TO: MSPM Distribution
FROM: J. H. Saltzer
SUBJ: BQ.6.00 and BQ.6.02
DATE: 07/20/67

Sections BQ.6.00 - BQ.6.02 together are being revised to provide an overview of the actively designed interprocess communication facility - (the former sections gave an overview of a projected interprocess communications facility).

Included is a new description of "how to use interprocess communication".

Published: 07/20/67
(Supersedes: BQ.6.00, 03/09/67;
BD.8.00, 08/29/66)

Identification

Overview of the Interprocess Communication Facility
Michael J. Spier, B. A. Tague

Purpose

In the life of a Multics process, the need arises at least once for some information to be provided by some other process. In order to permit parallel processing, a Multics "program" (task, job, console session) executes as a collection of separate processes known as a "process group" and consisting of an overseer process, one or more working processes and zero or more device manager processes which are interactive and dependent upon one another for information. This organization requires effective control communication between processes.

The Traffic Controller's process exchange entries block and wakeup provide basic control communication services (see MSPM sections BJ.3). The Interprocess Communication Facility described in this overview and in the following MSPM sections BQ.6.01-08 is an extension of the process exchange and allows a limited amount of control data to be associated with each call to "block" and "wakeup".

This overview discusses the problems involved in interprocess communication and the solutions adopted.

Introduction

The terms "event", "event channel", "event indicator" and "signalling of an event signal over an event channel" are fundamental to interprocess communication and are defined below.

An event is anything which is observed during the execution of some process (sending process) and which is of interest to some other process (receiving process) or perhaps to some other procedure of the first process. An event is a unique occurrence and can happen only once. Associated with an event is an event id which is a unique bit string. Furthermore, an event is associated with an event channel, which has a unique bit string name known as the event channel name. Several events of a similar kind may be associated with a common event channel which has a single event channel name.

An event channel is a first-in first-out queue of event indicators. The term "event indicator" refers to the control information associated with an event. Its value may vary, and usually becomes more precise as it is passed from the sending process, via the Interprocess Communication Facility, to the receiving process. An event indicator always implies a unitary value associated with an event and kept track of (internally) by the Interprocess Communication Facility, insuring that all events communicated to a receiving process are remembered.

An event channel is a data base shared between a receiving process and one or more sending processes. An event channel is always associated with exactly one receiving process and exactly one event channel name. It is the receiving process that creates, maintains, reads and eventually deletes an event channel.

Whenever reference is made to the "signalling of an event over an event channel", it implies the writing of an event indicator into an event channel and a call to "wakeup" for the receiving process.

Whenever reference is made to the reception of an event signal over an event channel, it implies the receiving process' awakening and reading of an event indicator out of an event channel.

Discussion

The Traffic Controller provides the basic tools for interprocess communication in its process exchange entries block and wakeup. A process which waits for some event to happen calls "block", to awake knowing that some other process has signalled to it the occurrence of an event. Still, it cannot be certain that the signalled event is the event it has waited for unless the sending process, by some private agreement between both processes, manages to associate the wakeup signal with some positive identification as to the nature of the event.

The Interprocess Communication Facility provides such event identification. By a systemwide convention*, no procedure within a process is allowed to call "block" or "wakeup" directly. The Interprocess Communication Facility's two main modules, the Event Channel Manager (ECM) and the Wait Coordinator (WC), are provided to handle calls to "wakeup" and "block" respectively.

A process always* blocks itself (and wakes up) in the WC which intercepts all wakeup signals sent to that process. The ECM associates every call to wakeup with a limited amount of control information (event indicator) which it writes into a systemwide accessible data base (event channel) that is provided by the receiving process. The WC is therefore capable of determining whether or not a received event signal is of current interest whereupon it either transmits the signal to the waiting procedure or continues waiting (calls "block" again).

By convention, a receiving process has complete control over its event channels. A sending process may not access any of these event channels unless it belongs to a process-group which has been granted permission to do so by the receiving process. Such permission is granted by making it known to the ECM (by means of a special call) which process-groups may have access to that channel and by communicating the event channel's name to the future sending processes. This communication is necessary due to the event channel name's uniqueness and is known as "basic interprocess communication".

By basic interprocess communication we refer to _any_ method of communication (going as far as interconsole messages) by which a receiving process may make an event channel name (and perhaps its own process id) known to a future sending process. This is usually done by having both processes agree upon a specific location within a given data base which is to serve as a mailbox. The future sending process has the location pre-set to zero and interprets any non zero value which it finds in there as being the receiving process' basic interprocess communication message. (Basic interprocess communication is further explained in MSPM section BQ.6.01).

*Note: A few hardcore supervisor routines do not make use of the Interprocess Communication Facility. Consequently, whenever a process executes in one of these routines, it does not respect the conventions mentioned above.

