Odd-Even sort with $O(n)$ processors

The following operation is the basic step in all comparison sort algorithm.
Compare and exchange : Given x and y, if $x > y$ exchange the values of x and y.

Goal: Given elements $a_0, \ldots, a_{n-1}$ sort them in ascending order.(wlog assume n is even)

Outline of the algorithm
There are $n-1$ rounds: round-0 to round-(n-2)
In round-i,
If i is even, j-th processor compares and exchanges a_{2j} and a_{2j+1}. (where $0 \leq j < n/2$)
If i is odd, j-th processor compares and exchanges a_{2j+1} and a_{2j+2}. (where $0 \leq j < n/2 - 1$)

Odd-Even sort with k processors

Goal: Given n elements $a_0, \ldots, a_{n-1}$ and k processors $p_0, \ldots, p_{k-1}$, sort the elements in ascending order. Wlog, assume $2k$ divides n and let $m = n/(2k)$. By A_j we refer to the block of elements $a_{mj}, \ldots, a_{mj+m-1}$.

Here, the compare and exchange operation is replaced by the following.

Merge and split : Given two sorted array of size m, merge them into one sorted array of size $2m$ (exactly like the merging operation in merge-sort which can be done in $O(m)$ time). And then split it from the middle into two arrays of size m each.

Outline of the algorithm
First p_j sorts A_{2j} and A_{2j+1} using any efficient sorting algorithm. (where $0 \leq j < k$)
Then there are $2k-1$ rounds: round-0 to round-$(2k-2)$
In round-i,
If i is even, p_j merges and splits A_{2j} and A_{2j+1}. (where $0 \leq j < k$)
If i is odd, p_jr merges and splits A_{2j+1} and A_{2j+2}. (where $0 \leq j < k-1$)

MPI Implementation

Initially p_0 generates n random elements. Scatter the data so that p_j has the data A_{2j} and A_{2j+1}.
Then p_j sorts A_{2j} and A_{2j+1}
At the begining of round-i, perform the necessary SendRecv operation to ensure that,
If i is even, p_j has the data A_{2j} and A_{2j+1}.
If i is odd, p_j has the data A_{2j+1} and A_{2j+2}.
Then perform the merge and split operation in p_j
After $2k-1$ rounds, gather all the data into p_0.