

Synthesis for FPGAs with Embedded Memory Blocks

Jason Cong and Kenneth Yan
Department of Computer Science
University of California, Los Angeles, CA 90095
{cong,kyan}@cs.ucla.edu

ABSTRACT

Embedded memory blocks (EMBs) are used in modern field programmable gate arrays (FPGAs) for implementation of on-chip memories or specialized logic functions[1]. In this paper, we propose an integrated approach with structural clustering and functional decomposition to minimize the circuit area using EMBs while preserving the circuit delay. The structural clustering method is based on the concepts of Maximum Fanout Free Cone (MFFC)[5] and Maximum Fanout Free Subgraph (MFFS)[5]. In order to effectively use EMB in large clusters, single-output and multiple-output functional decompositions are used to decompose large clusters so that the encoding functions or base functions can be implemented by EMBs. It also considers multiple EMBs for individual large cluster so that better area reduction can be obtained. We have developed an algorithm called EMB_Syn that can be used as a postprocessing tool in the FPGA synthesis flow. MCNC benchmarks are used to test EMB_Syn on Altera’s FLEX10K device family and the experimental results are compared with those by EMB_Pack[8] and SMAP[10]. When EMB_Syn is used as postmapping processing, it shows 45.06% and up to 5.23% improvements over EMB_Pack and SMAP, respectively, in terms of the covered area by EMBs.

1. INTRODUCTION

FPGAs have gained wide acceptance due to its programming flexibility and fast turnaround. Designers are able to greatly reduce the design and verification time in a variety of FPGA applications. Embedded memory blocks(EMBs) in modern FPGA chips further enhance the programming flexibility because they provide efficient implementation of on-chip memory or specialized logic functions. Figure 1 illustrates the device block diagram of FLEX10K from Altera[1]. In addition to the logic array of 4-LUTs, FLEX10K contains an embed-

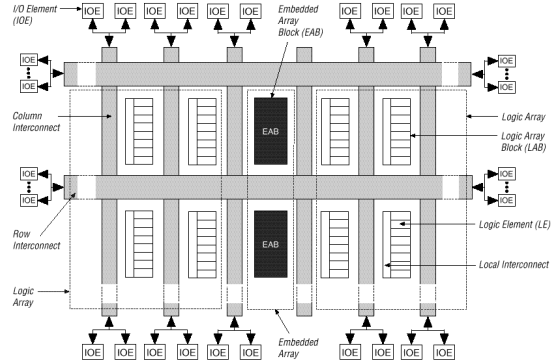


Figure 1: Altera FLEX10K device block diagram

ded memory array with a series of EMBs¹. Each EMB has 2K bits and can be configured as 256 X 8, 512 X 4, 1024 X 2, or 2048 X 1 random access memory. Therefore, each EMB can implement any single 11-input 1-output, 10-input 2-output, 9-input 4-output, or 8-input 8-output logic function. The number of available EMBs in the device depends on the device size². The delay of an EMB in FLEX10K devices is about 3 to 4 times that of a normal LUT. When EMBs are not used for on-chip memory or specialized logic implementation, they can be used to implement general logic functions as shown in [8] and [10]. In order to utilize these EMBs wisely as logic functions, one needs to extract single-output and multiple-output logic blocks from the circuit for EMB mapping and consider the corresponding delays that the EMBs impose on the circuit.

In [8], EMB_Pack uses MFFCs and MFFSs as EMB candidates and heuristic is devised to extract the root sets such that large MFFSs can be obtained. The root nodes computed by their heuristic are close to each other for better routability. Timing slack is computed for each node and it is used in the mapping phase to ensure the circuit delay is maintained when EMBs are added. There is a fixed number of EMBs available for mapping. Therefore, after extracting the single/multiple-output clusters³ from the circuit, a greedy approach is used to select the large clusters for map-

¹In Altera FLEX10K devices, LUT is called *logic element (LE)* and *embedded memory blocks (EMB)* is called *embedded array block*.

²The number of EMBs in Altera FLEX10K device family varies from 6 to 12.

