CS 3243 / 5243 Operating Systems
Phase II

You will need to get the jobs from RAM to the ReadyQueue via the STS (short term scheduler). The ReadyQueue should then dispatch jobs to the next available CPU. Make sure that you are processing the commands on the CPU, moving jobs to the wait or I/O queues as necessary, and remembering to do the context switching. The initial values on the CPU when you start any job are 1,3,5,7, respectively, for Registers A - D, and 9 for the Acc (Accumulator). Process each command and give me the values of each of these registers and accumulator after ALL instructions for that job are completed. When you context switch (move a job from a CPU to either the waiting queue or the I/O queue) that you store the current values of the registers, the accumulator, and the program counter in the PCB. Otherwise, when you switch the job back to running a CPU it will not know where it was.

Here are the commands and how we will implement them. This is a highly modified and simplified instruction set, but still gives us an idea how these commands are processed in the CPU.

add	A	B	10	Take register A, add it to register B, and add this value to ACC.
sub	B	C	5	Subtract register C from register B, add the result to ACC.
mul	C	D	6	Multiply register C by register D and add the result to ACC.
div	B	D	8	Divide register D by register B and add the result to ACC.
_rd	A	B	5	Send the job to the IO queue for 5 cycles.
_wr	C	D	7	Send the job to the IO queue for 7 cycles.
_wt	A	C	4	Send the job to the Wait queue for 4 cycles.
sto	A	B	7	Store the value 7 in the ACC (it replaces the current value)
rcl	C	D	9	Copy the ACC to register C (it replaces the current value)

At the beginning of each job, if it is a new job to the CPU, set the register and accumulator values as instructed. If the job is returning from the I/O or Wait queue, then load the values from the PCB to the appropriate registers and accumulator and set the program counter to the next line to be executed.

The big loop is this: HDD -> LTS -> RAM -> RQ -> CPU -> WAIT Q -> IO Q -> repeat
The loop starts with checking the RAM to see if there is room more for jobs, if so, LTS moves more jobs to the RAM; if the RQ has room, move more jobs to it; if there is a CPU available, move jobs to it; execute one instruction on each CPU (which may move jobs to a queue); decrement the remaining time for all jobs in the queues.

This should show you the way home for Phase II and give you a big headstart for Phase III (add multiple CPU's, track performance).

Please let me know if you have any questions,
Mike Franklin