Published: 11/1/66

Identification

Calendar Clock Wakeup Management
J. H. Saltzer, T. H. Van Vleck

Purpose

Associated with each Calendar Clock in the system is an alarm clock register which may be set by program in Master Mode. This register is continuously compared with the contents of the Calendar Clock and a system interrupt is generated when a match occurs. This alarm clock register is shared by all processes wishing to receive timed wakeup signals. Sharing is accomplished by the Calendar Clock Coordinator Procedure, which maintains an ordered list of requested wakeup times, and a system process, the Calendar Clock Manager Process. These procedures use the standard Wait-Coordinator conventions described in BD.8.01.

Calendar Clock Coordinator

The Calendar Clock Coordinator module is a resource management procedure which may be called from within any process. It has two entry points to arrange wakeups as follows:

```
call set_wake_time (calendar_time,   event_name);
```

and

```
call set_wake_interval (calendar_interval, event_name);
```

where

calendar_time is the time at which a wakeup is desired;

calendar_interval is an interval (measured from the instant of time that the call is received) after which a wakeup is desired;

event_name is a variable in which the Calendar Clock Coordinator procedure will place the unique event identification tag it receives from the Wait-coordinator.

The process may now, if it wishes, wait for the desired time to arrive by calling upon the Wait-coordinator and presenting the event identifier as described in BD.8.01, e.g.,

```
call wait  (event_name);
```

The Calendar Clock Coordinator procedure has a third entry point, to discard a previously requested wakeup. It is

called by

```
call reset_clock_wakeup (event_name);
```

giving as the argument "event_name" the event identifier received on the previous call to set_wake_time or set_wake_interval. This entry point will discard the wakeup request and inform the Wait-Coordinator that the event indentifier is no longer of interest by calling Reset-event.

A single process may make several set_wake_calls to arrange wakeups at several different times if it desires. It may then selectively delete any or all of the requested wakeups by use of the reset calls.

The two entries, set_wake_time, and set_wake_interval, permit two convenient methods of specifying when a wakeup is to occur. However, there is also an important difference in interpretation of the two calls in the case in which a process is temporarily interrupted, then restarted. (The interruption may arise, for example, by a quit signal or a normal system shutdown.) When the process is restarted, a wake_interval is continued without counting the interval of time that the process was interrupted. On the other hand, a wake_time is honored as an absolute time; the intervening time is counted as though the process had not been interrupted. Thus, a call to set_wake_interval for 1 minute, followed by a quit and save sequence 20 seconds later, will result in a wakeup signal 40 seconds after the process is resumed, even if resumed two days later.

On the other hand, a call to set_wake_time for 2:00 p.m., Friday, will cause a wakeup to occur at 2:00 p.m., Friday even if the system is off the air all day Thursday. (If an interrupted process with a pending wakeup time is resumed after the wakeup time has passed, it will receive its wakeup immediately upon resumption.)

To facilitate saving and restoring one process's state by another process, two additional entry points are provided. The call

```
call save_clock_wakeup (process_id, event_name, data)
```

searches the Wake-Time Table for the entry specified by "process_id" and "event_name", transmits the data to a structure pointed to by "data", and removes the entry from the Wake-Time Table.

The call

```
call restore_clock_wakeup (process_id, data)
```

replaces the appropriate entry in the Wake-Time Table. One use for these entries is in the procedure invoked when one process quits another. Every wakeup for a process being quit must be reset so that the process will stay quit. For each event in the Event Table (BD.8.01) of such a process which points to the calendar clock procedure, the quit-handling procedure will call "save_clock_wakeup," saving the entries in an auxiliary data base. When the process is restarted, "restore_clock_wakeup" will be called to start the clock again. Note that it must recompute the proper wakeup time for wakeup_intervals.

The Calendar Clock Wake-Time Table

The Calendar Clock Wake-Time Table is a system-wide data base which is maintained by the Calendar Clock Coordinator. It is a list, in order, of all wakeups which have been requested by all processes of the system. For each wakeup, there is an entry containing the following quantities:

1. Calendar Clock time of requested wakeup. This is the time at which the wakeup should occur.

2. Time/Interval switch. This switch, if set to Interval, indicates that item 1 represents the end of an interval requested. If set to time, item 1 represents the absolute time requested.

3. Identification of process requesting wakeup.

4. Event-name. This is the unique identification number associated with completion of the wakeup request.

The Wake-Time Table is an ordinary linear list, with the wakeup farthest in the future at the beginning of the list, all entries for closer times are moved down one position.

The header of the Wake-Time Table contains two items:

1. Number of entries in the table. This number tells how many wakeups have been requested, and enables the manager process to find the earliest wakeup.

2. Interlock. An interlock switch is necessary to keep more than one process at a time from modifying the table. A process wishing to modify the Table (for instance, to set a wakeup) must use the standard interlock procedures (BD. ?) to lock the table before modifying it; if the table is already locked, these procedures will block the process until the table becomes free. (Note that this interlock is set and reset only by the procedures described in this MSPM section.)

The Calendar Clock Manager Process

The Calendar Clock Manager process is a system process which is responsible for system interrupts coming from the calendar clock. When such an interrupt arrives, the process which happens to be running at the time schedules the Calendar Clock Manager process. (See section BK.2.05, the Calendar Clock Interrupt Handler.)