³Single-output logic for the 11X1 EMB configuration

ping into the available EMB resources. In SMAP[10], a *maximum volume k-feasible cut* is created for each node and a set of nodes separated from the PIs by the cut is computed. These nodes are root candidates for creating multiple-MFFCs below the k-feasible cut. The largest feasible MFFCs for each cut are selected for EMB mapping to minimize circuit area. Both techniques used in [8] and [10] are combinatorial in nature. They cannot effectively handle large Fanout-Free Cone (FFCs) or large Fanout-Free Subgraph (FFS) that exceed the EMB’s input/output limits. To go beyond the combinatorial limit during synthesis for FPGAs with EMBs, we like to decompose the large clusters so that they can be considered in the EMB implementation as well. For example, Figure 2(a) illustrates a 13-input 2-output logic function with 11 LUTs that cannot be covered by the 2K EMB configurations because 13-input 2-output is not compatible with any of the 4 EMB configurations (11 X 1, 10 X 2, 9 X 4, and 8 X 8). Figure 2(b) illustrates the decomposed logic function. The number of inputs and outputs for the subcircuit are 9 and 4 respectively and it can be implemented by a 9 X 4 EMB. Therefore, five 4-LUTs and one EMB are needed to implement the same circuit in Figure 2(a). A reduction of 6 LUTs is possible with 1 EMB in this example. With this motivation in mind, we want to investigate using functional decompositions (defined in Section 2) along with structural clustering to map logic blocks into available EMBs on FPGA chip so that circuit area can be minimized. The circuit delay has to be maintained after EMB mapping.

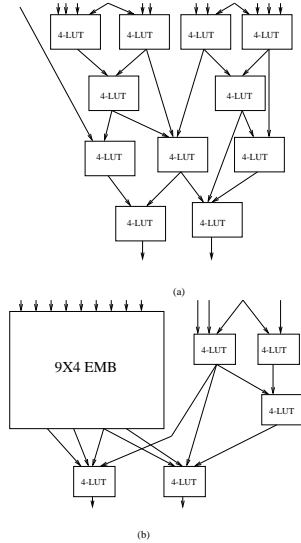


Figure 2: (a) A 13-input 2 output logic function that cannot be covered by a 2K EMB. (b) After decomposition on this logic function, part of the circuit can be mapped by a 2K 9X4 EMB. Therefore, one EMB and five 4-LUTs are needed to create the 2-output logic that is functionally equivalent to the one in Figure 2(a).

A comprehensive survey of FPGA mapping algorithms is presented in [4]. In this paper, we present a new FPGA synthesis method with new structural clustering and functional decomposition for the FPGAs with EMBs. The nov-

erty lies in (1) the integration of structural clustering and functional decomposition and (2) multiple EMBs are considered to cover a large cluster if the cover leads to good area reduction. After structural clustering, multiple-output functional decomposition is performed which allows sharing of the encoding functions among different outputs. We also consider the use of EMBs to implement both encoding functions and base functions after functional decomposition.

The remainder of this paper is organized as follows. Section 2 presents preliminaries and problem formulation. Section 3 discusses the EMB_Syn algorithm with MFFC/MFFS clustering and functional decomposition for FPGAs with EMBs. Heuristics for computing root sets for MFFSs are discussed. They take bound set selection into consideration and search for MFFSs that are more decomposable. Experimental results and conclusion are given in Section 4 and Section 5 respectively.

2. PRELIMINARIES AND PROBLEM FORMULATION

A boolean network can be represented as a directed acyclic graph (DAG) where each node represents a logic gate, and a directed edge (i, j) exists if the output of gate i is an input of gate j . A primary input (PI) node has no incoming edge and a primary output (PO) node has no outgoing edge. We use $input(v)$ to denote the set of nodes which are fanins of gate v , and $output(v)$ to denote the set of nodes which are fanouts of gate v . Given a subgraph H of the boolean network, $input(H)$ denotes the set of *distinct* nodes outside H which supply inputs to the gates in H . Node u is the *transitive fanin* of node v if there is a path from u to v . Similarly, node u is the *transitive fanout* of node v if there is a path from v to u .

A *cone* at v , denoted as C_v , is a subgraph consisting of v and its predecessors such that any path connecting a node in C_v and v lies entirely in C_v . The notation of $input(C_v)$ is also used to represent the set of distinct nodes outside C_v which supply inputs to the gates in C_v . A *maximum cone* at v , also the fanin network of v , denoted as N_v , is a cone consisting of v and all of its predecessors. A cone C_v is said to be *k-feasible* if and only if $|input(C_v)| \leq k$.

