I have 2 executable E1 and E2 programs corresponding to ER (SR in assignment pdf) and ET(ST) in assignment pdf.

A child process e1 is created and its PID (e1pid) is stored in the local variable. This process registers a signal handler for SIGALRM. Following this creation, the parent main program forks 2 more child processes ER and ET using exec system call. Both these programs accept PID of e1 as an argument. Both of these processes have appropriate clocks setup for this I have used setitmerval() system call.

I am using RDRAND instruction for SIGLARM handler in ER to read in a random number from the CPU, for this I implemented inline assembly in the program. This random number is then enqueued with the SIGTERM and is sent to e1’s SIGTERM handler, which is then printed by it.

A similar SIGALRM handler for ET. The ET process reads CPU timestamp counter, i have used inline assembly to run the rdtsc instruction

to implement this, and then converted it into readable string. Using setitimer() system call i set the interval which raises SIGLARM signal every time the timeout lapses. The random number in this process is obtained using the inline rdtsc instruction in the SIGLARM handler function. This random number is then enqueued with the SIGTERM and is sent to e1’s SIGTERM handler, which is then printed by it.