

# CSC3150 Assignment3 Report

## Introduction

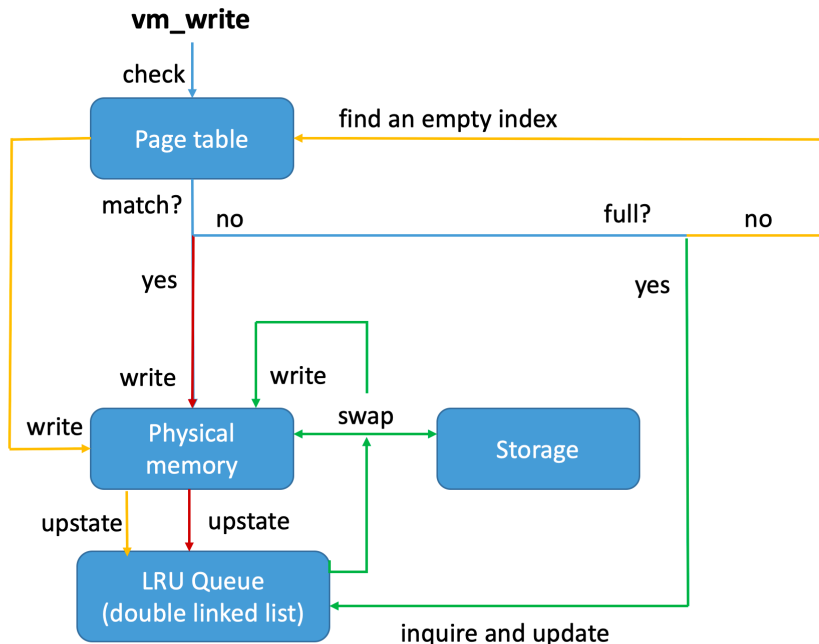
In this assignment, a simple virtual memory simulation is implemented in a kernel function of GPU that has single thread, limit shared memory and global memory. The source code of this virtual memory implementation is attached with this report, which is in the file "virtual\_memory.cu".

Specification of the GPU Virtual Memory:

- Secondary memory (global memory) - 128KB
- Physical memory (shared memory) - 48KB (32KB for data buffer and 16KB for page table setting)
- Page size - 32 bytes
- Page table entries - 1024 (32KB / 32 bytes)
- Page replacement policy for page fault: **LRU** (Least Recently Used)

## Program design

vm\_write



The above logic graph shows the basic logic of the implementation of `vm_write`.

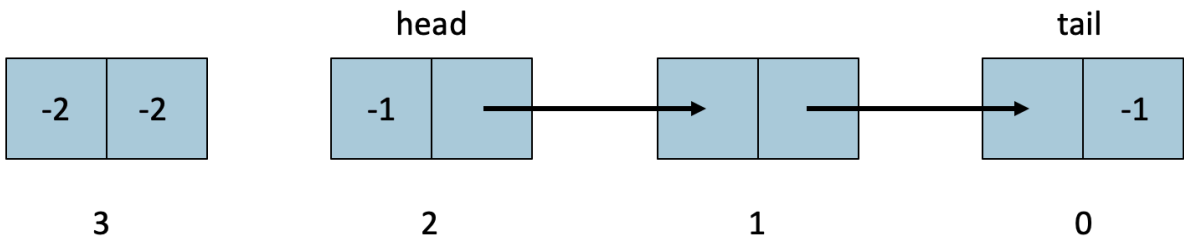
In the graph, the modules page table and physical memory are specified in the introduction. Module Storage relates to Secondary memory. What needs to be emphasized is that, the module Queue is a data structure used for implementing LRU. It uses double linked list to construct the queue.

There are 16KB memory space preset for page table setting but only half of it has been used, so there could be extra two integers for every index to record. To implement double linked list, we utilized these two integers - one integer represents the left element (next-accessed index) and the other represents the right element (previous-accessed index). If there is no left element, which means it is the head of the list, then fill with -1; if there is no right element, which means it is the tail of the list, then fill with -1 as well. All unused index were initialized with left = -2 and right = -2.

For example, if we access index 0, 1, 2 in order at beginning, then the corresponding record in index 0, 1, 2, 3 is as following:

| index | 0  | 1 | 2  | 3  |
|-------|----|---|----|----|
| left  | 1  | 2 | -1 | -2 |
| right | -1 | 0 | 1  | -2 |

Visualize this double linked list (queue):



Based on this setting, we could quickly get the least recent used index and swap it out when page replacements happen. Also, when some data in VM was accessed, we could easily update the queue by just change several numbers. These operations could be found in the code.

After the LRU queue has been built, we could go through the flow of the program.

As the logic graph shows, when *vm\_write* was called, it firstly checked the page table to get the current information of the page table, like whether there is a matching index or empty index, or whether the queue has the head and tail, the index will be recorded if it exists. Four variables were set for this purpose: *target\_index*, *head\_index*, *tail\_index*, *empty\_index* which is initially set as -1 to represent false:

```

int target_index, head_index, tail_index, empty_index;
target_index = empty_index = head_index = tail_index = -1;
int left, right;
for (int i = 0; i < vm->PAGE_ENTRIES; ++i){
    if (target_index < 0 && vm->invert_page_table[i] == base) target_index =
i; // find the corresponding index
    if (head_index < 0 && vm->invert_page_table[i + 2 * vm->PAGE_ENTRIES] ==
-1) head_index = i; // find the most recent index
    if (tail_index < 0 && vm->invert_page_table[i + 3 * vm->PAGE_ENTRIES] ==
-1) tail_index = i; // find the least recent index
    if (empty_index < 0 && vm->invert_page_table[i] == 0x80000000)
empty_index = i; // find the empty index
    if (target_index >= 0 && head_index >= 0) break; // to reduce running
time, break in advance if the necessary indexes have been found
}

```

If there is a matching index, then write the input into the data buffer (physical memory):

```

if (target_index >= 0) {
    // refresh the linked-list based on priority
    left = vm->invert_page_table[target_index + 2 * vm->PAGE_ENTRIES];
    right = vm->invert_page_table[target_index + 3 * vm->PAGE_ENTRIES];
    if (left != -1) { // target_index is not the head_index
        vm->invert_page_table[left + 3 * vm->PAGE_ENTRIES] = right; // left.next =
right
        vm->invert_page_table[target_index + 2 * vm->PAGE_ENTRIES] = -1; //
target.previous = -1
        if (right != -1) // target_index is not the last
            vm->invert_page_table[right + 2 * vm->PAGE_ENTRIES] = left; //
right.previous = left
        vm->invert_page_table[head_index + 2 * vm->PAGE_ENTRIES] = target_index;
// head.previous = target
        vm->invert_page_table[target_index + 3 * vm->PAGE_ENTRIES] = head_index;
// target.next = head
    }
    printf("Read: %d; page faults: %d\n", (int)addr, vm->pagefault_num_ptr[0]);
    return vm->buffer[target_index * vm->PAGESIZE + offset];
}

```

If there is no matching index, increment the counting on page faults:

```

vm->pagefault_num_ptr[0] += 1; // count the page fault

```

If there is empty index:

```

if (empty_index >= 0) { // the page table is not empty