When the Calendar Clock Manager gains control, it signals the first process in the wake-time table by calling entry point complete (event-name, process-i.d.); in the Wait-Coordinator. It then deletes that entry in the Wake-Time-Table, and sets the hardware calendar alarm clock to the value indicated by the next entry in the Wake-Time-Table. (If the time indicated has already passed, the Calendar Clock Manager merely wakes up the indicated process and goes on to the next entry in the Wake-Time-Table.)

The Calendar Clock Manager Process is initiated with an inter-process "give-call" from the System Operator at system initialization time. When first initiated, it places its own process identification number in the data base of the Calendar Clock Interrupt Handler. It then initializes the Wake-Time-Table, either to an empty state or by reference to a previously stored state.

At system shutdown time, a "shutdown" flag is placed in the Wake-Time-Table by the system operator, and the Calendar Clock Manager is wakened. It saves the current status of the waketime table, and then performs an inter-process return to its caller.

Subroutine Set-alarm

The hardware calendar clock is actually set by a master mode, machine language subroutine, named "set-alarm". Set-alarm is called with three arguments, as follows:

```
Call set-alarm (time,clock,err)
```

Where time is an integer to be placed in the alarm clock register, and clock is a logical system clock number. System Clocks are numbered as follows:

1. Primary system clock
2. First backup clock
3. 2nd backup clock

   etc.

If "clock" is higher than the number of clocks available to the system, variable "err" will be set to '1'b and no clock will be set.

Subroutine "set-alarm" determines which physical clock to address by referring to the system communication table. (See MSPM Section BK.1.04)

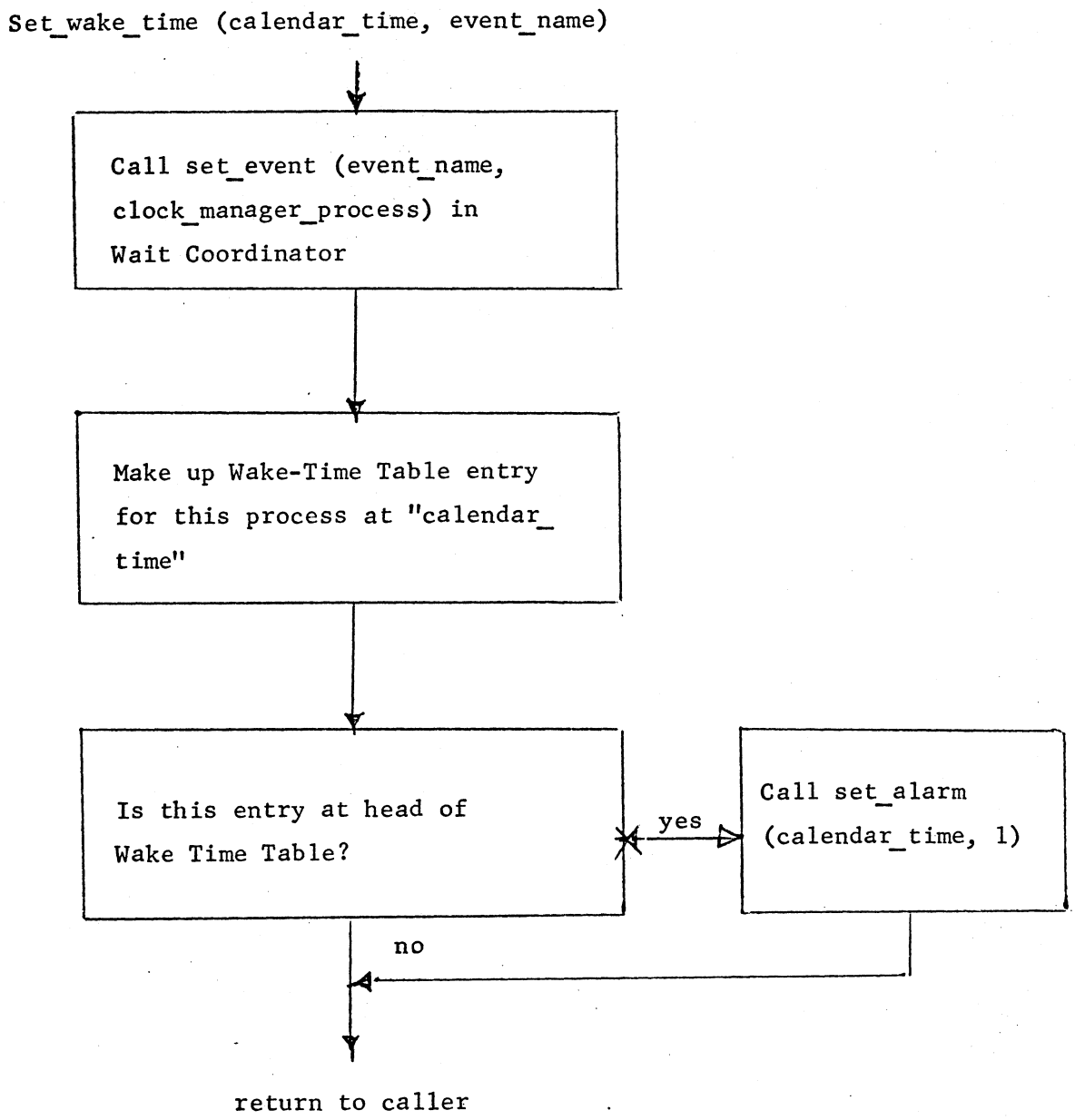

Figure 1 -- Calendar Clock Coordinator Procedure

Figure 2-- Procedure followed by Calendar Clock Manager Process