Caelan Mayberry & Justin Sybrandt

Design Decisions

The first design decision was the construction of the Process structure. This was a simple but still significant choice. A Process contains the information loaded from the file, the arrival and burst times, but it also has storage for the start, finish, and time remaining to completion. When a Process is calculated it stores the start and finish time, so that it can have its wait and turnaround time calculated later. Time remaining is used while the Process is being calculated to determine if it is completed or not. This choice allows us to return the newly calculated data within the Process affected.

Every scheduling type is a function call. Each call takes the array of processes and the number of processes available. FCFS – single does the calculations quickly and easily based upon arrival time.

In the FCFS – per core, we chose to create pseudo-queues from the original list of processes. The function randomizes the array to mimic the random assignment to a core and passes a position within the original array and the length for how long it should go in to array to each ‘core’, which is actually a call to FCFS – single. This provides the same effect and makes it easier to do the calculations.

RR – per core uses a similar method to FCFS – per core, that is it randomly sorts the array and hands the segments allotted to a RR – single function. The single function acts like a single core.

RR – load is a different design than the previous scheduling. While all of the others could be pre-assigned to a core, load would require more calculations to pre-assign because of the need to balance each core. With this in mind we wrote RR – load to do a second by second until the calculations are complete.

Once completed the main function then can look at each processes completed times. Our design choice of storing all relevant data to a Process made writing the output easy. Using the new data the program can run the calculations for turnaround and wait time for each process. It will then write all of the necessary information to a text file.