```

```

    vm->invert_page_table[empty_index] = base; // edit the new index of the
page table
    vm->invert_page_table[empty_index + vm->PAGE_ENTRIES] = 1; // change the
status
    vm->buffer[empty_index * vm->PAGESIZE + offset] = value; // write into
buffer
    printf("write: %d; page faults: %d\n", (int)addr, vm-
>pagefault_num_ptr[0]);
    if (head_index >= 0) { // head_index exists
        vm->invert_page_table[head_index + 2 * vm->PAGE_ENTRIES] =
empty_index;
        vm->invert_page_table[empty_index + 2 * vm->PAGE_ENTRIES] = -1;
        vm->invert_page_table[empty_index + 3 * vm->PAGE_ENTRIES] =
head_index;
    }
    else { // it is the first accessed index
        vm->invert_page_table[empty_index + 2 * vm->PAGE_ENTRIES] = -1;
        vm->invert_page_table[empty_index + 3 * vm->PAGE_ENTRIES] = -1;
    }
}

```

Otherwise the page table is full, then do the page replacement on the least recent used index, which is dequeued by the linked list we implemented:

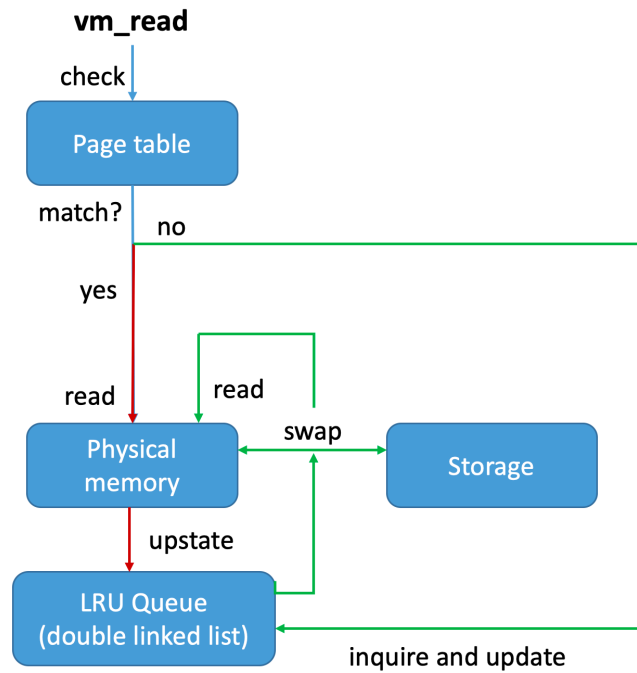
```

else { // the page table is full -- page swap
    int new_tail = vm->invert_page_table[tail_index + 2 * vm->PAGE_ENTRIES];
    // swap out to the storage
    for (int i = 0; i < vm->PAGESIZE; ++ i) {
        vm->storage[vm->invert_page_table[tail_index] * vm->PAGESIZE + i] =
vm->buffer[tail_index * vm->PAGESIZE + i];
    }
    vm->buffer[tail_index * vm->PAGESIZE + offset] = value;
    printf("write: %d; page faults: %d\n", (int)addr, vm-
>pagefault_num_ptr[0]);
    vm->invert_page_table[tail_index] = base; // edit the new index of the
page table
    vm->invert_page_table[new_tail + 3 * vm->PAGE_ENTRIES] = -1; //
new_tail.next = -1
    vm->invert_page_table[tail_index + 2 * vm->PAGE_ENTRIES] = -1; //
become head
    vm->invert_page_table[tail_index + 3 * vm->PAGE_ENTRIES] = head_index;
    vm->invert_page_table[head_index + 2 * vm->PAGE_ENTRIES] = tail_index;
}

```

The above is all the implementation about *vm\_write*.

**vm\_read**



As the logic graph shows, `vm_read` is quite similar to `vm_write`. It does not need to do empty check but in the swap step, it should do both swapping out and swapping in:

```
// swap out
for (int i = 0; i < vm->PAGESIZE; ++ i) {
    vm->storage[vm->invert_page_table[tail_index] * vm->PAGESIZE + i] = vm->
    >buffer[tail_index * vm->PAGESIZE + i];
}
// swap in
for (int i = 0; i < vm->PAGESIZE; ++ i) {
    vm->buffer[tail_index * vm->PAGESIZE + i] = vm->storage[base * vm->
    >PAGESIZE + i];
}
```

And `vm_read` should return the read data at last:

```
return vm->buffer[target_index * vm->PAGESIZE + offset];
```

All the other code of `vm_read` is almost the same as the `vm_write`, since they are somehow like reverse operation for each other. Please refer to source code for details.

## vm\_snapshot

For snapshot, just simply do the `vm_read` in the same order of input:

```

__device__ void vm_snapshot(VirtualMemory *vm, uchar *results, int offset,
                           int input_size) {
    printf("Start of snapshot:\n");
    for (int i = 0; i < input_size; ++i){
        results[i] = vm_read(vm, i); // read from VM in order
        printf("In snapshot - ");
    }
}

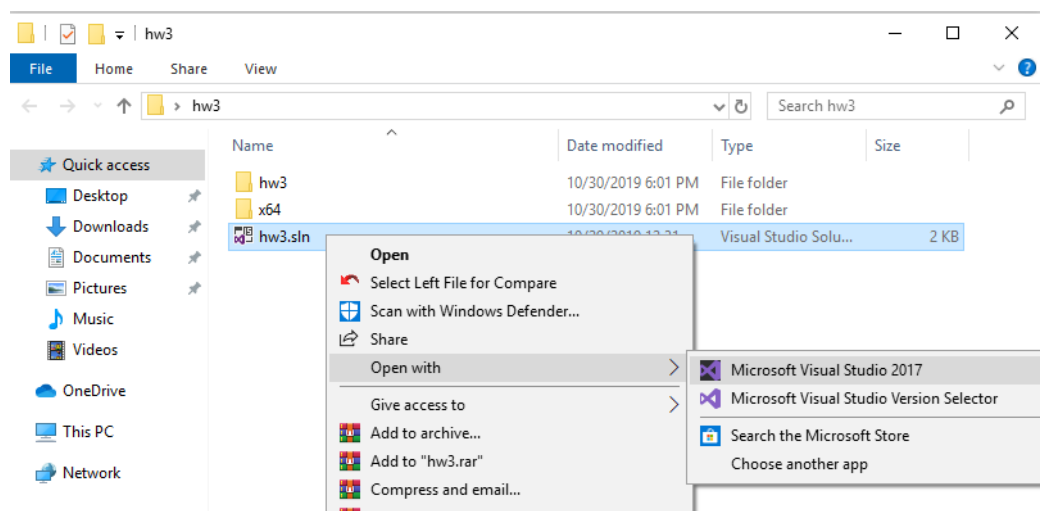
```

## Running environment

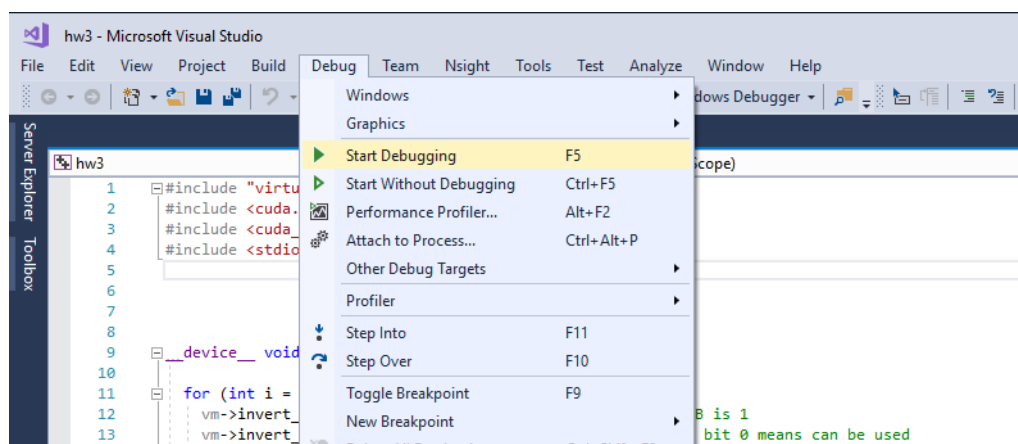
- Windows 10 Enterprise
- Microsoft Visual Studio 2017
- CUDA 9.2
- NVIDIA Geforce GTX 1060 6GB
- Compute capacity: 6.1

