COMP1411 (Spring 2023) Introduction to Computer Systems

Individual Assignment 3 Duration: 00:00, 01-Apr-2023 ~ 23:59, 02-Apr-2023

Name	**WANG Ruijie**
Student number	**22103808d**

Question 1. [4 marks]

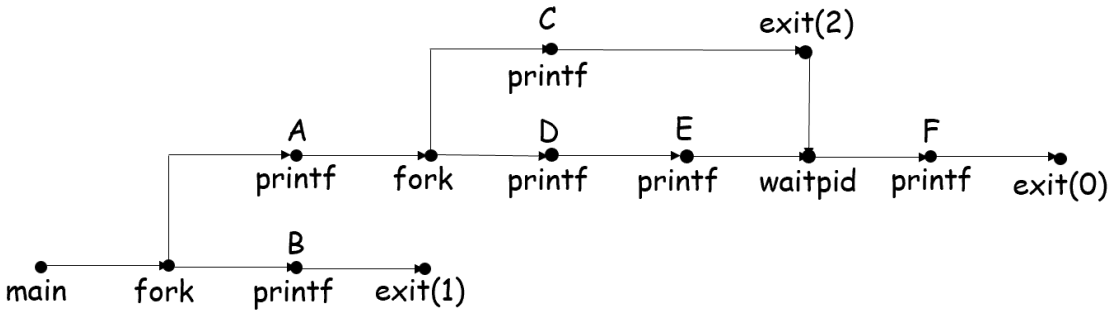
Consider the execution of the following function (written in the C language).

```
// all the needed headers are included
int main()
{
    int pid;
    if (fork() == 0){
        printf("A"); fflush(stdout);
    }
    else {
        printf("B"); fflush(stdout);
        exit(1);
    }

    if ((pid = fork()) == 0){
        printf("C"); fflush(stdout);
        exit(2);
    }

    printf("D"); fflush(stdout);

    printf("E"); fflush(stdout);

    waitpid(pid, NULL, 0);

    printf("F"); fflush(stdout);
    exit(0);
}
```

1(a) Draw the process graph for the execution of "main()".

In a process graph, each function, including main(), fork(), printf(), waitpid(), and exit(), is represented by a vertex. You can omit the fflush() function in the process graph. For each vertex, please write the function name below the vertex and for printf() write the output character above the vertex. Each edge must be directed, with the direction representing the happen before relationship.

Answer:



1(b) List all the feasible output of the program.

For example, if there are two feasible outputs, please given the answer as follows (listing only one feasible output in each line and number them):

(1) A B C D E F
(2) E D C B A F

Note that the above two outputs are only used to demonstrate the format to give your answers for question 1(b), they do not have any indication to the correct answers.

Answer:

(1)	B A C D E F	(10)	A C D B E F
(2)	B A D C E F	(11)	A D C B E F
(3)	B A D E C F	(12)	A D E B C F
(4)	A B C D E F	(13)	A C D E B F
(5)	A B D C E F	(14)	A D C E B F
(6)	A B D E C F	(15)	A D E C B F
(7)	A C B D E F	(16)	A C D E F B
(8)	A D B C E F	(17)	A D E C F B
(9)	A D B E C F	(18)	A D C E F B

Question 2 [3 marks]

Assume the system has a cache between the CPU and the main memory. Each time the CPU uses some data item, the CPU will always go the cache to fetch the data item; if the data item is found in the cache, then there is a cache hit, and the data item will be loaded to CPU; otherwise if the data item is not found in the cache, then the item will be loaded from the main memory and at the same time put into the cache (we assume that all the access data are guaranteed to be stored in the main memory).

Each **cache block** has the size of **16B** (B = bytes), and **the cache has 3 blocks**, which means the **total size** of the cache is **48B**. If one main memory address is accessed, the whole block containing the accessed address will be loaded into the cache.

We assume that the mapping from an address to a block number is given below:

Address range	Block number
0x00 – 0x0F	7
0x10 – 0x1F	6
0x20 – 0x2F	5
0x30 – 0x3F	4
0x40 – 0x4F	3
0x50 – 0x5F	2
0x60 – 0x6F	1
0x70 – 0x7F	0

The cache is managed with the **LRU** replacement policy: when all the 3 cache blocks are used up and a new data block will be loaded into cache, one data block out of the 3 blocks will be replaced out of the cache. The data block that was **least recently used** will be replaced. The cache is **empty** at the beginning.

