A microcomputer comprises memory (13, 19) and CPU (12). Communication channels (50-53, 61, 63) are provided to permit data transmission from an outputting process (X) to an inputting process (Y) in response to input and output instructions. The channels are single word locations (50-53) in memory or registers (61, 63) in serial links (25) arranged to hold values which ensure that data transmission occurs when both processes are at corresponding stages in their program sequence. CPU 12 requires only one input or output instruction in a process program in order to effect data transmission.
**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>KR</td>
<td>Republic of Corea</td>
</tr>
<tr>
<td>AU</td>
<td>Australia</td>
<td>LI</td>
<td>Liechtenstein</td>
</tr>
<tr>
<td>BE</td>
<td>Belgium</td>
<td>LK</td>
<td>Sri Lanka</td>
</tr>
<tr>
<td>BG</td>
<td>Bulgaria</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>MR</td>
<td>Mauritania</td>
</tr>
<tr>
<td>CH</td>
<td>Switzerland</td>
<td>MW</td>
<td>Malawi</td>
</tr>
<tr>
<td>CM</td>
<td>Cameroon</td>
<td>NL</td>
<td>Netherlands</td>
</tr>
<tr>
<td>DE</td>
<td>Germany, Federal Republic of</td>
<td>NO</td>
<td>Norway</td>
</tr>
<tr>
<td>DK</td>
<td>Denmark</td>
<td>RO</td>
<td>Romania</td>
</tr>
<tr>
<td>FI</td>
<td>Finland</td>
<td>SD</td>
<td>Sudan</td>
</tr>
<tr>
<td>FR</td>
<td>France</td>
<td>SE</td>
<td>Sweden</td>
</tr>
<tr>
<td>GA</td>
<td>Gabon</td>
<td>SN</td>
<td>Senegal</td>
</tr>
<tr>
<td>GB</td>
<td>United Kingdom</td>
<td>SU</td>
<td>Soviet Union</td>
</tr>
<tr>
<td>HU</td>
<td>Hungary</td>
<td>TD</td>
<td>Chad</td>
</tr>
<tr>
<td>JP</td>
<td>Japan</td>
<td>TG</td>
<td>Togo</td>
</tr>
<tr>
<td>KP</td>
<td>Democratic People's Republic of Korea</td>
<td>US</td>
<td>United States of America</td>
</tr>
</tbody>
</table>
MICROCOMPUTER WITH INTERPROCESS COMMUNICATION


BACKGROUND TO THE INVENTION

The above mentioned applications describe an improved microcomputer comprising a single integrated circuit device having a processor and memory in the form of RAM on the same integrated circuit device, the processor being arranged to operate in accordance with functions selected from a function set. Said function set includes direct functions and indirect functions, the indirect functions being used to select one of a variety of "operations". Said operations include the operation "synchronise" which is used to permit two processes to communicate with each other, the two processes being either on the same microcomputer or in other cases the two processes may be on separate respective microcomputers. As can be seen from the above mentioned patent applications, use of the operation "synchronise" has required two word locations in memory to provide a channel permitting process to process communication on the same microcomputer. One word of the channel was used to indicate the state of the channel and the other word was used to hold data for communication through the channel. Furthermore, process to process communication requires that each process includes in its program sequence two operations of "synchronise" for each message transmission in order to ensure that the message transmission occurs when the two processes are at corresponding stages in their program sequences. In some circumstances this may cause a process to be descheduled twice in order to effect synchronised message transmission.
OBJECTS OF THE INVENTION

It is an object of the present invention to provide an improved microcomputer wherein data transmission between processes may be effected by use of instructions which may result in improved performance due to the use of fewer instructions during process to process communication. It is an further object to implement a communication channel using less memory space.

In a preferred embodiment, the "synchronise" operation referred to in the above mentioned patent applications can be replaced by "input" and "output" operations as described below.

SUMMARY OF THE PRESENT INVENTION

The present invention provides a microcomputer comprising memory and a processor arranged to execute a plurality of concurrent processes, each in accordance with a program consisting of a plurality of instructions for sequential execution by the processor, each instruction designating a required function to be executed by the processor, said processor comprising (1) a plurality of registers and data transfer means for use in data transfers to and from said registers (2) means for receiving each instruction and loading into one of the processor registers a value associated with the instruction, and (3) control means for controlling said data transfer means and registers in response to each instruction received to cause the processor to operate in accordance with the instruction, wherein the microcomputer includes:-(a) scheduling means to enable the processor to share its processing time between a plurality of concurrent processes, said scheduling means comprising:-

(i) means for identifying one or more processes which form a collection awaiting execution by the processor
(ii) means for descheduling a process by interrupting execution of the current process
(iii) means for scheduling a process by adding it to said collection, and
(b) communication means to permit data transmission from one process to another when both processes are at corresponding stages in their program sequences, an outputting process operating to output data in response to an output instruction in its program and an inputting process operating to input data in response to an input instruction in its program, said communication means including:

(i) a channel comprising store means for holding a value indicating whether a process has executed an instruction to effect data transmission using that channel,

(ii) means, responsive to execution of an input or output instruction by one of the processes involved in the data transmission when said one process is the current process for testing the contents of said channel and arranged to operate said means to deschedule the current process if the channel does not contain a value indicating that the other process involved in the data transmission has reached a corresponding program stage,

the communication means being arranged such that an outputting process requires only one output instruction in its program sequence and an inputting process requires only one input instruction in its program sequence, whereby either process is not descheduled more than once, in order to effect such data transmission.

