Capsim™ Simulations and Buffer Management

The Following is a Contribution by
Bill Hughes, GE Corporate Research and Development,
Schenectady, New York.

Product Number CAP213

XCAD Corporation

Running Long Simulations

Under most circumstances Capsim manages the buffer space required for a simulation transparently to the user. The program creates a buffer for each input to a Star or Galaxy and space is allocated, as needed, as the simulation proceeds. There is a maximum buffer size that the program will permit, however, which can be set by the systems operator to a size appropriate for the memory capacity of the computer being used. In long simulations this maximum buffer size can be exceeded. In these cases the user will have to take specific action when setting up the simulation to make sure buffer overflow will not occure. This requires an understanding of how Capsim orders a simulation and how stars generate and consume samples.

The possibility of buffer overflow should be considered anytime the number of samples to be processed in some part of the simulation exceeds the maximum buffer size (typically ~12,000 samples). Figure 1 represents a simple case that can be expected to cause buffer overflow if the total number of samples is large enough.

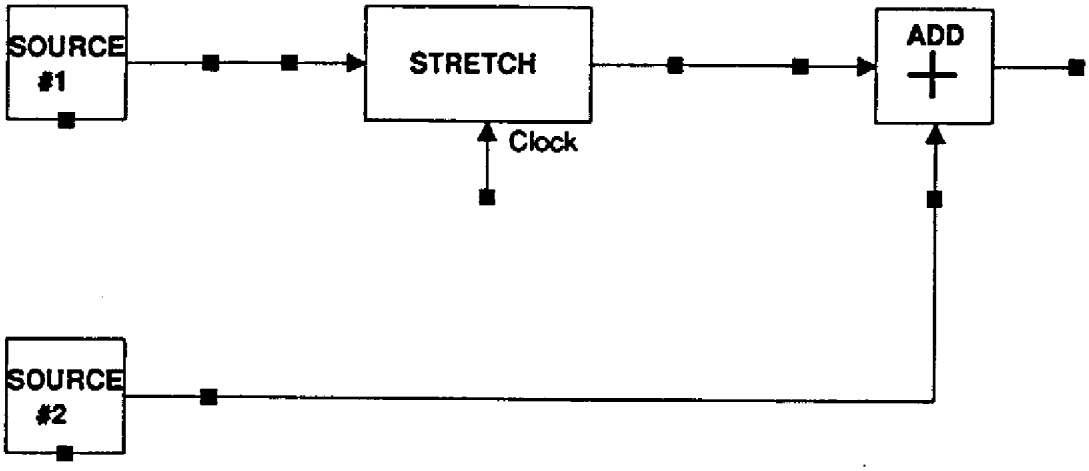

Figure 1

Simulation Control in Capsim

The Capsim program has a relatively simple paradigm for controlling the order of the simulation. This subject is covered in greater depth elsewhere in this manual but for the purposes of avoiding buffer overflow the process can be thought of as follows: Starting at the Universe level, the Capsim program determines a "visitation order" for all of the Stars or Galaxies at that level. For the present purpose, this ordering can be assumed to be random. Once the visitation order is established it remains fixed until the simulation completes. Each Star or Galaxy is visited, in the visitation order, to process the samples on its input buffer. When a Star is visited (called), control is passed to that Star's program

which then runs until it terminates (returns). Most Stars will process as much data as possible from the input buffers before returning control to Capsim. For a single input Star, the entire buffer will usually be processed. Where there are multiple inputs, at least one input buffer would be exhausted. Stars can always be written such that they stop short of exhausting an input buffer. This is done in some special cases but it is to be avoided in general.

When a Galaxy is visited, its component Stars and Galaxies are ordered and visited in turn, as described above, until no component of the Galaxy has any data in its input buffers that it can process. This means that the Galaxy itself has processed as much data as possible from its input buffers. Since the processing that takes place at the Galaxy level is under control of the Capsim program, there are no exceptions to this operation as there can be with Stars. Capsim then moves to the next higher Galaxy and continues visiting components there. This process continues in the initial fixed pattern until no Star or Galaxy is found that can process data from an input buffer. The run is deemed to be finished at this point and Capsim returns control to First tool.