12 main memory addresses are accessed sequentially, listed in Figure 2.

Please fill in Figure 2: input the corresponding data block number for each accessed address into the cache states represented by vertical boxes, and input whether the access to the address is cache hit or miss by filling the brackets below the address, with “M” for cache miss and “H” for cache hit.

For example, starting from empty cache, access address 0x5A and 0x64, the red contents are those you are supposed to fill in, as shown in Figure 1.

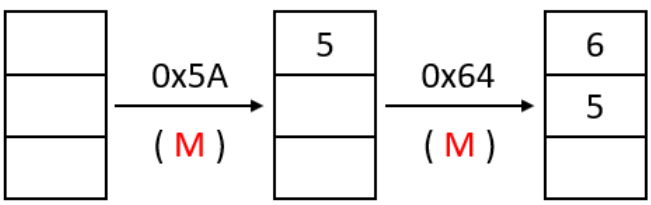
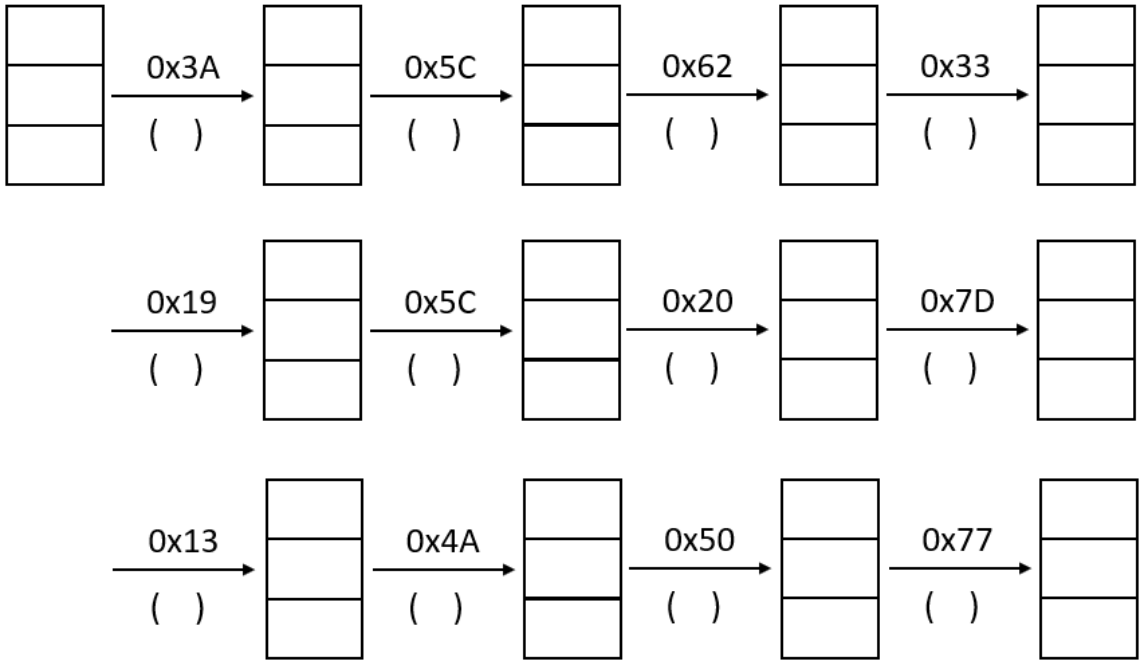
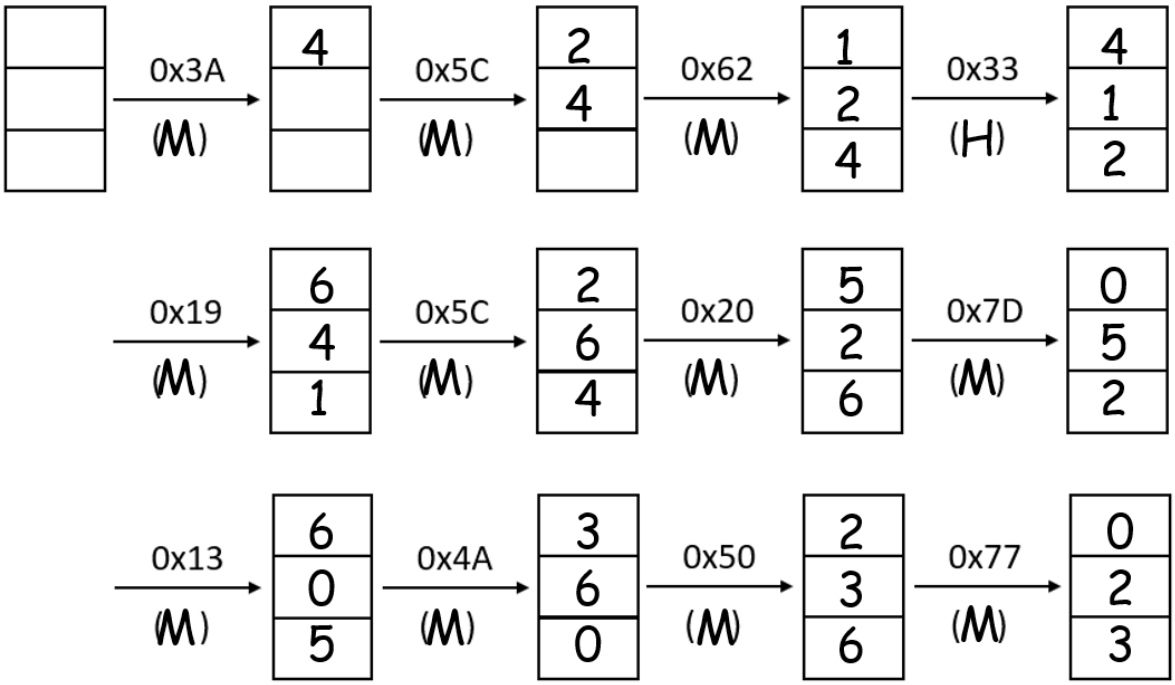