It will be understood that the term microcomputer relates to small sized computers generally based on integrated circuit devices but it does not impose any limit on how small the computer may be.

Preferably the scheduling means includes means for indicating the current process which is being executed by the processor and the communication means includes means for loading into said channel an identification of the current process if that current process is descheduled as a result of said testing of the contents of the channel.

Preferably said identification is a pointer value identifying an address of a memory workspace of the current process.
Preferably said channel is one of a plurality of channels, each comprising an addressable store location. Preferably the microcomputer is arranged to permit data transmission between processes which are executed on the same microcomputer and for this purpose the or each channel comprises a memory location. Preferably the microcomputer is arranged to permit external data transmission between processes which are executed on different microcomputers and in this case the or each channel may comprise a first register forming part of an external communication link.

In a preferred embodiment, the microcomputer comprises an integrated circuit device including a plurality of channels some in the form of registers and some provided by memory locations in memory on the same integrated circuit device as the processor. Preferably the channel provided by memory locations each comprise a single word location.

**DESCRIPTION OF THE PREFERRED EMBODIMENT**

An embodiment of the invention will now be described by way of example and with reference to the accompanying drawings in which:

Figure 1 is a block diagram showing the main features of the microcomputer,

Figure 2 shows an arrangement of memory workspaces, registers, memory channels and serial links in the microcomputer,

Figure 3 illustrates a succession of states of two workspace locations and one memory channel during process to process communication on one microcomputer,

Figure 4 illustrates a network of two interconnected microcomputers permitting process to process communication from one microcomputer to the other, and

Figure 5 shows a sequence of states for process and data registers in the microcomputers shown in Figure 4 during process to process communication between the two microcomputers.
The microcomputer described in this example comprises an integrated circuit device in the form of a single silicon chip having both a processor and memory in the form of RAM as well as links to permit external communication. The main elements of the microcomputer are illustrated in Figure 1 on a single silicon chip 11 using p-well complementary MOS technology. A central processing unit (CPU) 12 is provided with some read-only memory (ROM) 13 and is coupled to a memory interface 14 controlled by interface control logic 15. The CPU 12 incorporates an arithmetic logic unit (ALU), registers and data paths some of which are illustrated in Figure 2. The CPU 12 and memory interface 14 are connected to a bus 16 which provides interconnection between the elements on the chip 11. A service system 17 is provided with a plurality of input pins 18. The microcomputer is provided with a random access memory (RAM) 19 and ROM 20 and the amount of memory on the chip is not less than 1K byte so that the processor 12 can be operated without external memory. Preferably the memory on the chip is at least 4K bytes. An external memory interface 23 is provided and connected to a plurality of pins 24 for connection to an optional external memory. To allow the microcomputer to be linked to other microcomputers to form a network, a plurality of serial links 25 are provided having input and output pins. The input and output pins of one serial link may each be connected by a single wire, non-shared unidirectional connection, to the corresponding output and input pins of a serial link on another microcomputer. Each serial link is connected to a synchronisation logic unit 10 comprising process scheduling logic.

The block diagram shown in Figure 1 corresponds to that included in the above mentioned European Patent Application No 83307078.2, Japanese Patent Application No 221455/1983 and US Patent Applications Nos 552601, 552602, 553027, 553028 and 553029. To avoid unnecessary repetition of description, the full details of the construction and operation of that microcomputer will not be set out below but the description in the above mentioned patent applications is hereby
incorporated herein by reference.

As described in the above mentioned patent applications, the CPU 12 includes a plurality of registers the following of which are shown in Figure 2:-

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Register</th>
</tr>
</thead>
<tbody>
<tr>
<td>IB</td>
<td>Instruction buffer 30 for receiving sequentially from memory instructions from a current process program</td>
</tr>
<tr>
<td>OREGTR</td>
<td>An operand register 31 for receiving data derived from an instruction in the instruction buffer 30</td>
</tr>
<tr>
<td>WPTR REG</td>
<td>A register 32 for holding a workspace pointer (WPTR) of the current process</td>
</tr>
<tr>
<td>IPTR REG</td>
<td>A register 33 which holds an instruction pointer (IPTR) indicating the program stage of the current process</td>
</tr>
<tr>
<td>LPTR REG</td>
<td>A register 34 for holding a pointer to the workspace of the last process on the list of processes waiting to be executed</td>
</tr>
<tr>
<td>AREGTR</td>
<td>A first (A) register 35 for holding an operand for the ALU</td>
</tr>
<tr>
<td>BREGTR</td>
<td>A second (B) register 36 arranged as a stack with the AREGTR for holding operands for the ALU</td>
</tr>
</tbody>
</table>

The CPU also includes the arithmetic logic unit 37 and the contents of the CPU are interconnected to permit data transfer to and from said registers under the control of a microprogram held in a microinstruction ROM 13 (see Figure 4), which is arranged to respond to "input" and "output" operations defined below.
Figure 2 also illustrates a workspace region 40 provided in the memory 19 for a process X as well as a workspace region 41 for a process Y. Each workspace comprises a plurality of addressable locations as indicated arranged to hold a plurality of variables associated with the process as well as the value IPTR to indicate the program stage for that process when it is next rescheduled together with a pointer to the WPTR of the next process in a collection of processes awaiting execution by the processor. Figure 2 also indicates four memory channels 50, 51, 52 and 53 each comprising a single word addressable location in the memory 19. It will be understood that the number of channels is not limited to four and any number of channels may be provided as required. Figure 2 also illustrates the provision of serial link registers which are each addressable in the same way as the memory channels or other memory locations. Each serial link 25 includes an input data register 60, an input process register 61, an output data register 62 and an output process register 63. Figure 4 illustrates a network comprising two interconnected microcomputers 11a and 11b and for simplicity only one serial link is shown in block form on each of these microcomputers. The output process registers of each link form an output channel having associated with it an output data register as well as output control logic 70 which includes an output control state machine 71 and an output data state machine 72. The input process register forms an input channel having associated with it an input data register as well as input control logic 73 including an input control state machine 74 and an input data state machine 75. The control logic 70 and 73 as well as the control and data state machines are constructed and operated as described in the above mentioned patent applications with the exception of the succession of states of the control state machines and this will be described below.