Given a network N with a source s and a sink t , a cut (X, X') is a partition of the nodes in the network such that $s \in X$, $t \in X'$ and no nodes in X' provide input to any node in X . Clearly X' may be considered as a cone at t inside network N . A *cut-set* of a cut is the set of all nodes v such that $v \in X$ and v drives a node in X' . If the size of the cut is no more than k , the cut is said to be *k-feasible*. Given a cone C_v rooted at v , the *maximum-volume k-feasible cut* is the *k-feasible cut* (X, X') with the largest number of nodes in X' . A *fanout-free cone* (FFC) at v , denoted FFC_v is a cone of v such that for any node $u \neq v$ in FFC_v , $output(u) \subseteq FFC_v$.

Definition 1 (MFFC) In a directed acyclic graph which represents a combinational circuit, the *maximum fanout-free cone* (MFFC) of v , denoted $MFFC_v$, is an FFC of v such that for any non-PI node w , if $output(w) \subseteq MFFC_v$, then $w \in MFFC_v$.

For a given node set S , a *fanout-free subgraph* of S , denoted $FFS(S)$, is a subgraph consisting of S and its predecessors such that any path connecting a node in the subgraph and a node v , $v \in S$, lies entirely in the subgraph. Also, any node $u \notin S$ in $FFS(S)$, $output(u) \subseteq FFS(S)$.

Definition 2 (MFFS) The maximum fanout-free subgraph of S , denoted $MFFS(S)$, is an FFS of S such that for any non-PI node w , if $output(w) \in MFFS(S)$, then $w \in MFFS(S)$. S is called the root set of $MFFS(S)$, and the nodes in S are the root nodes of $MFFS(S)$. An $MFFS(S)$ is said to be a k -input m -output MFFS if $input(MFFS(S)) = k$ and $|S| = m$.

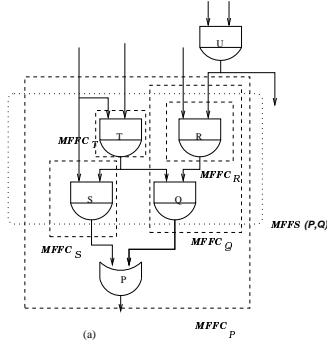


Figure 3: Examples of MFFC and MFFS in subcircuit.

Examples of MFFC and MFFS are shown in Figure 3. In combinational circuits, single-output MFFS is equivalent to MFFC. MFFCs and MFFSs are ideal candidates for EMB mapping because nodes covered by one MFFC or MFFS do not have fanouts outside this MFFC or MFFS. Each of them can be implemented by an EMB without duplication. *Feasible* MFFCs or MFFSs are clusters with input size and output size that do not exceed the EMB input size and output size for at least one EMB configuration. Otherwise they are *infeasible* MFFCs/MFFSs. To minimize the area of a K -LUT circuit is to cover part of the circuit using EMBs so that the remaining subcircuit can be implemented by the minimum number of K -LUTs.

Functional decomposition is a common technique to reduce the number of inputs to a fixed size. This technique transforms an n -variable function into a composition of several functions each depending on fewer than n variables. The decomposition schemes can be characterized by the nature of the subfunctions as follows.

Definition 3 Single-Output Decomposition Given a function $f(x, y)$ and a partition of its n input variables into the bound set $BS = x_1, \dots, x_b$ and the free set $FS = y_1, \dots, y_{n-b}$, single-output functional decomposition determines encoding functions d_1, \dots, d_c and the base function g such that

$$f(x_1, \dots, x_b, y_1, \dots, y_{n-b}) = g(d_1(x_1, \dots, x_b), \dots, d_c(x_1, \dots, x_b), y_1, \dots, y_{n-b}) \quad (1)$$

A decomposition is *non-trivial* when $c < b$.