Controls Provided to Control Data Flow

Capsim provides three ways in which data flow can be controlled. These are sufficient in most, if not all, cases to prevent buffer overflow, even when an arbitrarily large number of samples is to be processed. Flow control can be effected by: 1) Limiting the number of samples output by a Star on each call, 2) By collecting a select group of Stars into a Galaxy, and 3) By Pacing a source star's output. Methods 1 and 3 are effected by the Star code while method 2 is a part of Capsim itself.

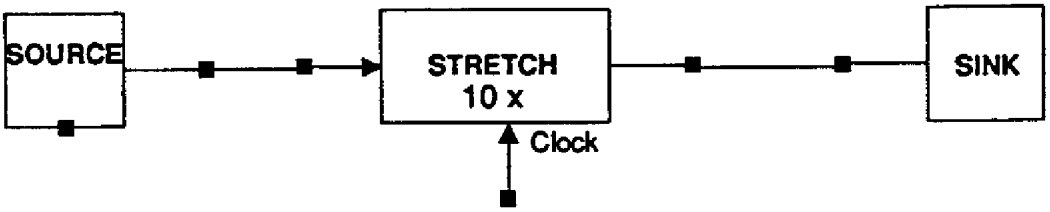

Figure 2

Consider the simple case of Fig. 2 in which a source Star provides input to a "stretch" Star which increases the number of samples by 10. If we set the source for 10,000 total samples we have a problem. The first time the source Star is called, it will output 10,000 samples and when "stretch" is subsequently called it will read these in and produce 100,000 output samples. In most cases this will exceed the maximum buffer size and Capsim will terminate and produce an error message. This case is mitigated by limiting the number of samples a source star can output in any one call. The Capsim Star design guidelines advise that sources should limit the number of samples they output on each call (when Pacing is not

being used). The suggested limit is NOSAMPLES, a global constant which can be set by the system manager to any integer. Normally NOSAMPLES = 128. All of the source Stars provided with Capsim observe this limitation. The case of Fig. 2 is not a problem for source stars which follow the guideline because the source will output NOSAMPLES, "stretch" will output 10 x NOSAMPLES and "sink" will empty its input buffer, on each pass through the simulation. Of course, if the stretch parameter is very large, "stretch" could exceed the maximum buffer size on a single call. For this reason, the Star design guideline recommends that all Stars that can have a large number of output samples for each input sample be limited to NOSAMPLES output per call.