As can be seen from Figure 4, the output data register 62 is connected to an output pin 80 which is connected by a single non-shared unidirectional wire 81 to an input pin 82 connected to the input data register 66. Similarly the output data register 64 is connected via an output pin 83 through a single non-shared unidirectional wire 84 to an input pin 85 connected to the input data register 60.
The program sequence for each process includes a succession of instructions incorporating functions and operations as described in the above mentioned patent applications.

In order to effect message transmission between processes either on the same microcomputer or between processes on different microcomputers, the above mentioned patent applications use the operation called "synchronise" which has code number 11 in the list of operations.

The present example provides an improved microcomputer which has improved performance due to the use of fewer operations during process to process communication and furthermore less memory space is required to implement a communication channel as each of the above mentioned channels 50 to 53 involves a single word location only rather than the two word locations for the channels 40 to 43 of the above mentioned patent applications. These changes are effected in the present example by arranging for the microcomputer to respond to additional operations entitled "input" and "output" which may be added to the list of operations given in the above mentioned patent applications and these additional operations may have code numbers 16 and 17. Using the OCCAM language notation which is set forth in the booklet entitled Programming Manual - OCCAM published and distributed by Inmos Limited in 1983 in the United Kingdom as well as "Process-Orientated Language Meets Demands of Distributed Processing", Electronics (November 31, 1982) both of which are hereby incorporated herein by reference, the operations "output" and "input" as well as the procedures "wait ( )" and "run (OREG)" are defined as follows:
output
   Definition:
   1  SEQ
   2  OREG := memory [ AREG ]
   3  IF
   4  OREG = READY
   5    SEQ
   6    memory [ AREG + 1 ] := BREG
   7    memory [ AREG ] := WPTR
   8    wait ( )
   9  OREG = NIL
  10  SEQ
  11    memory [ WPTR ] := BREG
  12    memory [ AREG ] := WPTR
  13    wait ( )
  14  TRUE
  15    SEQ
  16    memory [ OREG ] := BREG
  17    memory [ AREG ] := NIL
  18  run ( OREG )

input
   Definition:
   1  SEQ
   2  OREG := memory [ AREG ]
   3  IF
   4  OREG = READY
   5    SEQ
   6    memory [ WPTR ] := memory [ AREG + 1 ]
   7    memory [ AREG ] := NIL
   8  OREG = NIL
  9  SEQ
 10    memory [ AREG ] := WPTR
 11    wait ( )
 12  TRUE
 13    SEQ
 14    memory [ WPTR ] := memory [ OREG ]
 15    memory [ AREG ] := NIL
 16  run ( OREG )
The procedure "wait ()" is defined as follows:

PROC wait ( )
1   SEQ
2       memory [ WPTR - 1 ] := IPTR
3   for each external input request from a link
4       SEQ
5       OREG := link [ process ]
6       link [ process ] := NIL
7       memory [ OREG ] := link [ data ]
8       run ( OREG )
9   for each external output request from a link
10      SEQ
11      OREG := link [ process ]
12      link [ process ] := READY
13      run ( OREG )
14      WPTR := memory [ WPTR - 2 ]
15      IPTR := memory [ WPTR - 1 ]

The procedure "run ( OREG )" is defined as follows:

PROC run ( OREG )
1   IF
2      OREG <> READY
3       SEQ
4       memory [ LPTR - 2 ] := OREG
5       LPTR := OREG
6      OREG = READY
7      SKIP
As mentioned above, the succession of states of the output control state machine 71 and the input control state machine 74 need modification from those described in the above mentioned patent applications in accordance with the following tables of successive states:

### OUTPUT CONTROL STATE MACHINE 71

<table>
<thead>
<tr>
<th>State</th>
<th>Inputs</th>
<th>Outputs</th>
<th>Next State</th>
</tr>
</thead>
<tbody>
<tr>
<td>any</td>
<td>Reset</td>
<td>SetPregready</td>
<td>waitdata</td>
</tr>
<tr>
<td>waitdata</td>
<td>Pregready</td>
<td>waitdata</td>
<td></td>
</tr>
<tr>
<td>waitdata</td>
<td>Pregwptr</td>
<td>send1</td>
<td></td>
</tr>
<tr>
<td>send1</td>
<td>ΔDatagone</td>
<td>Datagone</td>
<td>send1</td>
</tr>
<tr>
<td>send1</td>
<td>Datagone</td>
<td>send2</td>
<td></td>
</tr>
<tr>
<td>send2</td>
<td>Datagone</td>
<td>send2</td>
<td></td>
</tr>
<tr>
<td>send2</td>
<td>ΔDatagone</td>
<td>waitack1</td>
<td></td>
</tr>
<tr>
<td>waitack1</td>
<td>ΔAckready</td>
<td>waitack1</td>
<td></td>
</tr>
<tr>
<td>waitack1</td>
<td>Ackready</td>
<td>waitack2</td>
<td></td>
</tr>
<tr>
<td>waitack2</td>
<td>Ackready</td>
<td>Acktaken</td>
<td>waitack2</td>
</tr>
<tr>
<td>waitack2</td>
<td>ΔAckready</td>
<td>Setrequest</td>
<td>waitsched</td>
</tr>
<tr>
<td>waitsched</td>
<td>Pregwptr</td>
<td>waitsched</td>
<td></td>
</tr>
<tr>
<td>waitsched</td>
<td>Pregready</td>
<td>waitdata</td>
<td></td>
</tr>
</tbody>
</table>

