IT3070E Operating Systems, Autumn 2021, Midterm exam

Instructions

Assume your student id is 20195678. The least significant digit of this number is 8. The second least significant digit is 7. We have 4 integers x_0, x_1, x_2, x_3. Assign to $x_0 =$ the least significant digit of your student id (here $x_0 = 8$). Assign to x_1 the second least significant digit of your student id (here $x_1 = 7$). Assign to x_2 the third least significant digit of your student id (here $x_2 = 6$). Assign to x_3 the fourth least significant digit of your student id (here $x_3 = 5$). Based on the values of x_0 to x_3, compute the values r_0 to r_{13} as indicate below.

- $r_0 = (x_0 \mod 5) + 1$; $r_1 = (x_1 \mod 6) + 1$; $r_2 = ((x_2 \times x_0) \mod 7) + 1$; $r_3 = ((x_3 \times x_1) \mod 8) + 1$;
- $r_4 = ((r_0 + r_2) \mod 9) + 1$;
- $r_5 = ((r_1 + r_3) \mod 9) + 1$;
- $r_6 = ((r_0 + r_1) \mod 9) + 1$;
- $r_7 = ((r_0 + r_1 + r_2) \mod 9) + 1$;
- $r_8 = ((r_1 + r_2) \mod 9) + 1$;
- $r_9 = ((2r_0) \mod 9) + 1$;
- $r_{10} = ((4r_1) \mod 8) + 1$;
- $r_{11} = ((3r_2) \mod 7) + 1$;
- $r_{12} = ((3r_3) \mod 7) + 1$;
- $r_{13} = ((5r_4) \mod 7) + 1$;

Question 1: Processes (20 pts)

Including the initial parent process, how many processes are created by the program below?

```
int main()
int i;
int k = r0
pid_t pid;
for (i=0;i < k;i++)
   pid = fork();
return 0;
```

Question 2: Round Robin preemptive priority scheduling (30 pts)

The following processes are being scheduled using a preemptive, round robin scheduling algorithm

Process	Priority	Burst time	Arrival
P_1	40		0
P_2	30		25
P_3	30		30
P_4	35		60
P_5	5		100

- The column Priority represents the priority of each process, the higher the number, the higher the priority.
- The column burst time is computed based on your student id number. $P_1 = (r_0 \times 3) + r_2$, $P_2 = (r_3 \times 3) + r_5$, $P_3 = (r_6 \times 3) + r_8$, $P_4 = (r_9 \times 3) + r_{11}$ and $P_5 = (r_{12} \times 3) + r_{13}$.
  - **Important: write on your answer sheet the burst time you have calculated for each of the 5 processes.**
- The column Arrival indicates when each process first arrived in the ready queue.
- The round robin quantum time is 10. If a process is preempted by a higher-priority process, the preempted process is placed at the end (the back) of the ready queue.

- You may assumed the CPU executes a process called P_{idle} with priority 0 during intervals where none of the 5 above processes is ready to execute.

1. What is the turnaround time of each process (15pts)

	P_1	P_2	P_3	P_4	P_5
Ttime					

2. What is the waiting time of each process (15pts)

	P_1	P_2	P_3	P_4	P_5
Wtime					

Question 3: Memory management(30pts)

A system implements a paged virtual address space for each process using a one-level page table. Each page is $1024 = 2^{10}$ bytes. We have a $8\text{KB} = 8192 = 2^{13}$ bytes process named *hust* to load into main memory. We have a 2^{16} bytes physical memory, that is 64 frames of 1024 bytes. Let assume the frames are numbered from 0 to 63, frame 0 starts at physical address 0. Frames occupy consecutive addresses in physical memory, that is frame 1 starts at address 1024, frame 2 at 2048, etc. Let $F_0 = r_4 + r_5 + r_6 + r_7$. Assume the first page of *hust* (page 0) is mapped in the page table to frame F_0. (2pts each sub-question, except Q1.13 which is 4 pts)

1. Let $F_2 = r_6 + r_7$. Assume the page table is stored in the beginning of frame F_2. What is the base address stored in the page table base register (PTBR)?
2. Assuming each entry of the page table stores the physical address of the frame where the corresponding logical page is stored. What is the size of one entry in the page table?
3. Which value is stored in the page-table length register (PTLR) for process *hust*?
4. What is the range of physical addresses process *hust* can legally access?
5. What is the corresponding physical address of the first logical address loaded in the program counter register?
6. What is the physical address of logical address $(r_8 \times r_9)$?
7. If the size of the physical memory was $2^{6\times r_8}$, how many bits there will be in an address send to main memory?
8. If the size of a process is $2^{4\times r_{11}}$ and page size is 2^{10}, how many pages there will be into the page table?
9. Size of process = $2^{4\times r_{11}}$, page size = 2^{20}, physical memory = 2^{64}, which bits of a logical address are used to index into the page table?
10. What is the range of the logical addresses of process *hust*?
11. Which address is stored in the CPU base register for process *hust*?
12. What value is stored in the limit register for process *hust*?
13. There is no TLB, thus every data/instruction access requires two memory accesses, one for the page table and one to reach the physical address of the data/instruction. Assume the program counter register is loaded with address 2063 (bin = 0100000001111)
    (a) What is the physical address of the page table entry for this reference?
    (b) What physical address corresponds to logical address 2063?

14. Let the value of F_1 be equal to the physical address of frame F_0. Find the logical address of physical address $F_1 + 3555 + (3 \times F_0)$

Question 4: Multilevel page table (20pts)

Given a process of 2^{16} bytes, physical memory is 2^{32}, and frame size is 1KB = 2^{10} bytes. Thus physical addresses are on 32 bits. Although only 16 bits are needed to address any byte in the process, logical addresses are still represented on 32 bits in the program counter register of this system. Assume the OS runs a 2 levels page table.

1. How many frames are used to store the page table for this 2^{16} bytes process? (15 pts, explain your answer, such as how many bits are used to index into a frame storing the process, how bits are used to index into a frame of the inner level page table, how many bits are used to index into a frame of the outer level page table, are they bits unused, etc, otherwise 0 point)
2. What will be the number of frames used by the page table for this process if the page table system was on a single level? (5 pts, explain your answer otherwise 0 point)