Definition 4 Multiple-Output Decomposition Given a function vector $f = (f_1, \dots, f_p)$ and a partition of its n input variables into the bound set $BS = x_1, \dots, x_b$, and the free set $FS = y_1, \dots, y_{n-b}$, multiple-output functional decomposition determines an encoding function assignment $A_f = d_1, \dots, d_q$ and the base functions g_1, \dots, g_p such that for each j , $1 \leq j \leq p$,

$$f_j(x_1, \dots, x_b, y_1, \dots, y_{n-b}) = g_j(d_1^j(x_1, \dots, x_b), \dots, d_{c_j}^j(x_1, \dots, x_b), y_1, \dots, y_{n-b}) \quad (2)$$

where $\forall_i d_i^j \in A_f$.

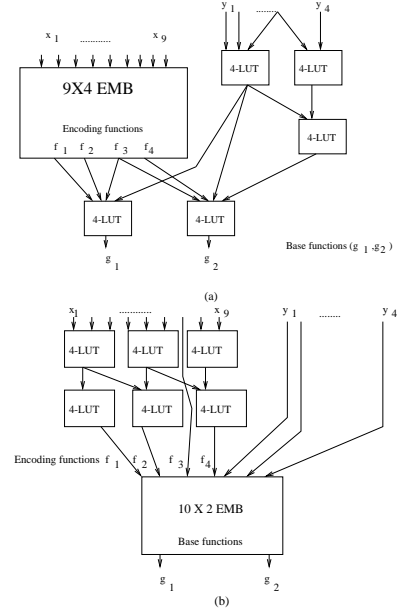


Figure 4: (a) Use a 2K EMB to implement the encoding functions of the logic function in Figure 2(a) after multiple-output functional decomposition. (b) Use a 2K EMB to implement the base functions of the same function.

For example, Figure 4 illustrates using a 2K EMB to implement the encoding functions or base functions after multiple-output functional decomposition of a 2-output logic function. Note that they result in different LUT reductions. In Figure 4, five 4-LUTs and one EMB are needed when the encoding functions are mapped by EMB. Six 4-LUTs and one EMB are needed when the base functions are mapped by EMB.

Problem 1 Given a mapped combinational K -LUT network N and the FPGA with λ EMBs whose architecture is described in the previous section, use functional decomposition and MFFC and MFFS clustering so that the circuit area is minimized while maintaining the circuit delay.

3. EMB_SYN ALGORITHM

3.1 Overview of EMB_Syn

The primary goal of EMB_Syn is to utilize the embedded memory blocks to minimize the circuit area while maintaining the circuit delay. Given a K -LUT network, EMB_Syn extracts large single-output and multiple-output fanout-free logic blocks and covers them entirely or partially by EMBs. It consists of 4 main phases: (1) It computes a set of MFFCs and MFFSs in the network for possible EMB implementations (Section 3.2) (2) The *infeasible* MFFCs are functionally decomposed to cater the EMB configurations (Section

3.3). (3) Select the clusters from the EMB candidates in (1) and (2) for mapping while preserving the circuit delay in the original K -LUT circuit (Section 3.4). (4) if the selected EMB candidate is a *feasible* MFFS, compute a larger *infeasible* MFFS from it and perform functional decomposition to see if better area reduction can be achieved.

Input: seed node, I/O specifications, bound set size
Output: MFFS and bound set variables
procedure shared_input_root_set(v, k, m, l)
1 /* v =seed node, k =input size,
2 m =output size, l =bound set size */
3
4 compute max-volume k -feasible cut of v
5 E = set of extremal nodes of the cut
6 m_root_set = select m nodes from E
7 with the biggest MFFCs below the cut
8
9 compute MFFS(m_root_set) below the cut
10 $retval.mffs_nodes$ = MFFS(m_root_set)
11 $retval.seed_nodes$ = v
12 $retval.num_input$ = k
13 $retval.num_output$ = m
14 $retval.bound_set$ = l inputs shared by max outputs
15 in m_root_set
16
17 return ($retval$)