### INPUT CONTROL STATE MACHINE 74

<table>
<thead>
<tr>
<th>State</th>
<th>Inputs</th>
<th>Outputs</th>
<th>Next State</th>
</tr>
</thead>
<tbody>
<tr>
<td>any</td>
<td>Reset</td>
<td>SetPregnil</td>
<td>receive1</td>
</tr>
<tr>
<td>receive1</td>
<td>ΔDataready</td>
<td>receive1</td>
<td></td>
</tr>
<tr>
<td>receive1</td>
<td>Dataready</td>
<td>test</td>
<td>test</td>
</tr>
<tr>
<td>test</td>
<td>Mbusy</td>
<td>test</td>
<td>test</td>
</tr>
<tr>
<td>test</td>
<td>(ΔMbusy)| Pregnil</td>
<td>SetPregready</td>
<td>waitproc</td>
</tr>
<tr>
<td>test</td>
<td>(ΔMbusy)| Pregwptr</td>
<td>Setrequest</td>
<td>waitproc</td>
</tr>
<tr>
<td>waitproc</td>
<td>ΔPregnil</td>
<td>waitproc</td>
<td></td>
</tr>
<tr>
<td>waitproc</td>
<td>Pregnil</td>
<td>receive2</td>
<td></td>
</tr>
<tr>
<td>receive2</td>
<td>Dataready</td>
<td>Datataken</td>
<td>receive2</td>
</tr>
<tr>
<td>receive2</td>
<td>ΔDataready</td>
<td>acksend1</td>
<td></td>
</tr>
<tr>
<td>acksend1</td>
<td>ΔAckgone</td>
<td>Ackgo</td>
<td>acksend1</td>
</tr>
<tr>
<td>acksend1</td>
<td>Ackgone</td>
<td>acksend2</td>
<td></td>
</tr>
<tr>
<td>acksend2</td>
<td>Ackgone</td>
<td>acksend2</td>
<td></td>
</tr>
<tr>
<td>acksend2</td>
<td>ΔAckgone</td>
<td>receive1</td>
<td></td>
</tr>
</tbody>
</table>
The way in which processes operate with input or output instructions will now be further described. The instructions are equally applicable to process to process communication on the same microcomputer or different microcomputers. In either case the process identifies a channel for use in the message transmission and that channel has an address either of a memory channel (such as 50 to 53) or a serial link provided by a process register (such as 61 or 63). In the above definitions of input and output as well as the procedure's wait and run, line numbers have been added to each stage of the definition for ease of reference although the line numbers form no part of the definition. In both input and output instructions the first line indicates that a sequence of events is to occur the first of which is defined in line number 2. The address of the channel to be used in the communication is loaded into AREGTR 35 so that the expression memory [ AREG ] identifies the channel. The contents of that channel are loaded into the OREGTR 31 so that the value in the channel is tested by the microcomputer to determine the state of the channel. In both the input and output instructions line 3 indicates various alternatives that may follow depending on the result of testing the contents of the OREGTR. The contents of the OREGTR may indicate that the channel already holds a first special value NIL, or a second special value READY or a third special value representing the workspace pointer of a descheduled process which has already executed an instruction to effect data transmission using that channel. In the case of an output instruction, if a test of the channel finds the special value NIL, the processor then carries out the sequence indicated in lines 11, 12 and 13 of the output instruction. If the channel contains the second special value READY, then the processor carries out the sequence specified in the three lines 6, 7 and 8 of the output definition. If on the other hand the channel is found to hold the workspace pointer of a descheduled process, this represents the condition TRUE in line 14 so that the processor then executes the sequence of three lines set out as 16, 17 and 18 of the output definition. Similarly, on execution of an input instruction, if testing of the OREGTR indicates that the channel has the special value READY, the processor follows the sequence of lines 5 and 7 of the input definition. If the channel has the special value NIL
then the processor executes the sequence of lines 10 and 11 of the input definition. If the channel has the workspace pointer of a descheduled process then the processor carries out the sequence defined in lines 14 to 16 of the input definition.

