeline.
**FIG. 3**

**TABLE 1 - COMPUTER 11**

<table>
<thead>
<tr>
<th>FETCH UP TO FOUR SEQUENTIAL CMDS.</th>
<th>COMPUTE $[R_A] + A_2 A_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>FETCH ALT. CMD. IF BRANCH CMD.</td>
<td>COMPUTE $[R_B] + B_2 B_1$</td>
</tr>
<tr>
<td>ALIGN. CMDS IN REG. 20</td>
<td>HOLDS $[R_B] + B_2 B_1$ FOR SUBSEQUENT SINGLE STG. CMDS.</td>
</tr>
<tr>
<td>SWITCH TO ALT. CMD IF BRANCH OCCURS</td>
<td>EXECUTES ALL BRANCH CMDS DIRECTLY</td>
</tr>
</tbody>
</table>

**TABLE 2 - COMPUTER 12**

**TABLE 3 - COMPUTER 13**

<table>
<thead>
<tr>
<th>FETCH ONE WORD OF AA</th>
<th>RECEIVE ADDRESSES OF OPERANDS FROM REG. 22/23</th>
</tr>
</thead>
<tbody>
<tr>
<td>FETCH ONE OR TWO WORDS OF BB</td>
<td>FETCH AND EXECUTE REMAINING OPERANDS</td>
</tr>
<tr>
<td>GENERATE BYTE POINTERS</td>
<td>SET FLAGS IN REG. 24</td>
</tr>
</tbody>
</table>

**TABLE 4 - COMPUTER 14**
**INTERNATIONAL SEARCH REPORT**

**International Application No:** PCT/US79/00402

### I. CLASSIFICATION OF SUBJECT MATTER

If several classification symbols apply, indicate all.

- **US. CL.** 364/200
- **INT. CL.** G 06F 7/00

### II. FIELDS SEARCHED

**Minimum Documentation Searched**

<table>
<thead>
<tr>
<th>Classification System</th>
<th>Classification Symbols</th>
</tr>
</thead>
<tbody>
<tr>
<td>U.S.</td>
<td>364-200; 364-900</td>
</tr>
</tbody>
</table>

**Documentation Searched other than Minimum Documentation to the Extent that such Documents are Included in the Fields Searched**

### III. DOCUMENTS CONSIDERED TO BE RELEVANT

<table>
<thead>
<tr>
<th>Category</th>
<th>Citation of Document, with indication, where appropriate, of the relevant passages</th>
<th>Relevant to Claim No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>US, A, 4,112,489 Published 05 September 1978, Wood</td>
<td>1-10</td>
</tr>
<tr>
<td>X</td>
<td>US, A, 4,041,461 Published 09 August 1977, Kratz et al</td>
<td>1-10</td>
</tr>
<tr>
<td>X</td>
<td>US, A, 3,949,379 Published 06 April 1976, Ball</td>
<td>1-10</td>
</tr>
<tr>
<td>A</td>
<td>US, A, 3,614,742 Published 19 October 1971, Watson et al</td>
<td>1-10</td>
</tr>
<tr>
<td>A</td>
<td>US, A, 3,713,107 Published 23 January 1973, Barsamian</td>
<td>1-10</td>
</tr>
<tr>
<td>A</td>
<td>US, A, 4,079,456 Published 14 March 1978, Lunsford et al</td>
<td>1-10</td>
</tr>
</tbody>
</table>

* Special categories of cited documents:
  - "A" document defining the general state of the art
  - "E" earlier document but published on or after the international filing date
  - "L" document cited for special reason other than those referred to in the other categories
  - "O" document referring to an oral disclosure, use, exhibition or other means
  - "P" document published prior to the international filing date but on or after the priority date claimed
  - "T" later document published on or after the international filing date or priority date and not in conflict with the application, but cited to understand the principle or theory underlying the invention
  - "X" document of particular relevance

### IV. CERTIFICATION

- **Date of the Actual Completion of the International Search:** 03 August 1979
- **Date of Mailing of this International Search Report:** 22 AUG 1979

Form PCT/ISA/210 (second sheet) (October 1977)