Table 1: Template for shared_input_root_set

Input: seed node, EMB input/output specifications
Output: MFFS with the best LUT reduction per EMB
procedure root_set_for_large_MFFS(v, k, m)
1 /* v =a seed node, k =EMB input size,
2 m =EMB output size */
3
4 compute max-volume k -feasible cut of v
5 E = compute extremal nodes of the cut
6 m_root_set = select m nodes from E
7 with the biggest MFFCs below the cut
8 compute MFFS(m_root_set) below the cut
9 $retval1.LUT_reduction$ =size of MFFS(m_root_set)
10 $retval1.num_EMB_needed$ =1
11 $retval1.mffs_nodes$ = MFFS(m_root_set)
12
13 compute MFFS(E) below the cut
14 num_EMB =number of EMBs to cover MFFS(E)
15 $retval2.LUT_reduction$ =size of MFFS(E)/ num_EMB
16 $retval2.num_EMB_needed$ = num_EMB
17 $retval2.mffs_nodes$ = MFFS(E)
18
19 if ($retval1.LUT_reduction > retval2.LUT_reduction$)
20 return ($retval1$)
21 else
22 return ($retval2$)

Table 2: Template for root_set_for_large_MFFS

Note that the *feasible* MFFCs and MFFSs can be replaced by EMBs directly without duplication to reduce circuit area. They are EMB candidates for the EMB selection process. We want to decompose the *infeasible* MFFCs and *infeasible* MFFSs so that more opportunities to reduce the circuit area can be explored. After functional decomposition, if the numbers of inputs and outputs of the encoding functions or the base functions are compatible with the EMB specifications, they become EMB candidates. If no such functional decomposition is possible, the functional decomposition rou-

tine returns failure and the *infeasible* MFFC/MFFS cannot be considered in the selection process for EMB mapping.

3.2 MFFC and MFFS computation

To compute $MFFC_v$ for node v , (i) initialize $MFFC_v = v$; (ii) construct $MFFC_v$ by repeatedly including a transitive fanin u of v as soon as $output(u) \subseteq MFFC_v$ and stop when no such predecessor exists. To compute $MFFS(S)$ for a given root set S , (i) initialize $MFFC(S) = S$; (ii) construct $MFFS(S)$ by repeatedly including a transitive fanin u of some node in S as soon as $output(u) \subseteq MFFS(S)$ and stop when no such predecessor exists. These algorithms run in $O(e)$ time which e is the number of edges in the combinational network.

The construction of MFFS depends heavily on how we choose the root set. The root sets are important because we want (1) a set of large *feasible* MFFSs. (2) a set of large MFFSs that can be covered by multiple EMBs. (3) a set of large infeasible MFFSs that are decomposable and produce good area reduction during EMB mapping. The effectiveness of multiple-output functional decomposition is determined by (1) The number of encoding functions that are shared by outputs. (2) The complexity of the encoding functions. (3) The complexity of the base functions. (1) leads directly to good reduction of the number of inputs to all base functions g_j in equation 2. This gives higher feasibility with different EMB configurations. (2) and (3) are important in EMB mapping because simple encoding and base functions can be mapped by a smaller number of LUTs and result in smaller circuit. In order to exploit the multiple-output functional decomposition, a good root set should produce maximum sharing of the encoding functions when multiple-output functional decomposition is performed. Bound set variables for decomposition can be pre-determined to help achieving this goal. Root set of size m are m -root set. Since it is computationally expensive to enumerate all $O(n^m)$ m -root sets for large circuits of n nodes, a heuristic is used to find the root sets with shared inputs efficiently. They are based on the following observation.

Observation: Input variables that are shared among the output functions are encouraged to be used as bound set variables because they are likely to produce encoding functions that are inherently shared by output functions.