Communication between processes on the same microcomputer
In this case, the channel to be used in the message communication will be a single memory location having a plurality of bit positions with a single address such as a single word location. As will be explained below, such a channel can only contain either the first special value NIL or the WPTR of a descheduled process. This will now be described for message transmission between a process X wishing to communicate via channel 50 with a process Y on the same microcomputer. Using the same memory location addresses as Figure 2, process X has a workspace location 90 for a variable and process Y has a workspace location 91 for a variable. The channel 50 has the same address as indicated in Figure 2. The channel 50 is a single word in memory providing a unidirectional communication channel which is shared by two and only two processes at any one time. If process X is to execute an output instruction, it first identifies the address of the channel 50 by loading the address of channel 50 into the AREGTR and the data to be transmitted is loaded into the BREGTR. On execution of the output instruction by process X, the contents of channel 50 are tested and in accordance with Figure 3 found to represent the first special value NIL. Figure 3 illustrates a sequence of stages each carrying the subscript a, b and c and the condition NIL is represented at 50a. As the channel does not have the workspace pointer of process Y waiting to receive the data, lines 11 to 13 of the output definition are followed so that process X causes the data to be transferred from the BREGTR into location 90 of its own workspace (as indicated at 90b) and its workspace pointer is loaded into the channel as indicated at 50b. Process X then executes a "wait" procedure which deschedules process X. Process X now waits until process Y is ready to input the data. When process Y approaches the corresponding stage of its program, it identifies the channel 50 by loading the address of channel 50 into the AREGTR and carries out an input instruction which on executing lines
2 and 3 of the input definition, locates the workspace pointer of process X in channel 50. This meets the condition TRUE in line 12 of the input definition so that process Y then carries out lines 14 to 16 of the input definition. This causes the data stored in the workspace location 90 of process X to be copied into location 91 of the workspace of process Y as indicated at 91c. It also loads the first special value NIL into channel 50 as indicated at 50c. It then causes a "run" procedure for process X which adds process X to the end of the list awaiting process operation. This is the position shown at the end of the sequence in Figure 3 with process Y having continued without interruption and process X waiting on a list.

As can be seen from the definition of the procedure "wait" line 2 of the definition has the effect of storing an indication of the present program stage of the process at a workspace location having an offset of one from the workspace pointer for the process, as shown at 92 in Figure 2. Lines 3 to 13 of the definition of "wait" refer to external communications. Line 14 of the definition of "wait" causes the WPTR REG 32 to be loaded with a pointer to the next process on the list awaiting execution (taken from location 93, which has an offset of 2 from the WPTR location as shown in Figure 2) and line 15 loads the register 33 with the program stage indication of that process as soon as it becomes the current process. As can be seen from the definition of "run" line 2 sets out the possibility of two alternative situations. The first of these is the condition in line 2 in which the contents of the OREGTR are not READY in which case the sequence of lines 4 and 5 is followed or alternatively the OREGTR does have the value READY as indicated in line 6 in which case no action is taken. This occurs in external communications which are to be described below. The effect of line 4 of the definition of run is to add the process which is to be scheduled to the end of the list by having its workspace pointer stored in the workspace of the process currently indicated as the last on the list and this is done at an offset of two from the WPTR of the process as indicated at address 10998 in Figure 2.
According to line 5 of the definition of run, the WPTR of the process being scheduled is entered into the LPTR REG 34 indicating that it is now the last on the list.

The second special value "READY" is only applicable in communication between processes on different microcomputers and will be described below.

Communication between processes on different microcomputers
This will be described with reference to the sequence illustrated in Figure 5 representing an outputting process X on microcomputer 11a transmitting a message via line 81 to an inputting process Y on microcomputer 11b. Successive stages of the registers are marked with subscripts a to d in Figure 5. External communication is effected in generally similar manner using identical input and output operations although the control logical for the serial links may set the process registers to a second special value READY and they may also provide input or output requests on lines 94 and 95 to the CPU 12 of each microcomputer in order to schedule any descheduled process. As can be seen from the tables showing the outputs from the control state machines, each output process register is reset to READY whereas each input process register is normally reset to NIL. This is the condition shown for the register 63a and register 67a in Figure 5. For the process X to output data, it first loads the address of the channel, which in this case is the output process register 63, into the AREGTR and loads the data to be output into the BREGTR. On executing an output instruction the contents of the channel are loaded into the OREGTR and tested and as indicated at 63a in Figure 5, this detects the second special value READY due to the reset condition caused by the output control state machine 71. The data register 62a is marked EMPTY indicating that the relevant data has not yet been loaded into the register. It will be understood that the word EMPTY marked on a data register means that the contents of the register have no significance. In accordance with the definition of the output instruction, lines 6, 7 and 8 of that definition cause the data to be transferred from the BREGTR into the output data register 62, as this has an address which
is an offset of one from the process register 63, and stores the
workspace pointer of process X in the channel 63b and executes a "wait"
procedure which deschedules process X. This is the position shown at
63b and 62b in Figure 5. The output control logic 70 due to changes in
state of the state machines causes output of the data along line 81 to
the input data register 66 of microcomputer 11b. For the purposes of
Figure 5, it will be assumed that process Y has already executed an
input instruction prior to receipt of data in the data register 66. At
the time the process Y executed the input instruction, the process
register 67 will have held the special value NIL as this was the reset
condition caused by operation of the control state machine. Process Y
will have tested the value in the channel and on finding NIL will have
carried out lines 9 and 10 of the input definition. In other words,
the workspace pointer of process Y will have been loaded into the
process register 67 and process Y will have been descheduled by the
wait procedure. This is the position shown at 67b and 66b in Figure 5
and is assumed to be the position when data is received via line 81
into the input data register 66. The input control logic 73 on
microcomputer 11b then changes state on receipt of the data to generate
an input request to the CPU 12 on line 94 of microcomputer 11b. In
order to understand the effect of this, it is necessary to consider
lines 3 to 7 of the definition of the procedure "wait". Whenever a
process being executed by microcomputer 11b is descheduled by a wait
procedure, the CPU of microcomputer 11b looks for any external input
request from a serial link as required by line 3 of the definition of
procedure "wait". If there are no input requests it moves to line 9 of
the definition where it looks for any external output request. In the
definition of "wait" link[process] indicates the contents of a process
register of serial link and link[data] indicates the contents of a data
register of a serial link. The CPU services any input requests or
output requests on lines 94 and 95 before executing the next process on
the waiting list. In the present case, when the CPU 12 of
microcomputer 11b next ceases executing a process, it will find an
input request from channel 67 and this will cause the CPU to carry out
lines 5 to 8 of the definition of the procedure "wait". This has the
effect, due to lines 5 and 8, of rescheduling process Y, line 6 loads
the value NIL into the process register as shown at 67b and line 7 causes the data from the input data register 66 to be transferred to the workspace of process Y. In addition, the input control logic 73 provides an output signal to the output control logic which causes the output channel 65 to transmit an acknowledge signal via line 84 to the microcomputer 11a. The input control logic 73 on microcomputer 11a then provides an output to the output control logic 70 causing an output request to the CPU 12 on the microcomputer 11a. When the processor of microcomputer 11a next deschedules a current process, the processor looks for any input or output requests from the link logic as previously described and will find an output request from the logic associated with the output register 63 and will therefore follow the sequence defined in lines 11 to 13 of the definition of the procedure "wait". This causes the workspace pointer of process X to be loaded into the OREGTR, the special value READY is loaded into the process register 63 and process X is rescheduled. It will be seen that in this sequence the output process and data registers 63 and 62 go through the changes indicated by the subscripts a, b and c in Figure 5 whereas the input process and data registers undergo the sequence illustrated by the subscripts a to d illustrated in Figure 5.