As soon as interprocess communication is established as explained above, the Facility works in the following way:

Every wakeup for a receiving process is associated with an event indicator which is written into an appropriate event channel. The receiving process associates every call directed to "block" with an event channel name and makes the call through its wait coordinator, which calls "block" only if the specified event channel has no event indicator written in it.

This organization allows event signals to be sent and received asynchronously. Any number of sending processes may signal any number of events of any nature to a receiving process. The Interprocess Communication Facility acts as a buffering system, making sure that no event signal is lost and that interested procedures of the receiving process are properly notified of the right event at the right time.

Implementation

Every process has a pair or segments named "event channel table" and "working queue" (together usually referred to, in this write up, as the Event Channel Table) which contain all the process' event channels. The reason for having two segments rather than one is due to technical considerations and has no conceptual significance. (See MSPM section BQ.6.03.) A process' event channel table is known and accessible to member processes of the same process group at a ring 1 level of protection, to all other processes (interprocess group communication) at a ring 0 level of protection.

Event channels are manipulated by the Event Channel Manager (ECM) which resides in ring 1. A number of ECM modules used for interprocess group communication reside in ring 0. Only one ECM procedure, named set_event (dedicated to the signalling of events), may access another process' event channel table. All other ECM modules which do the creation, maintenance, reading and deletion of event channels can be called by the receiving process only, to access its own table. Event channels are accessed within the event channel table by name and protected by software against illegal access by unauthorized processes. Furthermore, event channels are protected by a pair of associated ring numbers (one for the sending process, the other for the receiving process) which allow access to the event channel from privileged rings only in the respective processes. (See MSPM section BQ.6.05.)

Event channels have mode and type attributes. An event channel's signalling mode determines the amount of precise information that is associated with each signalled event. There are two signalling modes. The event-count mode associates only a unitary value with each event, the event-queue mode (an extension of the event-count mode) provides as supplementary information an event id and the sender's process id (remember that more than one process may signal over a single event channel).

An event channel has two type attributes, a sending type and a receiving type, which are independent of one another and which determine the way in which an event is signalled or received, respectively. The sending type indicates whether the channel is dedicated to the signalling of system interrupts (device-signal channel) or events internal to memory (communication channel). The receiving type differentiates channels which have to be explicitly interrogated (event-wait channels) from channels which automatically cause a procedure of the receiving process to be called whenever an event is signalled over them (event-call channels).

An event channel's signalling mode is specified at channel creation time and remains unchanged thereafter. The type attributes are declared after the channel was created and may be redeclared (by the receiving process) at any time. A newly created event channel has by default the communication/event-wait type attributes.

Event channels are discussed in detail in MSPM section BQ.6.03, the Event Channel Manager in MSPM section BQ.6.04.

Communication event signalling is straightforward in that the sending process writes the event indicator into the appropriate event channel directly. System event signalling is more complex due to the fact that device signal interrupts are liable to occur at any moment and be intercepted by any process that is currently running.

By a special system convention, processes interrupted by device signals put event indicators into special mailboxes, associated with all connected devices and known as device signal channels. These channels are located in the Device Signal Table which is wired down in the hardcore ring. Every device signal channel is coupled to a regular event channel belonging to the process which is currently authorized to use the device. The receiving process (whenever it executes in the WC) looks up the device signal channels associated with it and transcribes their contents into the corresponding event channels (which have the device-signal-channel sending type). The Device Signal Table is manipulated by the Device Signal Table Manager, residing in ring 0 (see MSPM section BQ.6.07).

The Interprocess Communication Facility is used by the Give Call Facility (described in MSPM section BQ.6.08) which allows procedure to procedure calls to be made over process boundaries. By using this facility, a procedure within a process may call (by its symbolic name) a procedure within a member process of the same process group, or within the first process.

System Interface

Figure 1 is a simplified display of the Interprocess Communication Facility and some of its users. Certain connections (as explained below) have been oversimplified for the sake of clarity. The vertical line is the boundary between the sending (right) and the receiving (left) processes. The horizontal lines represent ring "walls". The device signal table and the receiving process' event channel table are available to both processes, in the hardcore and administrative rings respectively. It is assumed that both processes belong to the same process group. In the case of interprocess-group communication, the event channel table is located in the receiving process' administrative ring and is accessible to the sending process in the hardcore ring. Normal event communication is done by having the sender call its set_event procedure in ring 1, which writes an event indicator into the receiver's event channel table and calls wakeup for the receiving process. The interrupt handlers write an event indicator into the systemwide device signal table via the Interprocess Group Event Channel Manager (path not shown in figure 1) and also call "wakeup" for the receiving process.