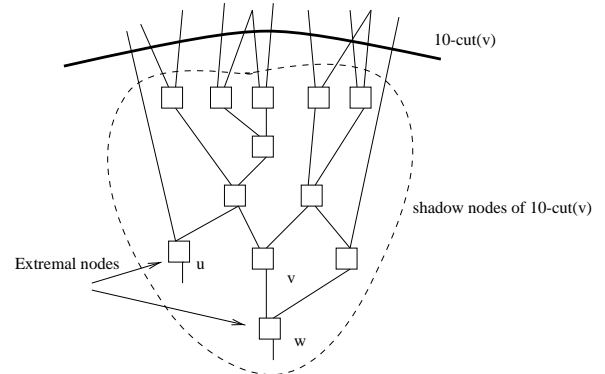


Figure 5: Shadow nodes and extremal nodes for node v .

```

procedure EMB_Syn ( $N, EC, \lambda$ )
1  /*  $N$ =input network,  $EC$ =EMB configurations,  $\lambda$ =number of EMBs */
2
3  /* compute MFFCs and timing slacks */
4  for each node  $v$  in  $N$  do
5    compute  $timing\_slack(v)$ 
6    compute  $MFFC_v$ 
7    if  $MFFC_v$  is feasible
8      add  $MFFC_v$  to  $EMB\_candidate$ 
9
10 /* compute MFFSs */
11 for each node  $v$  do
12   for each EMB configuration  $i$  in  $EC$  do
13     compute MFFS  $s$  using  $root\_set\_for\_large\_MFFS$  ( $v$ , num of EMB input, num of EMB output) (in Table 2)
14     add MFFS  $s$  to  $EMB\_candidate$ 
15
16 /* single-output functional decomposition */
17 for each infeasible MFFC  $c$  do
18   for each EMB configuration  $i$  in  $EC$  do
19      $success = single\_output\_func\_decomp(c, \text{number of input of EMB config. } i)$ 
20     if  $success$ 
21        $num\_1 = LUT\_reduction$  for this infeasible MFFC by mapping the encoding functions
22        $num\_2 = LUT\_reduction$  for this infeasible MFFC by mapping the base function
23        $LUT\_reduction = MAX(num\_1, num\_2)$ 
24     else
25        $LUT\_reduction = 0$ 
26     add MFFC  $c$  with best LUT_reduction to  $EMB\_candidate$ 
27
28 /* EMB selection */
29 while there is EMB available
30   select a non-overlapped cluster from  $EMB\_candidate$  for EMB mapping with highest LUT_reduction per EMB
31
32   if it is feasible MFFS do
33      $v$ =seed node,  $k$ =num of input,  $m$ =num of output,  $l$ =num of input for this cluster
34     repeat
35        $k = k+1$  /* increase the cut size */
36       compute infeasible MFFS  $s$  using  $shared\_input\_root\_set(v, k, m, l)$  (in Table 1)
37
38       /* multiple-output functional decomposition */
39       for each EMB configuration  $i$  in  $EC$  do
40          $success = multi\_output\_func\_decomp(s, \text{number of input for EMB config. } i)$ 
41         if  $success$ 
42            $num\_1 = LUT\_reduction$  for this infeasible MFFS by mapping the encoding functions
43            $num\_2 = LUT\_reduction$  for this infeasible MFFS by mapping the base functions
44            $LUT\_reduction = MAX(num\_1, num\_2)$ 
45         else
46            $LUT\_reduction = 0$ 
47       until no improvement in LUT reduction for this cluster
48       map cluster into a EMB or EMBs if circuit delay does not increase
49

```

Table 3: The EMB_Syn algorithm.

Heuristic : Shared Input Root Set from Shadow Nodes (Table 1) First, compute a *maximum-volume k -feasible cut* $(v)^4$ for a node v . Node v is called the seed node. Find the *shadow nodes* and *extremal nodes* of the *k -feasible cut* (v) as follows:

Definition 5 Shadow Node Given a node v and a *k -feasible cut* (v) . The nodes that form such cut are the *cut set*. A node s is called the *shadow node* if all paths from the primary inputs to node s go through the *cutset* of the *k -feasible cut* (v) .

Definition 6 Extremal Node The set of *extremal nodes* E is a subset of shadow nodes S such that for each node v in E , $u \in output(v)$ implies that $u \notin S$. Please see Figure 5 for illustration.

We set k = number of inputs in the EMB configuration. After finding a set of extremal nodes, we compute the MFFC below the cut for each extremal node. As in SMAP's approach, we select the m extremal nodes with the largest MFFCs below the cut as *m -root* for MFFS. To take decom-

position into consideration, the shared input variables are used as bound set which is used in functional decomposition later.

After the root sets are computed, we compute the MFFS that is below the *k -feasible cut*. The MFFS can be covered by the EMB regardless of whether $|input(MFFS(m-root))| > k$. In other words, the MFFS(m -root) below the cut is guaranteed to be *k -feasible* because only k inputs in the cut set are needed to generate the MFFS outputs.

