Question 5. Missing the TLB [12 MARKS]

TLB misses can be nasty. The following code can cause a lot of TLB misses, depending on the values of STRIDE and MAX. Assume that your system has a 32-entry TLB with a 8KB (2^{13} bytes) page size. Assume that the code runs on a 32-bit machine architecture. Also, assume that malloc below returns a page-aligned address.

```c
int value = 0;
int *data = malloc(sizeof(int) * MAX); // an array of MAX integers
for (int j = 0; j < 1000; j++) {
    for (int i = 0; i < MAX; i += STRIDE) {
        value = value + data[i];
    }
}
```

**Part (a) [6 MARKS]** Choose the minimum value for MAX and for STRIDE so that every access to the data array causes a TLB miss (100% TLB miss rate). Use the space below to show any calculations. If you make any assumptions, state them below.

STRIDE =

MAX =

**Part (b) [2 MARKS]** Would the page fault rate 1) increase, 2) remain the same, or 3) decrease, if the MAX value is made 10 times the value of MAX you have chosen in Part(a) (STRIDE is unchanged)? Explain your answer.

**Part (c) [2 MARKS]** Would the page fault rate 1) increase, 2) remain the same, or 3) decrease, if the MAX and STRIDE values are both made 10 times the values of MAX and STRIDE you have chosen in Part(a)? Explain.

**Part (d) [2 MARKS]** Say we doubled the TLB size (64 entries). With the MAX and STRIDE values you have chosen in Part(a), would the TLB miss rate be 1) less than 10%, 2) around 50%, or 3) greater than 90%?
Question 5. Missing the TLB [12 MARKS]

TLB misses can be nasty. The following code can cause a lot of TLB misses, depending on the values of STRIDE and MAX. Assume that your system has a 32-entry TLB with a 8KB (2^{13} bytes) page size. Assume that the code runs on a 32-bit machine architecture. Also, assume that malloc below returns a page-aligned address.

```c
int value = 0;
int *data = malloc(sizeof(int) * MAX); // an array of MAX integers
for (int j = 0; j < 1000; j++) {
    for (int i = 0; i < MAX; i += STRIDE) {
        value = value + data[i];
    }
}
```

Part (a) [6 MARKS] Choose the minimum value for MAX and for STRIDE so that every access to the data array causes a TLB miss (100% TLB miss rate). Use the space below to show any calculations. If you make any assumptions, state them below.

Assuming that the TLB entries are replaced using LRU (or randomly, or any reasonably replacement algorithm), if the array is 33 pages long, and a new page is accessed when data[i] is accessed, then each such access will likely cause a TLB miss. This will not cause a page fault because 33 pages is just 33 * 8KB = 264KB of memory, which is pretty small on modern machines.

$$\text{STRIDE} = \text{STRIDE} \geq 8\text{KB/sizeof(int)} = 8\text{KB}/4 = 2048$$

We divide by sizeof(int) above because data is an int array. With a stride value of 2048, each read of data[i] will access a new page.

$$\text{MAX} = \text{MAX} \geq 33 \times \text{STRIDE} = 33 \times 2048 = 67584$$

If the answer mentioned that the local variables and code will require 1 page each, we also accepted 31 \times \text{STRIDE}.

Part (b) [2 MARKS] Would the page fault rate 1) increase, 2) remain the same, or 3) decrease, if the MAX value is made 10 times the value of MAX you have chosen in Part(a) (STRIDE is unchanged)? Explain your answer.

Assuming any contention (other programs are running), page fault will increase. Page fault will remain the same, only if a lot of physical memory is available.

Part (c) [2 MARKS] Would the page fault rate 1) increase, 2) remain the same, or 3) decrease, if the MAX and STRIDE values are both made 10 times the values of MAX and STRIDE you have chosen in Part(a)? Explain.

Page fault will remain the same because the number of unique pages that are accessed compared to Part(a) is the same.

Part (d) [2 MARKS] Say we doubled the TLB size (64 entries). With the MAX and STRIDE values you have chosen in Part(a), would the TLB miss rate be 1) less than 10%, 2) around 50%, or 3) greater than 90%?

The TLB miss rate will be less than 10%, likely close to 0, because the number of TLB entries is larger than 33, and we are accessing 33 pages repeatedly.