QUESTION 1

Let's consider an array of integers of size 8. We would like to sort the elements of the array in an ascending way using the Bitonic sort algorithm. We focus on step No 2. Give the following information related to every stage ***i*** of step 2:

1. Give the IDs of threads involved in stage ***i***
2. Infer the condition that is satisfyied by the involved thread in stage ***i*** .
3. Give for every thread, involved in stage ***i*** , the bitonic sequence (just give an interval [***b***, ***e***] where ***b*** is the index of the first cell of the sequence and the ***e*** is the index of the last cell of the sequence) that the thread is processing
4. Specify for every thread whether the corresponding sequence is sorted in an ascending (+BM) or a descending (-BM) way.

QUESTION 2

Let's consider an array of integers of size 8. We would like to sort the elements of the array in an ascending way using the Bitonic sort algorithm. We focus on step No 1. Give the following information related to step 1 .

1. Give the IDs of threads involved instep 1.
2. Give for every thread the bitonic sequence (just give an interval [***b***, ***e***] where ***b*** is the index of the first cell of the sequence and the ***e*** is the index of the last cell of the sequence) that the thread is processing
3. Specify for every thread whether the corresponding sequence is sorted in an ascending (+BM) or a descending (-BM) way.

QUESTION 3

Let's consider an array of integers of size 32. We would like to sort the elements of the array in an ascending way using the Bitonic sort algorithm:

1. Give the number of steps that are required to sort the elements of the array.
2. Give the size of bitonic sequences in every step.
3. Give the bitonic sequences (just give an interval [***b***, ***e***] where ***b*** is the index of the first cell of the sequence and the ***e*** is the index of the last cell of the sequence) that are processed in every step.
4. Specify, in every step, for every bitonic sequence whether the sequence is sorted in an ascending (+BM) or a descending (-BM) way.

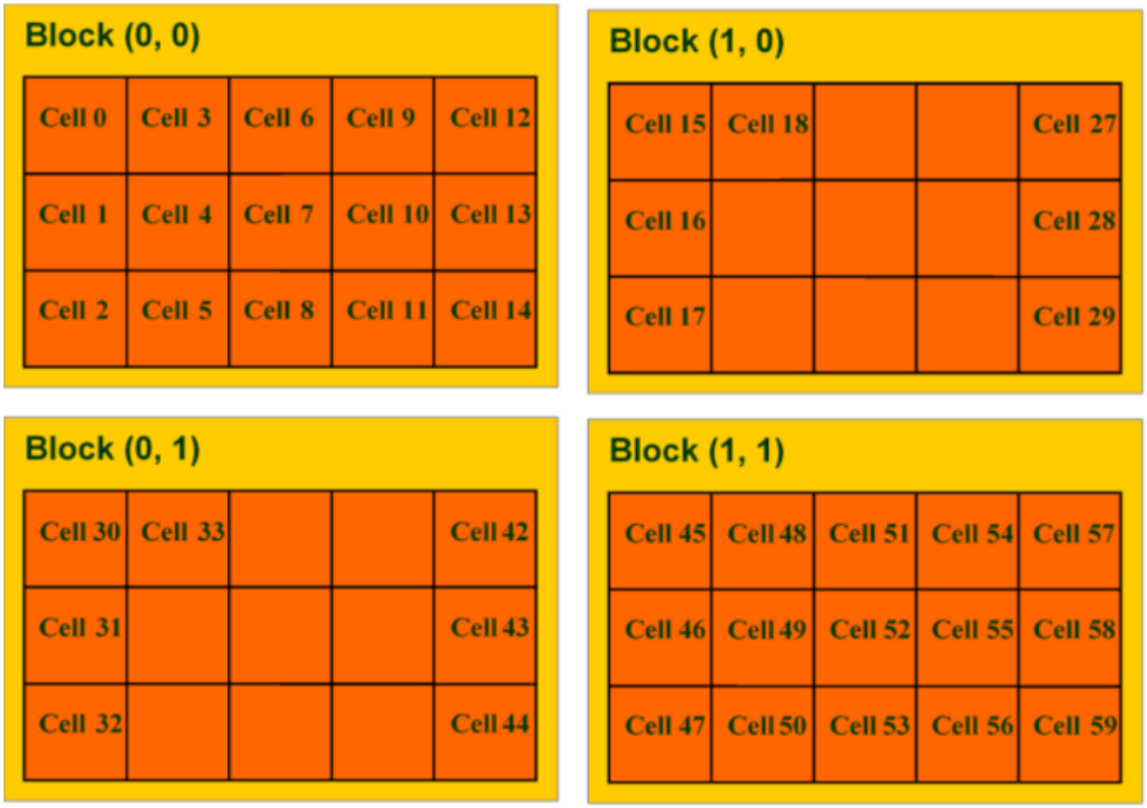
QUESTION 4

Let's consider 2 arrays of integers called A and B respectively. Let's consider that we would like to write a C program that runs in parallel and that computes the sum of the 2 arrays:

$$C[i] = A[i] + B[i]$$

1. Write the kernel (called ***add_kernel***) that will run on N Blocks of M threads where every thread processes W elements.
2. Write the main that will call the kernel ***add_kernel*** to compute the sum of two arrays in parallel and displays the result.

Question 5: Find the cell_index



Question 6

```
__global__ void Kernel_A(int *data) {
    data[threadIdx.x] = threadIdx.x;
    __syncthreads();
    if (threadIdx.x == 0) {
        Kernel_C<<< 1, 256 >>>(data);
        Kernel_D<<< 1, 256 >>>(data);
        cudaDeviceSynchronize();
    }
    __syncthreads();
}
__global__ void Kernel_C(int *data) {
    data[threadIdx.x] = threadIdx.x;
    __syncthreads();
    if (threadIdx.x == 0) {
        Kernel_E<<< 1, 256 >>>(data);
        cudaDeviceSynchronize();
    }
    __syncthreads();
}

void host_launch(int *data) {
     kernel_A<<< 1, 256 >>>(data);
     kernel_B<<< 1, 256 >>>(data);
     cudaDeviceSynchronize();
}
```

1. Give and explain the order of execution of the given parallel nested kernels.
2. Explain the role of the `__syncthreads()` statements.
3. Explain the role of the `cudaDeviceSynchronize()` statements

QUESTION 7

Let's consider that a kernel A launches a Kernel B. Explain the memory synchronization (what the child kernel can see and when the parent kernel can read the child writes) between the parent and the child kernels.