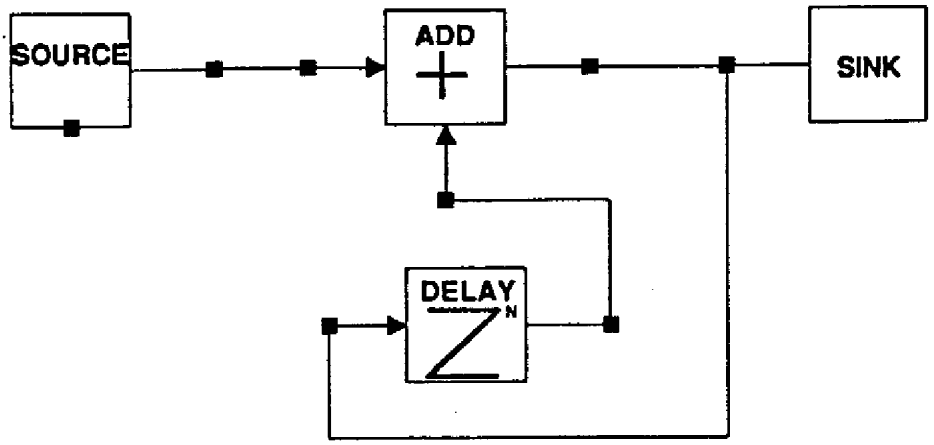

Figure 3

In the case of Fig. 3, additional control is required. Here a source star feeds one input of an "add" star which is part of a feedback loop. On the first pass through the simulation the source would output 128 (assuming NOSAMPLES = 128) samples and the "delay" would output 1 sample, i.e. 0.0 (this is the way delay Stars work). On the next pass the source Star would output another 128 samples and the "add" star would read in one sample (because there is only one sample on its second input) and output one sample. The delay Star would output one sample. Eventually the input buffer (from the source) on the "add" Star will overflow. Overflow can be avoided by using method 2). If the feedback loop is incorporated into a Galaxy, as shown in Fig 4., the operation is much different. When the source is visited it outputs 128 samples. Then Capsim goes into the Galaxy and stays there until all 128 samples on the Galaxy input buffer are processed, by alternately visiting the "add" and "delay" Stars. This not only prevents buffer over flow but it speeds up the simulation and minimizes buffer memory requirements. On the down-side it uses hierarchy to solve a simulation problem whereas we would like to reserve hierarchy for other purposes i.e. to hide complexity, create new objects, and group elements as they will be implemented.

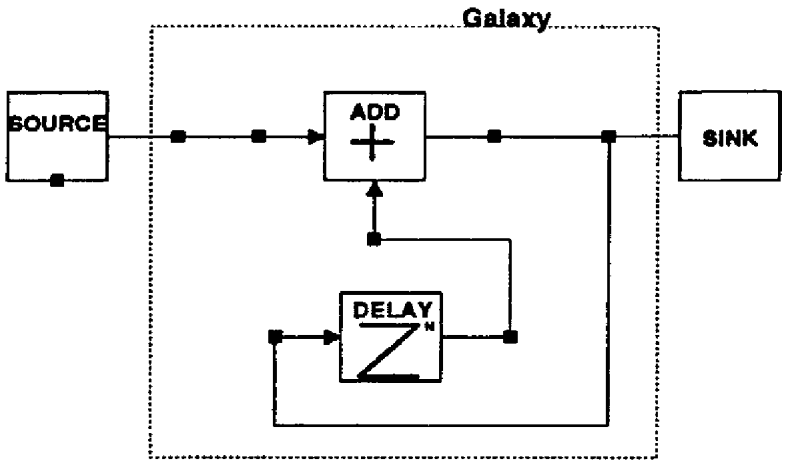

Figure 4

Not all data flow problems can be solved by the Galaxy approach together with a limitation of source output to NOSAMPLES/call. Fig 5 is a simple example of a simulation that requires additional controls to prevent buffer overflow. A source, limited to 128 samples/call, is followed by a "stretch" Star that stretches by 128. If the "stretch" star is allowed to empty its buffer on each call, then its output buffer will overflow. If it is limited to 128 samples/call then its input buffer will overflow as the number of samples becomes large. To provide a means to deal with situations of this type and to eliminate the necessity of forming Galaxies where they are not desirable for other reasons, a method of Pacing source Stars has been provided.

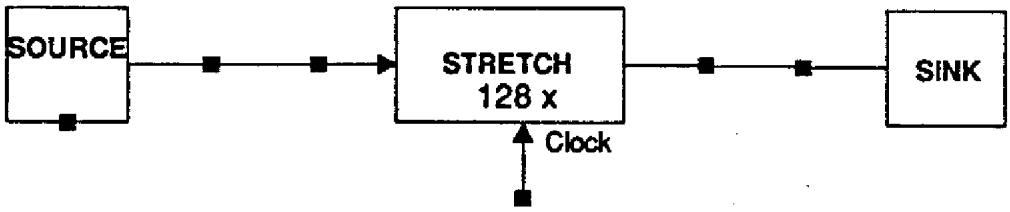

Figure 5

The Capsim Star design guidelines call for each source star to have a Pacing input which can be used or ignored by the user. If the Pacing input is left unconnected the source Star produces a maximum number of samples determined by the parameter "num_of_samples" and then quits. The number of samples that can be output per call is limited to NOSAMPLES.

If the Pacing input is connected, two additional control parameters can be used, i.e. "samples_first_time" and "pace_rate". The first time the Star is called to output samples, it outputs a number equal to the parameter "samples_first_time". On the first, and each

subsequent call, the star empties its Pace input buffer and keeps a running total of how many samples have been removed. At each call, after the first, the star outputs as many samples as possible without the running total of the samples output exceeding either of the following:

- (Total samples seen on the Pace input to that point) x (pace_rate)

  rounded to the nearest integer + samples_first_time.

