3.2 Deriving the Optimal Data Layout

The previous lemma tells us that, in order to avoid remote accesses, after at most $\lg n$ steps we have to remap the data again. A remap strategy that will execute exactly $\lg n$ steps locally before remapping again will generate the smallest possible number of remaps. Therefore, with respect to the number of data remaps, we will have a communication optimal parallel bitonic sort algorithm.

The example in Figure 3.3 ($N = 256$ elements, $P = 16$ processors) shows how we can remap the data so that after each remap operation exactly $\lg n$ steps will execute locally. The example shows only the last $\lg P$ stages of a bitonic sorting network and assumes that the first $\lg n$ stages execute locally under a blocked layout. In this example we execute only 7 data remaps, while in the cyclic-blocked implementation we were executing 8 remaps. Furthermore, we will show that at each remap we transfer less elements than in the case of a cyclic-blocked remap. Figure 3.4 presents the bit patterns of the absolute addresses of the nodes that go on the same processor at each remap operation in Figure 3.3, where the shaded bits represent the processor number. The bit pattern address representation will play an important role in deriving an optimal data layout.

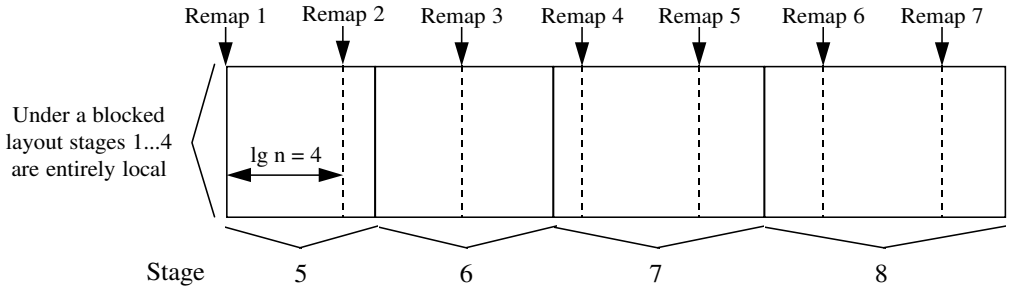

Figure 3.3: *Executing* $\lg n$ *steps locally after each remap operation* ($N = 256$ *elements,* $P = 16$ *processors,* $n = N/P = 4$ *elements per processor).*