## Steps to run the program

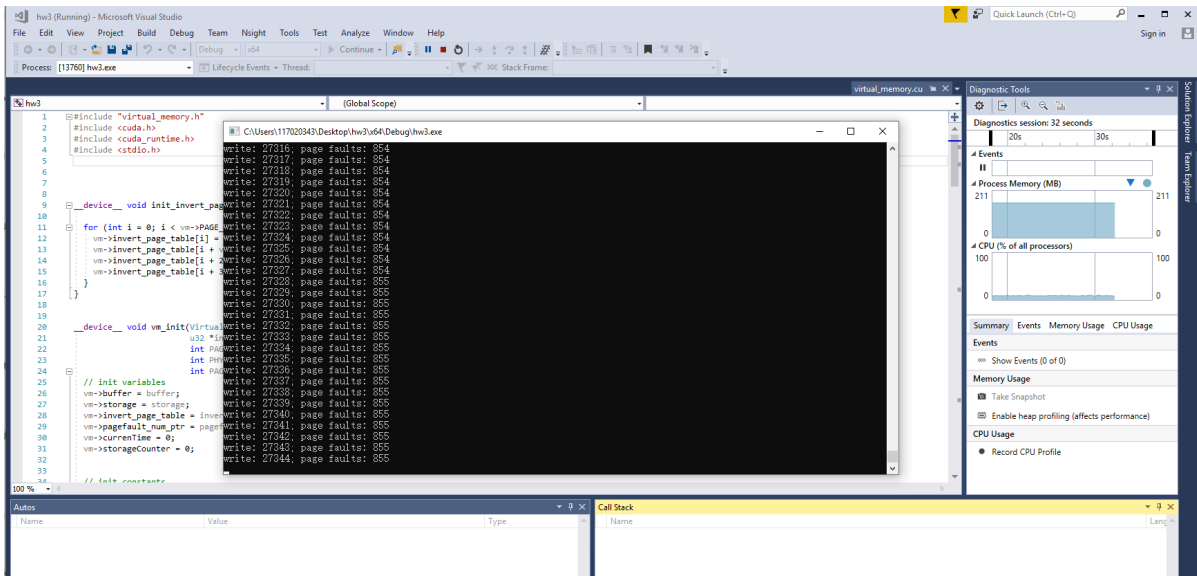
1. In the folder of 'hw3', open 'hw3.sln' with VS 2017:



2. In the VS 2017, select 'Debug' item in the menu and click 'Start Debugging' (or directly press F5):

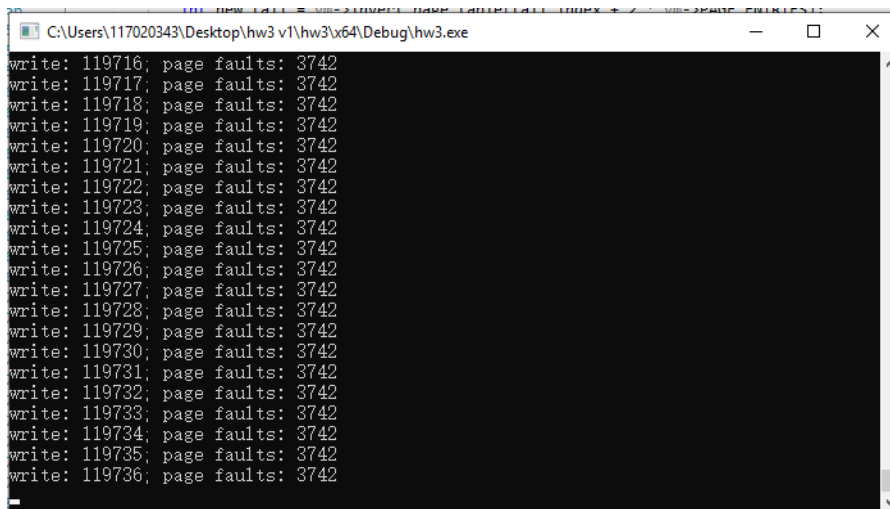


Then the program will start running:

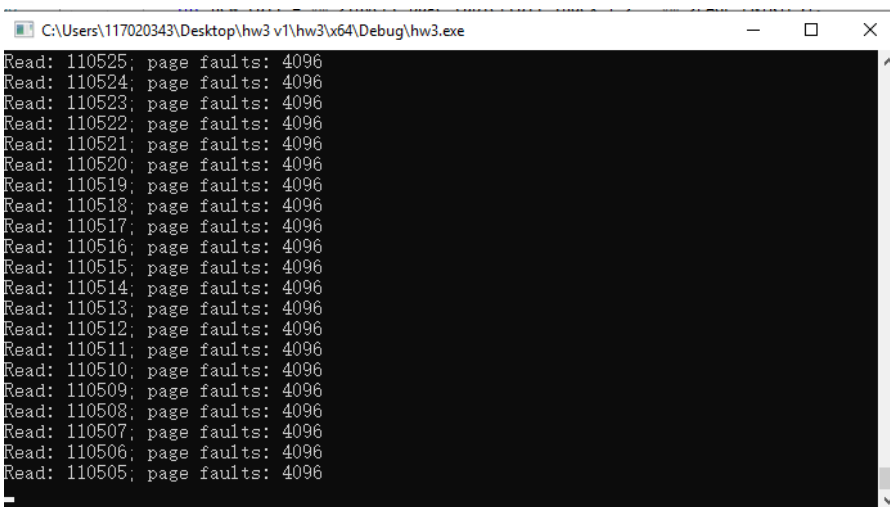


## Sample output

vm\_write (part):



vm\_read (part):



vm\_snapshot (part):

```
C:\Users\117020343\Desktop\hw3 v1\hw3\hw3\Debug\hw3.exe

In snapshot - Read: 27034; page faults: 4942
In snapshot - Read: 27035; page faults: 4942
In snapshot - Read: 27036; page faults: 4942
In snapshot - Read: 27037; page faults: 4942
In snapshot - Read: 27038; page faults: 4942
In snapshot - Read: 27039; page faults: 4942
In snapshot - Read: 27040; page faults: 4943
In snapshot - Read: 27041; page faults: 4943
In snapshot - Read: 27042; page faults: 4943
In snapshot - Read: 27043; page faults: 4943
In snapshot - Read: 27044; page faults: 4943
In snapshot - Read: 27045; page faults: 4943
In snapshot - Read: 27046; page faults: 4943
In snapshot - Read: 27047; page faults: 4943
In snapshot - Read: 27048; page faults: 4943
In snapshot - Read: 27049; page faults: 4943
In snapshot - Read: 27050; page faults: 4943
In snapshot - Read: 27051; page faults: 4943
In snapshot - Read: 27052; page faults: 4943
In snapshot - Read: 27053; page faults: 4943
In snapshot -
```