- num_of_samples

The number of samples on any single call is still limited to NOSAMPLES.

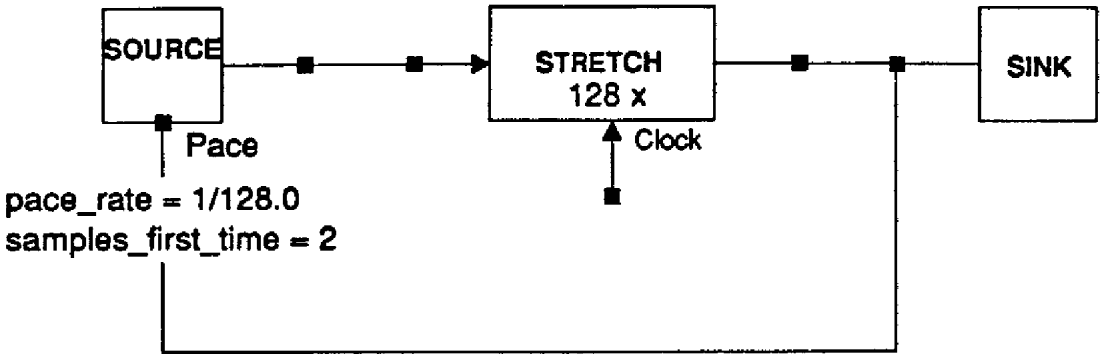

Figure 6

In the simulation of Fig. 6 the action will be as follows: On the first pass the source will output two samples. The "stretch" star will input a sample and output 128 samples (assuming NOSAMPLES = 128) and the "sink" will empty its input buffer. On the next pass the source will empty its Pace buffer and determine that it has seen 128 samples there. It then calculates its total output target as 128/128.0 + 2 = 3 allowing it to output another sample. The "stretch" star will read in another sample and output 128. On the third pass the source Star will empty it's Pace buffer and calculate it's output target as 256/128.0 + 2 = 4, allowing it to output another sample. The process will repeat until the total number of samples output by the source equals num_of_samples. The source will stop outputting samples at that point but it will continue to empty its Pace buffer to avoid any possible overflow there. The simulation will continue until both "stretch" and "sink" have emptied their input buffers. By using pacing in this way, a simulation of any length can be run and the maximum buffer size will never be greater than 128 samples.

Another feature of source pacing is illustrated in Fig 7. If the Pace input is connected and the num_of_samples parameter is given a negative value, then the number of samples the source Star will output is unlimited. In Fig 7 the "sine" Star outputs a number of samples = samples_first_time the first time it is called. After that, it will match it's output to the number

of samples on the "mixer" Star's #1 input thus assuring that the "mixer" gets as many samples as it needs from the "sine" Star no matter how long the simulation runs. This is often convenient for ancillary source Stars, such as local oscillators, noise generators, clocks, etc. It is usually undesirable to have these sources be the limitation on the number of samples to be run. For most simulations, it will be found convenient to pace all of the source stars but one and set their num_samples parameter to -1. The remaining source, usually the primary data source, will have num_of_samples set to a positive value and it will control the length of the simulation. There must, of course, be some limitation on the number of samples or the simulation will never terminate.