Figure 1

Figure 2



Answer:



Question 3. [3 marks]

Considering a virtual memory system with paging:

Assume that the **page size** is **2KB (B = bytes)**, and each **physical and virtual address** is represented by a **16-bit binary** or equivalently a **4-digit hex-decimal** number.

A process was assigned **3 physical pages/frames** with frame number 2, 4, 8. This means that no matter how many virtual pages the process has, all its virtual pages must be mapped into only the 3 given frames.

Now it is **time 200** (a larger time value indicates a later point in time). The current content of the page table is shown below with Table 1, in which the "**Install time**" column shows the **time when the virtual page is installed** in the corresponding row (e.g, 160 indicates that virtual page 4 is installed at time 160).

Page numbers are given in the **decimal format**. The "-" symbol in the virtual page column indicates that no virtual page has been mapped to the frame that are listed in the same row with the virtual page. The "-" symbol in the "Install time" column indicates that this physical page is not installed yet.

Assume that **FIFO replacement** is used for page replacement. This means, if all the 3 physical pages have been mapped with 3 virtual pages, and a 4th different virtual page is accessed, among the 3 mapped virtual pages, the **earliest installed item** will be replaced by the 4th new virtual page.

From now on, the following virtual addresses will be accessed sequentially:

0x2564, **0x1FFF**, **0x5CD8**, **0x4337** at time **220**, **250**, **280** and **300** respectively.

Please show the page table for each data accesses, by filling in the blanks of Table 2. Note that you are required to input the page numbers in decimal format.

Table 1: The original page table

Virtual page number	Physical page number	Install time
4	2	160
8	4	100
-	8	-

Answer:

Table 2: The page table after the four accesses

(1)

Virtual page number	Physical page number	Install time
4	2	160
8	4	100
4	8	220

(2)

Virtual page number	Physical page number	Install time
4	2	160
3 8	4	250 100
4 3	8	220 250

(3)

Virtual page number	Physical page number	Install time
11 9	2	280 160
3 11	4	250 280
4 3	8	220 250

(4)

Virtual page number	Physical page number	Install time
11 8	2	280 300
3 11	4	250 280
8 3	8	300 250