Final output in console:

```
Microsoft Visual Studio Debug Console

In snapshot - Read: 131059; page faults: 8193
In snapshot - Read: 131060; page faults: 8193
In snapshot - Read: 131061; page faults: 8193
In snapshot - Read: 131062; page faults: 8193
In snapshot - Read: 131063; page faults: 8193
In snapshot - Read: 131064; page faults: 8193
In snapshot - Read: 131065; page faults: 8193
In snapshot - Read: 131066; page faults: 8193
In snapshot - Read: 131067; page faults: 8193
In snapshot - Read: 131068; page faults: 8193
In snapshot - Read: 131069; page faults: 8193
In snapshot - Read: 131070; page faults: 8193
In snapshot - Read: 131071; page faults: 8193
In snapshot - pagefault number is 8193

C:\Users\117020343\Desktop\hw3 v1\hw3\hw3\Debug\hw3.exe (process 9664) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .
```

Output in file 'snapshot.bin', compared with input file 'data.bin' (part):

| user_program.cu  | data.bin | virtual_memory.cu | snapshot.bin   |
|--|----------|-------------------|--|
| 00000000 37 0C 60 51 38 5A 25 0B 13 2A 2A 01 0C 07 05 2F 7. Q8ZK.***.../   |          |                   | 00000000 37 0C 60 51 38 5A 25 0B 13 2A 2A 01 0C 07 05 2F 7. Q8ZK.***.../   |
| 00000010 62 0B 3A 2B 3E 35 2B 3C 0E 27 60 3B 04 53 2A 3A b...+54<.....S*   |          |                   | 00000010 62 0B 3A 2B 3E 35 2B 3C 0E 27 60 3B 04 53 2A 3A b...+54<.....S*   |
| 00000020 5E 5A 5B 31 1F 1B 0B 01 14 04 01 20 0B 05 1E 08 Z[1.....          |          |                   | 00000020 5E 5A 5B 31 1F 1B 0B 01 14 04 01 20 0B 05 1E 08 Z[1.....          |
| 00000030 44 57 02 1D 5C 61 28 05 23 58 3F 5A 46 39 63 3F DW..a(.#X?ZF9c?   |          |                   | 00000030 44 57 02 1D 5C 61 28 05 23 58 3F 5A 46 39 63 3F DW..a(.#X?ZF9c?   |
| 00000040 62 59 3F 1C 44 49 1C 57 1C 51 46 5A 55 63 32 34 bY?.DI.W.QFZUc24  |          |                   | 00000040 62 59 3F 1C 44 49 1C 57 1C 51 46 5A 55 63 32 34 bY?.DI.W.QFZUc24  |
| 00000050 56 33 21 1D 2F 18 21 21 0B 30 16 50 38 15 2A 35 V31./..0.P8.*5    |          |                   | 00000050 56 33 21 1D 2F 18 21 21 0B 30 16 50 38 15 2A 35 V31./..0.P8.*5    |
| 00000060 3D 38 20 1C 50 3B 43 08 5B 24 31 4C 23 62 4F 48 =8.P.C.[!L#b0H    |          |                   | 00000060 3D 38 20 1C 50 3B 43 08 5B 24 31 4C 23 62 4F 48 =8.P.C.[!L#b0H    |
| 00000070 01 3F 64 63 57 54 20 31 1F 05 1D 56 4D 16 5A 26 .?dWT1...VM.Lz    |          |                   | 00000070 01 3F 64 63 57 54 20 31 1F 05 1D 56 4D 16 5A 26 .?dWT1...VM.Lz    |
| 00000080 1E 15 11 09 20 53 44 16 13 11 31 05 0E 1C 4C 0E ...SD...1...L     |          |                   | 00000080 1E 15 11 09 20 53 44 16 13 11 31 05 0E 1C 4C 0E ...SD...1...L     |
| 00000090 2A 1B 41 50 0A 60 1D 5D 34 39 1E 51 1E 14 46 3B *.AP..]49.Q.F.    |          |                   | 00000090 2A 1B 41 50 0A 60 1D 5D 34 39 1E 51 1E 14 46 3B *.AP..]49.Q.F.    |
| 000000a0 5C 56 44 4B 45 57 61 27 03 61 2B 45 4C 46 22 12 .VDKEwa'.a+ELF"   |          |                   | 000000a0 5C 56 44 4B 45 57 61 27 03 61 2B 45 4C 46 22 12 .VDKEwa'.a+ELF"   |
| 000000b0 60 62 61 39 2D 19 31 61 21 4F 4D 3F 32 62 49 29 `ba9-.1a!OM?2bI)  |          |                   | 000000b0 60 62 61 39 2D 19 31 61 21 4F 4D 3F 32 62 49 29 `ba9-.1a!OM?2bI)  |
| 000000c0 53 5C 10 03 4F 40 5D 51 3C 57 31 24 38 23 05 03 S\..O@]Q[W!\$8H.. |          |                   | 000000c0 53 5C 10 03 4F 40 5D 51 3C 57 31 24 38 23 05 03 S\..O@]Q[W!\$8H.. |
| 000000d0 20 01 0C 4D 4E 3C 19 0A 5A 35 18 5B 32 31 20 54 ..MN<..Z5.[21 I   |          |                   | 000000d0 20 01 0C 4D 4E 3C 19 0A 5A 35 18 5B 32 31 20 54 ..MN<..Z5.[21 I   |
| 000000e0 28 63 57 12 0E 1F 33 49 12 33 3C 49 55 40 1C 45 (cW...3I.3IU0.E   |          |                   | 000000e0 28 63 57 12 0E 1F 33 49 12 33 3C 49 55 40 1C 45 (cW...3I.3IU0.E   |
| 000000f0 11 27 2D 2E 32 15 37 28 49 1F 1E 4A 4F 0D 39 46 '-.2.T(L..TO.9F   |          |                   | 000000f0 11 27 2D 2E 32 15 37 28 49 1F 1E 4A 4F 0D 39 46 '-.2.T(L..TO.9F   |
| 00000100 3F 5F 58 4C 1A 5A 01 2B 28 3C 43 4D 4C 5E 2D 2C ?XL.Z.+<CML"-     |          |                   | 00000100 3F 5F 58 4C 1A 5A 01 2B 28 3C 43 4D 4C 5E 2D 2C ?XL.Z.+<CML"-     |
| 00000110 20 29 59 22 3D 5F 19 55 4D 36 3A 37 13 42 19 51 .Y'=-.UM6.7.B.Q   |          |                   | 00000110 20 29 59 22 3D 5F 19 55 4D 36 3A 37 13 42 19 51 .Y'=-.UM6.7.B.Q   |
| 00000120 3D 40 09 26 35 09 50 2C 14 2E 14 5F 5C 40 5A 4B =@.&5.P....@ZK    |          |                   | 00000120 3D 40 09 26 35 09 50 2C 14 2E 14 5F 5C 40 5A 4B =@.&5.P....@ZK    |
| 00000130 38 4E 08 10 19 20 34 01 26 09 38 08 1B 20 28 57 8N...4.&8..(W     |          |                   | 00000130 38 4E 08 10 19 20 34 01 26 09 38 08 1B 20 28 57 8N...4.&8..(W     |
| 00000140 2F 30 4C 33 38 37 5E 4C 34 42 16 5F 51 0C 46 25 /OL387 L4B.Q.FW   |          |                   | 00000140 2F 30 4C 33 38 37 5E 4C 34 42 16 5F 51 0C 46 25 /OL387 L4B.Q.FW   |
| 00000150 29 1D 34 41 3D 04 42 32 40 49 39 2A 38 60 50 02 .)4A=-B2@I9*8.P.  |          |                   | 00000150 29 1D 34 41 3D 04 42 32 40 49 39 2A 38 60 50 02 .)4A=-B2@I9*8.P.  |
| 00000160 2C 37 34 33 3D 61 1A 0D 3E 64 3B 2B 0B 1C 4F 03 .743a...>d+.0.    |          |                   | 00000160 2C 37 34 33 3D 61 1A 0D 3E 64 3B 2B 0B 1C 4F 03 .743a...>d+.0.    |