If in the above example data had been transferred via line 81 to the input data register 66 of the microcomputer 11b before process Y carried out an input instruction, the input control logic of microcomputer 11b would, following the above table for the input control state machine, change the value of the input process register 67 to special value READY so that subsequent execution of the input instruction by process Y would locate the special value READY and process Y would carry out the sequence according to lines 6 and 7 of the definition of input. This would mean that data would be transferred from the input data register 66 to the workspace of process Y and the special value NIL would be loaded into the process register 67 and process Y would continue without interruption.

It will be appreciated that in the above examples, the microcomputer is arranged to effect communication between processes when both processes
are at corresponding stages in their program sequences. If either process attempts to transmit data when the other is not at a corresponding stage, the process which initiates the transmission is held up until both processes are at corresponding stages. Each process involved in the communication requires only one input or output instruction in its program sequence so that neither process is descheduled more than once in order to effect the data transmission. Furthermore, for communications between processes on the same microcomputer, the memory space occupied by communication channels is reduced in requiring only one word of memory for each channel. The input and output instructions allow for addressing of workspace locations. This is advantageous in reducing the number of instructions or operations required to permit process to process communication and it permits more efficient transfer of data into process workspace locations.
CLAIMS:

1. A microcomputer comprising memory and a processor arranged to execute a plurality of concurrent processes, each in accordance with a program consisting of a plurality of instructions for sequential execution by the processor, each instruction designating a required function to be executed by the processor, said processor comprising (1) a plurality of registers and data transfer means for use in data transfers to and from said registers (2) means for receiving each instruction and loading into one of the processor registers a value associated with the instruction, and (3) control means for controlling said data transfer means and registers in response to each instruction received to cause the processor to operate in accordance with the instruction, wherein the microcomputer includes:—

(a) scheduling means to enable the processor to share its processing time between a plurality of concurrent processes, said scheduling means comprising:—

(i) means for identifying one or more processes which form a collection awaiting execution by the processor

(ii) means for descheduling a process by interrupting execution of the current process

(iii) means for scheduling a process by adding it to said collection, and

(b) communication means to permit data transmission from one process to another when both processes are at corresponding stages in their program sequences, an outputting process operating to output data in response to an output instruction in its program and an inputting process operating to input data in response to an input instruction in its program, said communication means including:—

(i) a channel comprising store means for holding a value indicating whether a process has executed an instruction to effect data transmission using that channel,

(ii) means, responsive to execution of an input or output instruction by one of the processes involved in the data transmission when said one process is the current process for testing the contents of said channel and arranged to operate said means to deschedule the current process
if the channel does not contain a value indicating that the other process involved in the data transmission has reached a corresponding program stage, the communication means being arranged such that an outputting process requires only one output instruction in its program sequence and an inputting process requires only one input instruction in its program sequence, whereby either process is not descheduled more than once, in order to effect such data transmission.

2. A microcomputer according to claim 1 in which the scheduling means includes means for indicating the current process which is being executed by the processor and the communication means includes means for loading into said channel an identification of the current process if that current process is descheduled as a result of said testing of the contents of the channel.

3. A microcomputer according to claim 2 in which said memory provides for each process a workspace having a plurality of addressable locations including locations for recording variables associated with the process, and in which one of said processor registers is arranged to hold a workspace pointer value identifying an address of the workspace of the current process.

4. A microcomputer according to claim 3 in which the said identification to be loaded into said channel is said workspace pointer value.

5. A microcomputer according to any one of the preceding claims wherein said channel is one of a plurality of channels, each comprising an addressable store location.

6. A microcomputer according to any one of claims 1 to 5 wherein the communication means is arranged to permit data transmission between processes which are executed on the same microcomputer and said channel comprises a memory location.
7. A microcomputer according to claim 6 wherein said microcomputer comprises an integrated circuit device and the channel comprises a memory location in memory on the same integrated circuit device as the processor.

8. A microcomputer according to claim 6 or claim 7 wherein the store means forming a channel consists of a single memory location having a bit size equal to that necessary for holding the address of another memory location.

9. A microcomputer according to any one of claims 2 to 8 wherein the means for testing the contents of a channel is arranged, on detection of the identification of a descheduled process in said channel, to operate said means to schedule the process identified by said identification.

10. A microcomputer according to any one of claims 6 to 9 in which said communication means is arranged to store in a first memory location associated with the current process data to be output by a current outputting process if that current process becomes descheduled as a result of the test on the contents of the said channel following execution of an output instruction.

