Pseudo Code:

The process contains two steps:
Step 1: Generate the bitonic sequence
Step 2: Perform the bitonic sort

Algorithm:

1. A random array is generated.
2. The array is distributed across the processes using scatter operation.
3. Each process now needs to compare an element in its portion with another element,
4. There will be log n steps where n is the data size
5. In the first step we compare the elements at positions which only differ at the first bit e.g. we will compare 0000 and 0001, 0010 and 0011 and so on. For the second iteration we will compare the elements that differ at the second bit and then compare the bits that differ at the first bit, and so on,
   There are two possibilities for the same:
   1. The other element is in the same process, this case is easy since it has access to the element and can compare and swap if necessary.
   2. The other element in some other process, in this case blocking send and receive, is used to exchange data. If the rank of the next process is greater than the current process then the current process will send its data and then receive the data from the other process. If the rank of the next process is less than the current process then the current process will first receive the data and then send the data. To summarize the process with lesser rank will send the data first and then receive data, while the process with greater rank will receive data first and then send data.
      The plus and minus switches are handled depending on the number of iterations and the postion of the current element
6. I have used a gather operation to get the data on the process with rank = 0, not needed but just to check that the generated sequence is bitonic.
7. Now all processes have a part of elements, and the sequence is bitonic so we can go ahead and perform bitonic sort.
8. For bitonic sort we first compare the elements that differ at the most significant bit, then at the second most significant bit and so on.
9. Finally a gather operation to get the entire array at the process with rank = 0