| 00000170 39 52 44 45 25 21 46 35 39 4E 5E 0C 49 4A 0D 44 ORDEX'F50N'.I.T.D |          |                   | 00000170 39 52 44 45 25 21 46 35 39 4E 5E 0C 49 4A 0D 44 ORDEX'F50N'.I.T.D |
| 00000180 1C 10 47 59 0C 30 35 1A 2F 0B 14 09 5B 32 0C 63 ...GY.05./..[2.c  |          |                   | 00000180 1C 10 47 59 0C 30 35 1A 2F 0B 14 09 5B 32 0C 63 ...GY.05./..[2.c  |
| 00000190 1F 1F 13 14 3F 58 48 47 41 11 52 59 2A 5E 09 16 ....XHGArY*..     |          |                   | 00000190 1F 1F 13 14 3F 58 48 47 41 11 52 59 2A 5E 09 16 ....XHGArY*..     |
| 000001a0 3D 4F 3E 48 1A 0E 31 19 4C 44 21 12 45 60 10 34 =O>H..1.LD!..E.4  |          |                   | 000001a0 3D 4F 3E 48 1A 0E 31 19 4C 44 21 12 45 60 10 34 =O>H..1.LD!..E.4  |
| 000001b0 1A 22 47 28 49 5E 0A 25 3E 5B 1A 04 24 22 19 60 .G(I'.>)[.\$.     |          |                   | 000001b0 1A 22 47 28 49 5E 0A 25 3E 5B 1A 04 24 22 19 60 .G(I'.>)[.\$.     |
| 000001c0 40 56 14 59 33 44 0A 1A 58 62 60 38 5D 0B 07 47 @V.Y3D..Xb 8].G   |          |                   | 000001c0 40 56 14 59 33 44 0A 1A 58 62 60 38 5D 0B 07 47 @V.Y3D..Xb 8].G   |
| 000001d0 61 1D 0A 15 16 14 3A 54 3E 23 27 62 14 3F 5D 53 a.....T*#b.7j5    |          |                   | 000001d0 61 1D 0A 15 16 14 3A 54 3E 23 27 62 14 3F 5D 53 a.....T*#b.7j5    |
| 000001e0 3D 0C 17 62 20 58 48 13 25 16 1A 1D 55 55 33 51 0..b XK*...U0UQ   |          |                   | 000001e0 3D 0C 17 62 20 58 48 13 25 16 1A 1D 55 55 33 51 0..b XK*...U0UQ   |
| 000001f0 0D 3D 01 57 20 0A 16 5D 60 3C 5A 0F 4A 23 31 15 =.W...]<Z.JW1.    |          |                   | 000001f0 0D 3D 01 57 20 0A 16 5D 60 3C 5A 0F 4A 23 31 15 =.W...]<Z.JW1.    |
| 00000200 62 18 46 1D 0B 60 63 63 12 19 1B 36 3D 4E 22 49 b.P..cc...6=N'I   |          |                   | 00000200 62 18 46 1D 0B 60 63 63 12 19 1B 36 3D 4E 22 49 b.P..cc...6=N'I   |
| 00000210 5A 56 0B 15 30 20 41 2B 2B 07 0A 10 29 3A 58 26 ZV..0 A++...X&    |          |                   | 00000210 5A 56 0B 15 30 20 41 2B 2B 07 0A 10 29 3A 58 26 ZV..0 A++...X&    |
| 00000220 21 09 13 2B 05 11 5D 4A 5D 14 1B 35 31 0C 4E 5A !..+..]J].51.NZ   |          |                   | 00000220 21 09 13 2B 05 11 5D 4A 5D 14 1B 35 31 0C 4E 5A !..+..]J].51.NZ   |
| 00000230 31 58 0A 30 48 1A 5B 0E 20 34 52 48 3D 45 3E 5E 1X.OH.L[.4RH=E>   |          |                   | 00000230 31 58 0A 30 48 1A 5B 0E 20 34 52 48 3D 45 3E 5E 1X.OH.L[.4RH=E>   |
| 00000240 45 20 58 22 64 51 3B 5D 34 25 61 64 64 1A 29 30 M X'q;]4*add.)0   |          |                   | 00000240 45 20 58 22 64 51 3B 5D 34 25 61 64 64 1A 29 30 M X'q;]4*add.)0   |
| 00000250 42 02 60 59 26 3E 5B 23 52 31 3F 5F 5E 21 B..T.&6;T@R18..         |          |                   | 00000250 42 02 60 59 26 3E 5B 23 52 31 3F 5F 5E 21 B..T.&6;T@R18..         |
| 00000260 1A 52 12 1A 3E 1C 4E 0D 40 12 40 3F 2C 04 3E 3D R...>.F.@.0?..>   |          |                   | 00000260 1A 52 12 1A 3E 1C 4E 0D 40 12 40 3F 2C 04 3E 3D R...>.F.@.0?..>   |
| 00000270 39 09 31 53 2E 02 29 56 25 4B 67 5C 45 50 4C 5F 9..1S..)YKwVEPL   |          |                   | 00000270 39 09 31 53 2E 02 29 56 25 4B 67 5C 45 50 4C 5F 9..1S..)YKwVEPL   |
| 00000280 3D 2D 14 4A 48 29 56 23 3A 01 31 01 38 0A 3D 0C -=.JH)W#..1.8.=   |          |                   | 00000280 3D 2D 14 4A 48 29 56 23 3A 01 31 01 38 0A 3D 0C -=.JH)W#..1.8.=   |
| 00000290 13 3D 5F 10 3F 57 02 33 3D 28 5E 52 47 45 4C 20 .=-.7W.3=(RGEL    |          |                   | 00000290 13 3D 5F 10 3F 57 02 33 3D 28 5E 52 47 45 4C 20 .=-.7W.3=(RGEL    |
| 000002a0 41 2F 39 24 57 2B 1E 2C 5F 16 61 33 1F 09 3E 01 A/8\$W+...a3.>.   |          |                   | 000002a0 41 2F 39 24 57 2B 1E 2C 5F 16 61 33 1F 09 3E 01 A/8\$W+...a3.>.   |
| 000002b0 46 38 11 54 5F 46 22 37 3D 1B 24 1F 2F 3F 0E 0B F8.I F'=\$./?..   |          |                   | 000002b0 46 38 11 54 5F 46 22 37 3D 1B 24 1F 2F 3F 0E 0B F8.I F'=\$./?..   |
| 000002c0 09 47 62 2F 0D 47 5B 07 5C 27 09 4A 2F 47 1B 10 .Gb/G/L\..J/G..   |          |                   | 000002c0 09 47 62 2F 0D 47 5B 07 5C 27 09 4A 2F 47 1B 10 .Gb/G/L\..J/G..   |
| 000002d0 1A 2B 63 48 40 54 1B 4C 3E 0E 06 08 1D 14 4E 59 .+cHBT.L.....FY   |          |                   | 000002d0 1A 2B 63 48 40 54 1B 4C 3E 0E 06 08 1D 14 4E 59 .+cHBT.L.....FY   |



# Page fault analysis

According to the final output in the console, the input size is 131072 ( $2^{17}$ ) and the total page faults is **8193**.

```
__device__ void user_program(VirtualMemory *vm, uchar *input, uchar *results,
                             int input_size) {
    for (int i = 0; i < input_size; i++)
        vm_write(vm, i, input[i]);

    for (int i = input_size - 1; i >= input_size - 32769; i--)
        int value = vm_read(vm, i);

    vm_snapshot(vm, results, 0, input_size);
}
```

The code above is the test program in *user\_program.cu*. By running this test program, it

1. writes on the virtual memory with  $2^{17}$  unsigned characters from input buffer in order
2. reads the virtual memory for 32769 ( $2^{15} + 1$ ) unsigned characters in the reverse order
3. does the snapshot, which reads the virtual memory in order (same as 1) for  $2^{17}$  unsigned characters.