On the receiving process' side, the key module is the wait coordinator which performs two functions in the given order:

a. Calls the interprocess group event channel manager to transcribe the receiving process' device signal channel's contents into associated event channels in the event channel table.

b. Calls the event channel manager to read the event channel which is currently waited on, and in the case of a negative result calls the traffic controller's entry "block".

As can be seen, the I/O system is a user of the Interprocess Communication Facility. It converts the intercepted I/O interrupts into event signals which are then retrieved by the wait coordinator. On the other hand, the basic file system does not use the Facility, it accesses the traffic controller's entries "block" and "wakeup" directly.

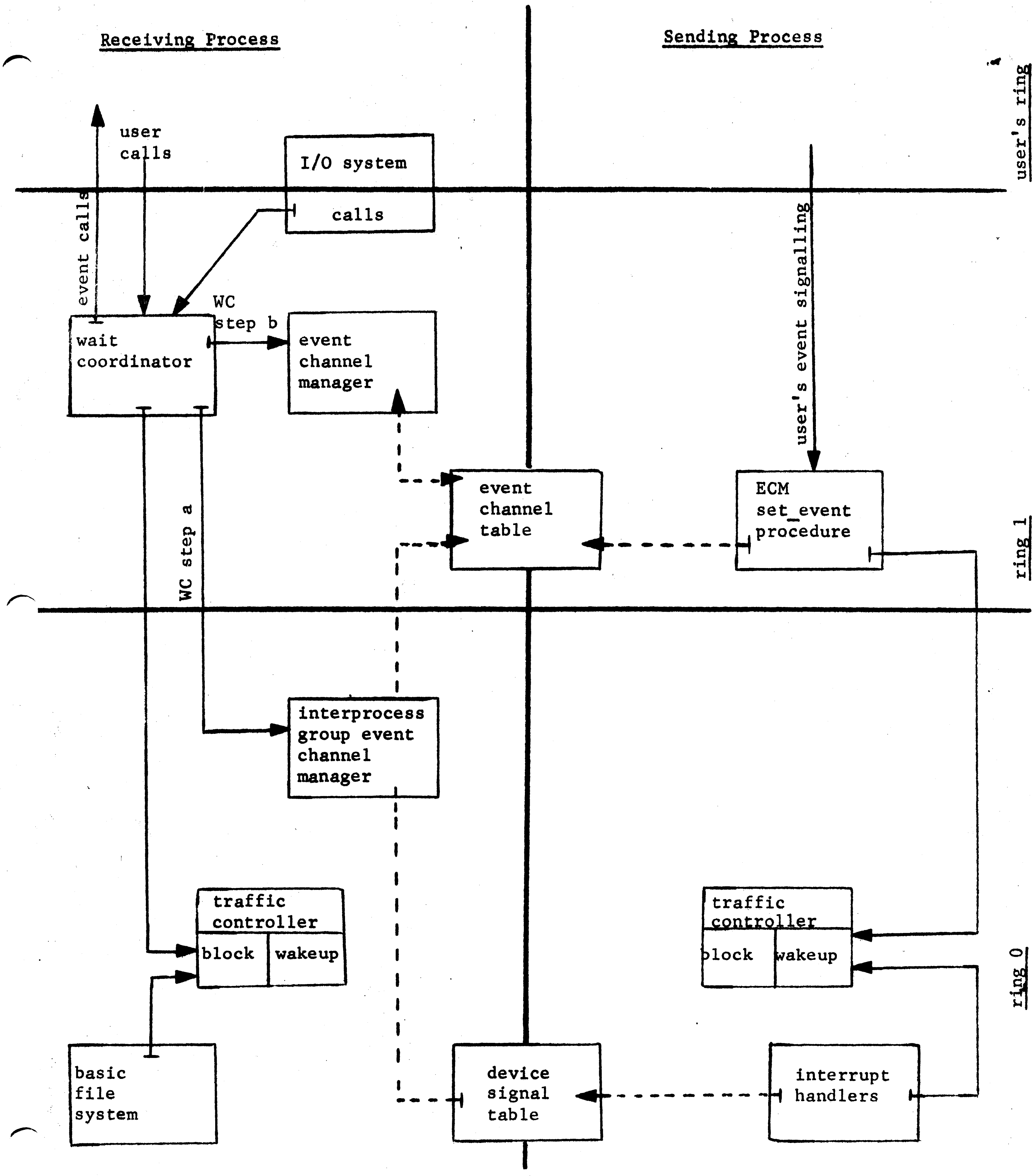
Receiving Process
Sending Process
user's ring
user calls
event calls
I/O system
calls
user's event signalling
WC step b
wait coordinator
event channel manager
WC step a
event channel table
ECM set_event procedure
ring 1
interprocess group event channel manager
traffic controller
block
wakeup
traffic controller
block
wakeup
ring 0
basic file system
device signal table
interrupt handlers

Figure 1