11. A microcomputer according to claim 10 including means for transferring stored data from said first memory location associated with the descheduled process to a second memory location associated with a current inputting process in response to execution of an input instruction by an inputting process which locates in said channel an indication of the process associated with said first memory location.

12. A microcomputer according to any one of claims 6 to 9 in which said communication means is arranged to store data output through a channel by a current process in a memory location associated with an inputting process if that inputting process has been descheduled as a result of the execution of an input instruction using the same channel.
13. A microcomputer according to claim 4 and 8 in which the or each channel is arranged to hold a workspace pointer value for a descheduled process or a first special value indicating that no process has yet executed an input or output instruction using that channel, and means is provided to store output data in a memory location associated with a descheduled output process using that channel, and to store in a memory location associated with an input process, input data after execution of an input instruction by an inputting process.

14. A microcomputer according to any one of claims 1 to 5 wherein the communication means is arranged to permit external data transmission between processes which are executed on different microcomputers and said channel comprises a first register forming part of an external communication link.

15. A microcomputer according to claim 14 in which each external communication link has an output channel and an input channel each including a second register for holding data to be output or input through the link.

16. A microcomputer according to claim 14 or claim 15 in which the or each register of a link is addressable as a memory location.

17. A microcomputer according to any one of claims 14 to 16 in which means is provided to load into the first register of a link for use by an inputting process, a first special value to indicate that no process has yet executed an input instruction using that channel or an indication of a process that is descheduled after executing an input instruction using that channel.

18. A microcomputer according to claim 15 and 17 in which means is provided responsive to receipt of data in a second register of an input channel to load into said first register of the input channel a second special value if no inputting process has executed an input instruction using that channel before receipt of data in said second register.
19. A microcomputer according to any one of claims 14 to 16 in which means is provided to load into the first register of a link for use by an outputting process, a special value to indicate that no process has yet executed an output instruction using that channel or an indication of a process that is descheduled after executing an output instruction using that channel.

20. A microcomputer according to claims 15 and 19 in which the communication means is arranged to respond to detection of said special value in a first register of a link by descheduling a current process if executing an output instruction, and loading data to be transmitted into the second register of the link.

21. A microcomputer according to claims 15 and 18 in which the communication means is arranged to respond to detection of said first special value in a said first register of a link by descheduling a current process if executing an input instruction and loading into said first register an indication of that process.

22. A microcomputer according to claim 18 or claim 21 in which the communication means is arranged to respond to detection by an inputting process of said second special value in a said first register of a link by transferring data from the second register of the link to a memory location associated with the inputting process and permitting the processor to continue execution of the program sequence of the inputting process.

23. A microcomputer according to any one of claims 15 to 22 in which the communication means includes means to transfer data from a said second register of a link to a memory location associated with an inputting process on a first microcomputer and means for generating an acknowledgment signal for transmission to an outputting process on a second interconnected microcomputer.

24. A microcomputer according to claim 23 in which each link includes control means arranged to respond to receipt of data or an
acknowledgment signal by a second register associated with an input channel of a link for generating a request signal to the processor of the microcomputer providing the link, to schedule any process which according to the contents of the first register of that link is descheduled.

25. A microcomputer according to claim 24 in which said control means comprise programmed logic array devices having a programmed succession of output signals.

26. A microcomputer according to claims 3 and 8 wherein each channel comprises a single addressable word location in memory arranged to hold a special value indicating that no process has yet executed an input or output instruction using that channel or a pointer indicating the workspace address of a process which is descheduled after executing an input or output instruction using that channel.

27. A microcomputer according to claim 26 including means for transferring data from an address in the workspace of an outputting process to an address in the workspace of an inputting process without descheduling the current process if the current process on executing an input or output instruction locates a pointer to the workspace of a descheduled process in said channel.

28. A network of microcomputers comprising a plurality of microcomputers as claimed in any one of the preceding claims interconnected to permit data transmission on execution of input and output instructions.

29. A method of effecting data transmission between two processes executed on a microcomputer wherein an addressable memory location is used as a channel to enable the processes to be at corresponding stages in their program sequences when the data transmission occurs, said method comprising (i) executing a current process in accordance with a sequence of program steps; (ii) indicating which process or processes await execution by the processor, and (iii) executing an output
instruction by a process wishing to transmit data, the processor in response to the output instruction (a) identifying the address of the channel to be used for the data transmission (b) testing the contents of said channel to detect either an indicator of a descheduled process awaiting an input through the channel or a special value indicating that no process is awaiting use of the channel, (c) in response to said special value storing the data to be transmitted in a region of memory associated with the process, loading into the channel an indicator of that process and descheduling the current process and (d) in response to an indicator of a waiting process, transferring the data to be transmitted to a region of memory associated with the waiting process, loading said special value into the channel and causing the waiting process to be rescheduled for execution by the processor.