Analyse each steps to find how page fault happened:

1. Initially, the page table is empty with 1024 available indexes and each index is corresponding to a page with 32 bytes. Since 1 uchar takes 1 byte, there are total  $2^{17}$  bytes (4096 pages) written in this step. And there will not be two access on the same data address since the input is in order.
  1. For the first 1024 pages, each page just directly used a unused index to write the data into physical memory. Since using an new index caused 1 page fault, here total 1024 page faults were caused.
  2. After step 1, the page table was full, so the page replacement happened for the writing of the rest 3072 pages. Each page replacement caused 1 page, then total 3072 pages faults were caused.

Therefore, for the  $2^{17}$  times of *vm\_write*, 4096 page faults happened.

2. After the *vm\_write* of 4096 pages, the early 3072 pages have been swapped out to the storage (secondary memory), and the most recent 1024 pages are still in the physical memory. In this step, 32769 unsigned characters are read in reverse order, which consists of 1024 pages plus 1 byte.
  1. For the first 1024 pages, since the data was read in reverse order, the target data is the most recent data which is still in the physical memory. Thus the page table could directly access the target data and there is no page fault caused here.
  2. For the additional 1 single byte, it is not recent enough to be included in the page table, so the page replacement is required to swap the corresponding memory from storage into physical memory. This caused 1 page fault.

Therefore, for the 32796 times of *vm\_read*, 1 page fault happened.

3. For the *vm\_snapshot*, the data was read in order for  $2^{17}$  times. Since the first 1024 pages of data is the least recent indexed set, 1024 page replacements are required to get access to the data. After that, the same for the rest 3072 pages of data. A page replacement represents 1 page fault, so total 4096 page faults happen for the *vm\_snapshot*.

Therefore, based on the above analysis, total  $4096 + 1 + 4096 = 8193$  page faults is supposed to happen, which is the same as the counting result of the program.

## Encountered problems

### Problem

To implement the LRU algorithm, how to get the least recent used data set?

### Solution

Observing that there is spare space in the memory for page table setting, I utilized that space to construct a simplified double linked list of indexes by recording the previous used index and the next used index for each index. This could function as a queue so that we could dequeue the least recent used indexed set when page replacement happens.

### Problem

When debugging, the program exited unexpectedly.

### Solution

Function *printf()* was used to keep track of the variables during program running and the output showed that all the variables were going well until the moment that the program exited. This may be caused by segment fault, which refers to the problem of memory overflow. I checked the code and found that, when transferring data to the physical memory, the base of the index was wrongly set as the base address of the input data. Finally I changed the base index to the corresponding index of the page table and the problem was solved.

## What I learned

- Simple CUDA programming
  - Configuration of environment
  - CUDA memory hierarchy
  - Data transfer between memory
  - ...
- Mechanism of virtual memory
  - How does page table work
  - LRU algorithm of page replacement
  - Analysis based on page fault
  - ...

## Bonus

---

Source code has been attached in folder "CSC3150\_A3\_Bonus"

### ✓ Launch 4 threads in kernel function

In *main()*, create 4 threads for kernel function *mykernel*:

```
mykernel<<< 1, 4, INVERT_PAGE_TABLE_SIZE>>>(input_size);
```

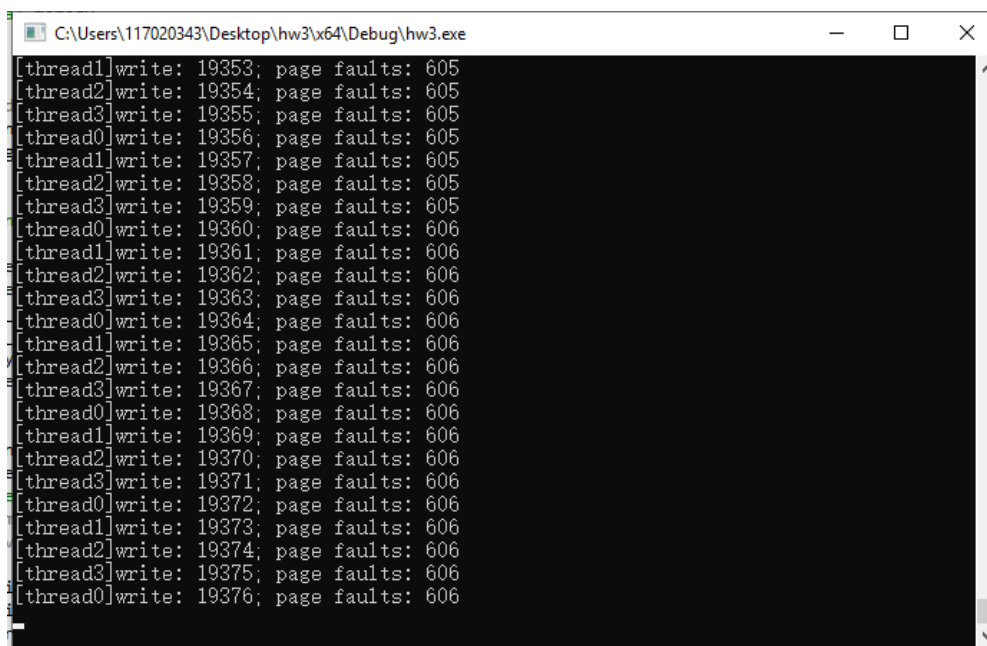
In function *mykernel*, identify the thread id and pass it into VM:

```
int i = blockDim.x * blockIdx.x + threadIdx.x;
```

In each thread, for *vm\_read*, *vm\_write*, *vm\_snapshot*, using thread id to identify the task, exit when the task is not the correct one:

```
if (addr % 4 != vm->tid) return ;
```

