2.2 Scheduling an event

An event can be scheduled at any point by calling **`EventDoCallback`**. This function takes as arguments an **`EventCallbackHdr`**, as well as the procedure to call, the argument for the procedure, and the amount of simulated time until the event should occur.

An event can be removed or rescheduled by calling **`EventCallbackRemove`** or **`EventCallbackUpdate`**, respectively. Note that **`EventCallbackUpdate`** will only schedule events to occur earlier than originally specified, not later. To reschedule an event for a later time, you must remove the event and then schedule it with a second call to **`EventDoCallback`**.

SimOS removes events from the event queue when they are triggered. Therefore procedures which must be called regularly should reschedule themselves. The same **`EventCallbackHdr`** structure may be reused each time.

Dennis Geels
IBM Austin Research Lab
512 838 9356
arlsup07@austin.ibm.com

Timed Callback in SimOS-PPC

1 Introduction

Timed callback routines allow SimOS to postpone procedure calls until a specified future time. Simulated events which require delay or an asynchronous interface can be scheduled and added to a queue, which the simulator monitors periodically. When the specified amount of simulated time has expired, the event occurs.

The timed callback routines provide a simple, centralized interface for scheduling events, providing a great deal of flexibility yet requiring only a single entry point from the various cpu simulator main loops.

SimOS uses event callbacks to simulate delay in external devices. For example, fulfilling disk requests immediately would give unrealistic timing statistics and possibly cause problems for a simulated system expecting delay. Rather, the simulator schedules the disk operation using the timed callback routines and perhaps a disk simulator module for more accurate delays. The event is later triggered, completing the disk operation and raising an interrupt on the simulated machine.

Event callbacks are also used to schedule actions on behalf of the simulator, such as polling the inputs for the simulated console. A polling event is scheduled by the simulator, which, when triggered, checks the input buffers for the console, raises an interrupt if necessary, and reschedules itself.

2 Implementation

SimOS uses a simple queue to schedule events, sorted by time. The main loop of the cpu simulators make regular calls to **`EventProcess`**, often as the alias **`EventProcessSingleQueue`**. This function will initiate any pending events which were scheduled at or before the current time.

2.1 Events

An event consists of a procedure to call, the argument to pass into the procedure, and the number of the cpu which the event affects. This information, along with the scheduled time, is stored in an **`EventCallbackHdr`** structure. SimOS uses the structure in its internal queues as well as passing it as an additional argument to the event procedure.

SimOS usually keeps all events in a single queue. However, when embra is running in parallel mode, SimOS maintains an array of queues, one for each cpu. The macros **`EVENT_LOCK`** and **`EVENT_UNLOCK`** are then used to synchronize access to the queues.