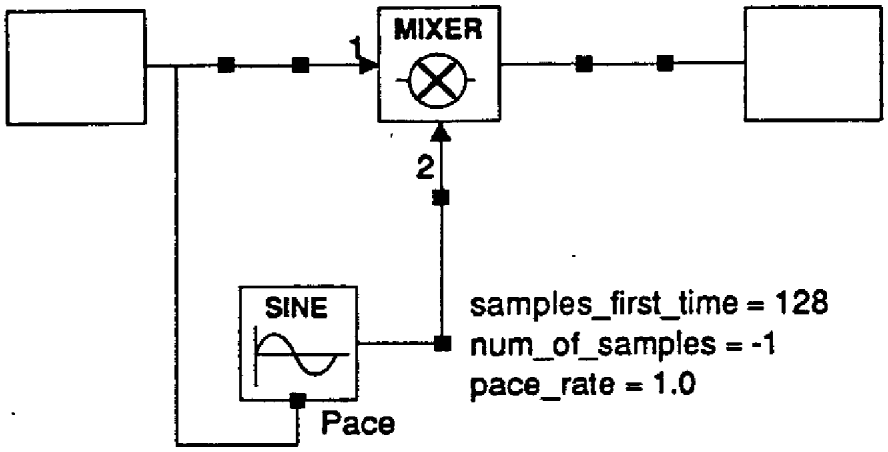

Figure 7

It is possible to set up configurations that cannot be stabilized by the methods provided. The network illustrated in Fig. 8, for example, will overflow the buffer on the #1 input to the "add" Star if the number of samples to be run is large enough.

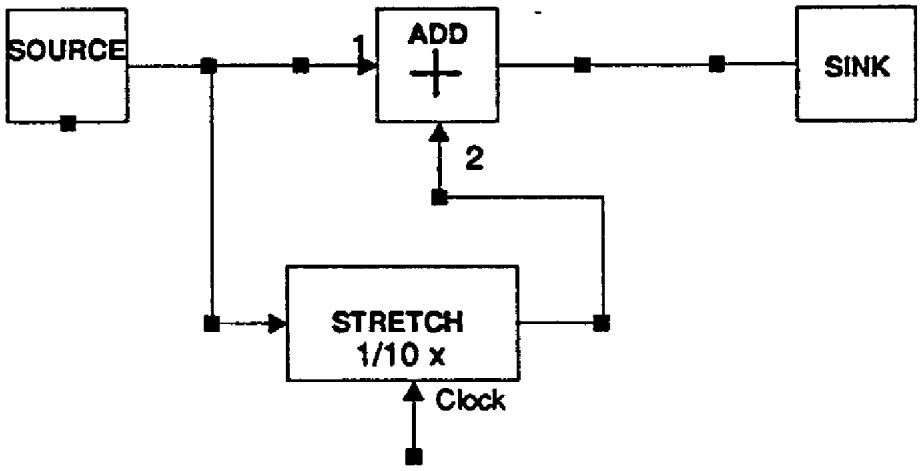

Figure 8

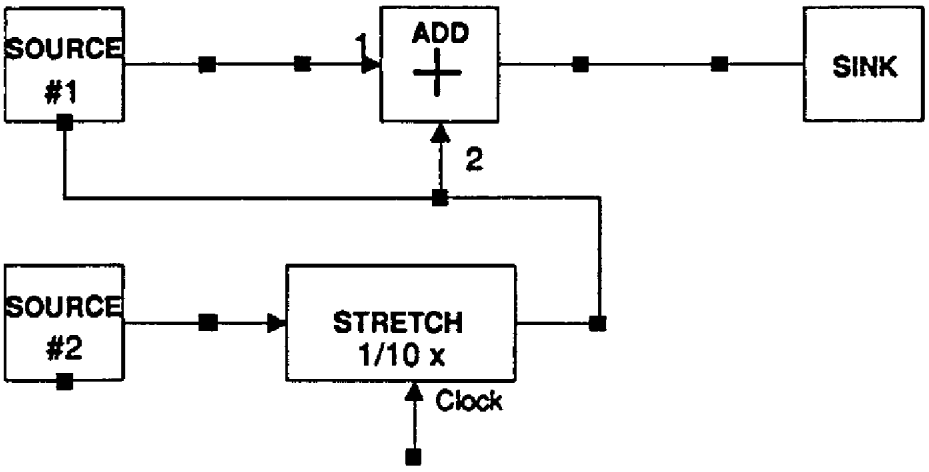

Figure 9

In most, if not all, of these cases it is possible to reconfigure the network so that the desired simulation can be run. Fig. 9 illustrates one way to reconfigure the network of Fig. 8. Two instances of the source are used. If they are given the same parameter values, except for the control parameters, they will produce the same stream of sample values. By pacing source 1, however, we keep it from producing an excessive number of samples on the "add" Star input.