### ✓ Non-preemptive priority scheduling



```
C:\Users\117020343\Desktop\hw3\x64\Debug\hw3.exe
[thread1]write: 19353; page faults: 605
[thread2]write: 19354; page faults: 605
[thread3]write: 19355; page faults: 605
[thread0]write: 19356; page faults: 605
[thread1]write: 19357; page faults: 605
[thread2]write: 19358; page faults: 605
[thread3]write: 19359; page faults: 605
[thread0]write: 19360; page faults: 606
[thread1]write: 19361; page faults: 606
[thread2]write: 19362; page faults: 606
[thread3]write: 19363; page faults: 606
[thread0]write: 19364; page faults: 606
[thread1]write: 19365; page faults: 606
[thread2]write: 19366; page faults: 606
[thread3]write: 19367; page faults: 606
[thread0]write: 19368; page faults: 606
[thread1]write: 19369; page faults: 606
[thread2]write: 19370; page faults: 606
[thread3]write: 19371; page faults: 606
[thread0]write: 19372; page faults: 606
[thread1]write: 19373; page faults: 606
[thread2]write: 19374; page faults: 606
[thread3]write: 19375; page faults: 606
[thread0]write: 19376; page faults: 606
```

```
C:\Users\117020343\Desktop\hw3\x64\Debug\hw3.exe
[thread0]Read: 107308; page faults: 4096
[thread3]Read: 107307; page faults: 4096
[thread2]Read: 107306; page faults: 4096
[thread1]Read: 107305; page faults: 4096
[thread0]Read: 107304; page faults: 4096
[thread3]Read: 107303; page faults: 4096
[thread2]Read: 107302; page faults: 4096
[thread1]Read: 107301; page faults: 4096
[thread0]Read: 107300; page faults: 4096
[thread3]Read: 107299; page faults: 4096
[thread2]Read: 107298; page faults: 4096
[thread1]Read: 107297; page faults: 4096
[thread0]Read: 107296; page faults: 4096
[thread3]Read: 107295; page faults: 4096
[thread2]Read: 107294; page faults: 4096
[thread1]Read: 107293; page faults: 4096
[thread0]Read: 107292; page faults: 4096
[thread3]Read: 107291; page faults: 4096
[thread2]Read: 107290; page faults: 4096
[thread1]Read: 107289; page faults: 4096
[thread0]Read: 107288; page faults: 4096
[thread3]Read: 107287; page faults: 4096
[thread2]Read: 107286; page faults: 4096
[thread1]Read: 107285; page faults: 4096
```

```
C:\Users\117020343\Desktop\CSC3150_A3_Bonus\x64\Debug\hw3.exe
In snapshot - [thread0]Read: 77148; page faults: 6508
In snapshot - [thread1]Read: 77149; page faults: 6508
In snapshot - [thread2]Read: 77150; page faults: 6508
In snapshot - [thread3]Read: 77151; page faults: 6508
In snapshot - [thread0]Read: 77152; page faults: 6509
In snapshot - [thread1]Read: 77153; page faults: 6509
In snapshot - [thread2]Read: 77154; page faults: 6509
In snapshot - [thread3]Read: 77155; page faults: 6509
In snapshot - [thread0]Read: 77156; page faults: 6509
In snapshot - [thread1]Read: 77157; page faults: 6509
In snapshot - [thread2]Read: 77158; page faults: 6509
In snapshot - [thread3]Read: 77159; page faults: 6509
In snapshot - [thread0]Read: 77160; page faults: 6509
In snapshot - [thread1]Read: 77161; page faults: 6509
In snapshot - [thread2]Read: 77162; page faults: 6509
In snapshot - [thread3]Read: 77163; page faults: 6509
In snapshot - [thread0]Read: 77164; page faults: 6509
In snapshot - [thread1]Read: 77165; page faults: 6509
In snapshot - [thread2]Read: 77166; page faults: 6509
In snapshot - [thread3]Read: 77167; page faults: 6509
In snapshot - [thread0]Read: 77168; page faults: 6509
In snapshot - [thread1]Read: 77169; page faults: 6509
In snapshot - [thread2]Read: 77170; page faults: 6509
In snapshot -
```

✓ Print the times of page fault of whole system before the program end

```
Microsoft Visual Studio Debug Console
In snapshot - [thread0]Read: 131056; page faults: 8193
In snapshot - [thread1]Read: 131057; page faults: 8193
In snapshot - [thread2]Read: 131058; page faults: 8193
In snapshot - [thread3]Read: 131059; page faults: 8193
In snapshot - [thread0]Read: 131060; page faults: 8193
In snapshot - [thread1]Read: 131061; page faults: 8193
In snapshot - [thread2]Read: 131062; page faults: 8193
In snapshot - [thread3]Read: 131063; page faults: 8193
In snapshot - [thread0]Read: 131064; page faults: 8193
In snapshot - [thread1]Read: 131065; page faults: 8193
In snapshot - [thread2]Read: 131066; page faults: 8193
In snapshot - [thread3]Read: 131067; page faults: 8193
In snapshot - [thread0]Read: 131068; page faults: 8193
In snapshot - [thread1]Read: 131069; page faults: 8193
In snapshot - [thread2]Read: 131070; page faults: 8193
In snapshot - [thread3]Read: 131071; page faults: 8193
pagefault number is 8193

C:\Users\117020343\Desktop\CSC3150_A3_Bonus\x64\Debug\hw3.exe (process 8904) exited with
code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging
->Automatically close the console when debugging stops.
Press any key to close this window . . .
```

✓ Modify paging mechanism to manage multiple threads.

✓ Correctly dump the contents to "snapshot.bin"