30. A method of effecting data transmission according to claim 29 wherein a process wishing to receive data executes an input instruction, the processor in response to the input instruction (a) identifying the address of the channel to be used for data transmission (b) testing the contents of said channel to detect whether an indicator of a descheduled process awaiting an output through the channel or a special value indicating that no process is awaiting use of the channel, (c) in response to said special value loading into the channel an indicator of that process and descheduling the current process and (d) in response to an indicator of a waiting process, transferring the data to be transmitted from a region of memory associated with the waiting process to a region of the memory associated with the current process, loading said special value into the channel and causing the waiting process to be rescheduled for execution by the processor.
Fig. 3.
Fig. 5.
# INTERNATIONAL SEARCH REPORT

**International Application No:** PCT/GB 84/00124

### I. CLASSIFICATION OF SUBJECT MATTER

According to International Patent Classification (IPC) or to both National Classification and IPC

**IPC:**

<table>
<thead>
<tr>
<th>Classification System</th>
<th>Classification Symbols</th>
</tr>
</thead>
<tbody>
<tr>
<td>G 06 F 9/46</td>
<td>G 06 F 9/46; G 06 F 15/16; G 06 F 9/22; G 06 F 15/06</td>
</tr>
</tbody>
</table>

### II. FIELDS SEARCHED

Minimum Documentation Searched

<table>
<thead>
<tr>
<th>Classification System</th>
<th>Classification Symbols</th>
</tr>
</thead>
<tbody>
<tr>
<td>G 06 F 9/46</td>
<td>G 06 F 9/46; G 06 F 15/16; G 06 F 9/22; G 06 F 15/06</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>Euromicro 1981 Paris, September 8-10 (Amsterdam, NL) Conte et al.: &quot;TOMP80 - A multiprocessor prototype&quot;, pages 401-410, see page 406, column 2, lines 3-20; figure 8; page 407, column 1</td>
<td>1,11,12,19-22,28,30</td>
</tr>
<tr>
<td>A</td>
<td>Nachrichtentechnische Zeitschrift NTZ, vol. 35, no. 12, December 1982 (Berlin, DE) Schan: &quot;Prozesskommunikation beim IAPX 432&quot;, pages 744-748 see page 744, column 3, lines 61-67, page 746, column 1, lines 42-64; column 3, lines 24-34; figure 4; page 747; column 1, lines 40-62; figures 6,7; page 748, column 1, lines 12-23</td>
<td>5-8,26</td>
</tr>
<tr>
<td>A</td>
<td>US, A, 3805247 (ZUCKER) 16 April 1974 see column 1, lines 6-10; column 4, lines 12-24; column 12; column 13, lines 1-33; figures 4,9</td>
<td>1-4</td>
</tr>
</tbody>
</table>

* Special categories of cited documents:
  - **A** document defining the general state of the art which is not considered to be of particular relevance
  - **E** earlier document but published on or after the international filing date
  - **L** document which may throw doubts on priority claim(a) or which is cited to establish the publication date of another citation or other special reason (as specified)
  - **O** document referring to an oral disclosure, use, exhibition or other means
  - **P** document published prior to the international filing date but later than the priority date claimed

**T** later document published 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; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step

**Y** document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

**A** document member of the same patent family

### IV. CERTIFICATION

**Date of the Actual Completion of the International Search:** 26th July 1984

**Date of Mailing of this International Search Report:** 21.08.84

**International Searching Authority:** EUROPEAN PATENT OFFICE

**Signature of Authorized Officer:** G.L.M. Kruydenberg
<table>
<thead>
<tr>
<th>Category</th>
<th>Citation of Document, with indication, where appropriate, of the relevant passages</th>
<th>Relevance to Claim No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Software &amp; Microsystems, vol. 1, October 1982, no. 6 (Camberley, GB) Shoja et al.: &quot;Some experiences of implementing the ADA concurrency facilities on a distributed multiprocessor computer system&quot;, pages 147-152, see page 147, column 1, lines 21-23; column 2, lines 32-35; page 148, column 1, lines 1-6; page 149, column 1, lines 33-46; column 2, lines 27-43; page 150, column 1, lines 15-39; page 151, column 2, lines 56-61</td>
<td>1, 14, 23, 24</td>
</tr>
<tr>
<td>A</td>
<td>Electro/80 Conference Record, 13-15 May 1980, vol. 5 (Boston, US) Smith: &quot;Multi-function single chip micro-computers&quot;, pages 1-10 see page 1, column 1, lines 1, 2; page 6, column 2, lines 22-30; page 7, column 1, lines 8-15; figure 7</td>
<td>16, 25</td>
</tr>
<tr>
<td>A</td>
<td>Electronics, 30 November 1982 (New York, US) Taylor et al.: &quot;Process-oriented language meets demands of distributed processing&quot; pages 89-95, see page 90, column 1, lines 24-32, 51-61; column 2, lines 1-19; page 94, column 2, lines 1, 2 (cited in the application)</td>
<td>1, 5, 14, 23</td>
</tr>
<tr>
<td>P, X</td>
<td>Electronics International, vol. 56, no. 23, 17 November 1983 (New York, US) Barron et al.: &quot;Transputer does 5 or more MIPS even when not used in parallel&quot;, pages 109-115; see page 110, column 1, lines 20-24; column 2, lines 20-26; page 111, column 1, lines 19-31; column 2, lines 52-56; page 112, column 1, lines 18-22; column 2, lines 10-26, 44-52; figure 2</td>
<td>1-4, 9, 11, 12, 17-22</td>
</tr>
</tbody>
</table>
This Annex lists the patent family members relating to the patent documents cited in the above-mentioned international search report. The members are as contained in the European Patent Office EDP file on 13/08/84.

The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

<table>
<thead>
<tr>
<th>Patent document cited in search report</th>
<th>Publication date</th>
<th>Patent family member(s)</th>
<th>Publication date</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>GB-A- 1412690</td>
<td>05/11/75</td>
</tr>
<tr>
<td></td>
<td></td>
<td>JP-A- 49062046</td>
<td>15/06/74</td>
</tr>
</tbody>
</table>

For more details about this annex:
see Official Journal of the European Patent Office, No. 12/82