| data.bin | virtual_memory.cu       | user_program.cu         | main.cu            | snapshot.bin                     |
|----------|-------------------------|-------------------------|--------------------|----------------------------------|
| 00000000 | 37 0C 60 51 38 5A 25 0B | 13 2A 2A 01 0C 07 05 2F | 7. Q8ZK...*.*/     | 00000000 07 0C 60 51 38 5A 25 0B |
| 00000001 | 62 0B 3A 2B 3E 35 2B 3C | 0E 27 60 3B 04 53 2A 3A | b. +>5+<...S:      | 00000001 62 0B 3A 2B 3E 35 2B 3C |
| 00000002 | 5E 5A 5B 31 1F 1B 0B 01 | 14 04 01 20 0E 05 1E 03 | Z[1.....           | 00000002 5E 5A 5B 31 1F 1B 0B 01 |
| 00000003 | 44 57 02 1D 5C 61 28 05 | 23 5B 3F 5A 46 39 63 3F | DW. \a(<X#Z2F9c?   | 00000003 44 57 02 1D 5C 61 28 05 |
| 00000004 | 62 59 3F 1C 44 49 1C 57 | 1C 51 46 5A 55 63 32 34 | bY?..DI.W.QFZUc24  | 00000004 62 59 3F 1C 44 49 1C 57 |
| 00000005 | 56 33 21 1D 2F 18 21 21 | 0B 30 16 50 38 15 2A 35 | V3!./...0.P8.*5    | 00000005 56 33 21 1D 2F 18 21 21 |
| 00000006 | 3D 38 20 1C 50 3B 43 08 | 5B 24 31 4C 23 62 4F 48 | =8 .P:C.[ \$1L#bOH | 00000006 3D 38 20 1C 50 3B 43 08 |
| 00000007 | 01 3F 64 63 57 54 20 31 | 1F 05 1D 56 4D 16 5A 26 | .?dcWT 1...VM.Z&   | 00000007 01 3F 64 63 57 54 20 31 |
| 00000008 | 1E 15 11 09 20 53 44 16 | 13 11 31 05 0E 1C 4C 0E | .... SD...1...L    | 00000008 1E 15 11 09 20 53 44 16 |
| 00000009 | 2A 1B 41 50 0A 60 1D 5D | 34 39 1E 51 1E 14 46 3B | *.AP. \J49.Q..F    | 00000009 2A 1B 41 50 0A 60 1D 5D |
| 0000000a | 5C 56 44 4B 45 57 61 27 | 03 61 2B 45 4C 46 22 12 | \VDKEW\$a..aELF    | 0000000a 5C 56 44 4B 45 57 61 27 |
| 0000000b | 60 62 61 39 2D 19 31 61 | 21 4F 4D 3F 32 62 49 29 | ba9-.1a!OM?2bI     | 0000000b 60 62 61 39 2D 19 31 61 |
| 0000000c | 53 5C 10 03 4F 40 5D 51 | 3C 57 31 24 38 23 05 03 | S\..o@]Q<W!\$8#..  | 0000000c 53 5C 10 03 4F 40 5D 51 |
| 0000000d | 20 01 0C 4D 4E 3C 19 0A | 5A 35 18 5B 32 31 20 54 | ..MN<..Z5.[21 I    | 0000000d 20 01 0C 4D 4E 3C 19 0A |
| 0000000e | 28 63 57 12 0E 1F 33 49 | 12 33 3C 49 55 40 1C 45 | (cW...3I.3<I0@.E   | 0000000e 28 63 57 12 0E 1F 33 49 |
| 0000000f | 11 27 2D 2E 32 15 37 28 | 49 1F 1E 4A 4F 0D 39 46 | ..- 2.7(I..J0.9F   | 0000000f 11 27 2D 2E 32 15 37 28 |
| 00000010 | 3F 5F 58 4C 1A 5A 01 2B | 28 3C 43 4D 4C 5E 2D 2C | ? XL.Z.+(CML -     | 00000010 3F 5F 58 4C 1A 5A 01 2B |
| 00000011 | 20 29 59 22 3D 5F 19 55 | 4D 36 3A 37 13 42 19 51 | )Y"-..UM6.7.B.Q    | 00000011 20 29 59 22 3D 5F 19 55 |
| 00000012 | 3D 40 09 26 35 09 50 2C | 14 2E 14 5F 5C 40 5A 4B | =@.&5.P....\@ZX    | 00000012 3D 40 09 26 35 09 50 2C |
| 00000013 | 38 4E 08 10 19 20 34 01 | 26 09 38 08 1B 20 28 57 | 8N... 4.&.8..(W    | 00000013 38 4E 08 10 19 20 34 01 |
| 00000014 | 2F 30 4C 33 38 37 5E 4C | 34 42 16 5F 51 0C 46 25 | /OL387'L4B..Q.F#   | 00000014 2F 30 4C 33 38 37 5E 4C |
| 00000015 | 29 1D 34 41 3D 04 42 32 | 40 49 39 2A 38 60 50 02 | ).4A=.B2@I9*8 P    | 00000015 29 1D 34 41 3D 04 42 32 |
| 00000016 | 2C 37 34 33 3D 61 1A 0D | 3E 64 3B 2B 0B 1C 4F 03 | ..743=a..>d+..0.   | 00000016 2C 37 34 33 3D 61 1A 0D |
| 00000017 | 39 52 44 45 25 21 46 35 | 39 4E 5E 0C 49 4A 0D 44 | 9RDE#!F59N'.IJ.D   | 00000017 39 52 44 45 25 21 46 35 |
| 00000018 | 1C 10 47 59 0C 30 35 1A | 2F 0B 14 09 5B 32 0C 63 | ...GT.05./...[2.c  | 00000018 1C 10 47 59 0C 30 35 1A |
| 00000019 | 1F 1F 13 14 3F 58 48 47 | 41 11 52 59 2A 5E 09 16 | ....?XHG\$A.RY*..  | 00000019 1F 1F 13 14 3F 58 48 47 |
| 0000001a | 3D 4F 3E 48 1A 0E 31 19 | 4C 44 21 12 45 60 10 34 | =>H..1.LD!..E'.4   | 0000001a 3D 4F 3E 48 1A 0E 31 19 |
| 0000001b | 1A 22 47 28 49 5E 0A 25 | 3E 5B 1A 04 24 22 19 60 | "G(I'.%>[.\$.E".   | 0000001b 1A 22 47 28 49 5E 0A 25 |
| 0000001c | 40 56 14 59 33 44 0D 1A | 58 62 60 38 5D 0B 07 47 | @W.Y3D..Xb 8]..G   | 0000001c 40 56 14 59 33 44 0D 1A |
| 0000001d | 61 1D 0A 15 16 14 3A 54 | 3E 23 27 62 14 3F 5D 53 | A.....T#?'b.?)S    | 0000001d 61 1D 0A 15 16 14 3A 54 |
| 0000001e | 30 0C 17 62 20 58 4B 13 | 25 16 1A 1D 55 55 33 51 | 0..b XK.%...UU3Q   | 0000001e 30 0C 17 62 20 58 4B 13 |
| 0000001f | 0D 3D 01 57 20 0A 16 5D | 60 3C 5A 0F 4A 23 31 15 | ..w..J<Z.J#I.      | 0000001f 0D 3D 01 57 20 0A 16 5D |
| 00000200 | 62 19 46 1D 0B 60 63 63 | 12 19 1B 36 3D 4E 22 49 | b.F.. cc...6=N'I   | 00000200 62 19 46 1D 0B 60 63 63 |
| 00000201 | 5A 56 0B 15 30 20 41 2B | 2B 07 0A 10 29 3A 58 26 | ZV..0 A++..)X&     | 00000201 5A 56 0B 15 30 20 41 2B |
| 00000202 | 21 09 13 2B 05 11 5D 4A | 5D 14 1B 35 31 0C 4E 5A | !..+. [J].51.NZ    | 00000202 21 09 13 2B 05 11 5D 4A |
| 00000203 | 31 58 0A 30 48 1A 5B 0E | 20 34 52 48 3D 45 3E 5E | IX.OH.[. 4RH=E>    | 00000203 31 58 0A 30 48 1A 5B 0E |
| 00000204 | 4E 20 58 22 64 51 3B 5D | 34 25 61 64 64 1A 29 30 | N X"dq:]4kadd.'0   | 00000204 4E 20 58 22 64 51 3B 5D |
| 00000205 | 42 02 60 59 1B 26 36 3B | 59 23 52 31 38 5F 5E 21 | B..Y.&6:YWR18.'!   | 00000205 42 02 60 59 1B 26 36 3B |
| 00000206 | 1A 52 12 1A 3E 1C 46 0D | 40 12 40 3F 2C 04 3E 3D | .R..>.F.@?..)=     | 00000206 1A 52 12 1A 3E 1C 46 0D |
| 00000207 | 39 09 31 53 2E 02 29 56 | 25 4B 57 5C 45 50 4C 5F | 9.15..)WKW\$EPL    | 00000207 39 09 31 53 2E 02 29 56 |
| 00000208 | 3D 2D 14 4A 48 29 56 23 | 3A 01 31 01 38 0A 3D 0C | =-.JH)YH..1.8.-    | 00000208 3D 2D 14 4A 48 29 56 23 |
| 00000209 | 13 3D 5F 10 3F 57 02 33 | 3D 28 5E 52 47 45 4C 20 | ..-7W.3=(RGEL      | 00000209 13 3D 5F 10 3F 57 02 33 |
| 0000020a | 41 2F 39 24 57 2B 16 2C | 5F 16 61 33 1F 09 3E 01 | A/9\$W+...a3.>.    | 0000020a 41 2F 39 24 57 2B 16 2C |
| 0000020b | 46 38 11 54 5F 46 22 37 | 3D 1B 24 1F 2F 3F 0E 0B | F8.T.F'7=.\$./?..  | 0000020b 46 38 11 54 5F 46 22 37 |
| 0000020c | 09 47 62 2F 0D 47 5B 07 | 5C 27 09 4A 2F 47 1B 10 | .Gb/.G[.\'J/G..    | 0000020c 09 47 62 2F 0D 47 5B 07 |
| 0000020d | 1A 2B 63 48 40 54 1B 4C | 3E 0E 06 08 1D 14 46 59 | ..+cH@T.L>....FY   | 0000020d 1A 2B 63 48 40 54 1B 4C |