We devise a heuristic in Table 2 to compute a set of large feasible MFFSs and large MFFSs that can be covered by multiple EMBs. First, we compute a *max-volume k -feasible cut* for a seed node v . We compute *shadow nodes* and *extremal nodes* for the cut. Select m extremal nodes that have the biggest MFFCs below the cut and compute the MFFS below the cut for them. This is the LUT reduction if a single EMB is used to cover this MFFS. To consider the possibility of using multiple EMBs to cover the large cluster, we also use all extremal nodes as root set and compute the MFFS below the cut for them. Next, we compute the number of EMBs that are needed to cover the cluster and the aver-

⁴We use FlowPack to compute such cut.

age LUT reduction for each EMB. The single or multiple EMB implementation can be selected in final implementation later depending on which one results in higher average LUT reduction for each EMB.

3.3 Functional Decomposition for Infeasible MFFCs and MFFSs

One major goal of multiple-output functional decomposition is to find a maximum number of encoding functions that can be shared among different output functions and therefore minimizing the number of functions in encoding function assignment A_f . Ideally, the number of encoding functions q is smaller than the sum over the number of encoding functions needed to decompose each individual output. For synthesis and EMB mapping, our objective of using functional decomposition is to use the encoding function outputs as the EMB outputs. In other words, for a $k \times m$ EMB, we choose k input variables as bound set for functional decomposition and these become the inputs to the EMB. To get a successful functional decomposition for the EMB covering, we need to obtain no more than m encoding functions. These encoding functions can be implemented as the EMB outputs.

Functional decomposition methods based on *Binary Decision Diagrams (BDD)* were proposed in papers like [9] and [11]. In [9] encoding functions can be computed efficiently by the cut set in the OBDD of the single-output function. In IMODEC[11] multiple-output functional decomposition, due to the *implicit* computation in the crucial steps of their algorithm and manipulating only the *preferable* functions, their decomposition algorithm is very efficient and it produces good decomposition results. Please see [11] for details. Therefore, the multiple-output functional decomposition technique in [11] is implemented in this paper. Let num_global_class = the number of global compatible classes for multiple-output functions. In IMODEC, $log(num_global_class)$ is the minimum number of encoding functions required to decompose the multiple-output functions. Therefore we only continue the decomposition if $log(num_global_class)$ does not exceed the number of EMB outputs. We try functional decomposition for the different EMB configurations. The size of the bound set is the same as the number of EMB inputs. We fill the bound set with shared input variables of the MFFS. If the number of EMB input exceeds the number of shared variables, we fill the remaining bounded variables with the non-shared input variables.

The encoding function mapping or base function mapping in EMB_Syn is done by using PRAETOR[7] program for area minimization. The number of LUTs used for the encoding functions or base functions is subtracted from the size of the MFFC or MFFS to obtain the LUT reduction for EMB implementation.

3.4 Selection from EMB Candidates

A fixed number of EMBs are available on a FLEX10K chip depending on the device size. After we have a set of candidates for EMB implementation, we need to select a subset of them to be replaced by the EMBs. We want to select MFFCs or MFFSs such that the circuit area can be minimized. Currently a greedy approach similar to [8] is used. We iteratively select the MFFC or MFFS with the largest reduction without overlapping with the other selected MFFCs or MFFSs. Before EMB selection, a *timing slack* for

every node in the network is computed which represents the maximum delay that can be added to this single node to maintain the critical delay of the circuit. We only replace the clusters with EMBs if the timing constraint is not violated. For a *feasible* MFFS, we incrementally increase its input size(cut size) and compute a larger infeasible MFFS below the cut using the heuristic in Table 1. Next, multiple-output functional decomposition is used to decompose the MFFS and check if better LUT reduction can be achieved. The EMB_Syn algorithm is summarized in Table 3.

4. EXPERIMENTAL RESULTS

EMB_syn has been implemented with C language on SUN SPARC workstation and is integrated into the RASP system[6]. Our experimental setup is focused on using EMB_Syn in postmapping process. Cutmap[3] is an improvement over FlowMap[2], which considers area minimization during delay optimization. It is used to generate the initial mapped circuits with 4-LUTs as Altera provides 4-LUTs in the FLEX10K device family. The delay of the EMBs is set to be three times of the LUT delay. Table 4 shows the CPU run time for running EMB_Syn on a Sun Ultra-5/10 330MHz dual CPUs with 1 gigabytes of memory. EMB_Pack and EMB_Syn are used as postprocessing optimization and their results are presented in Table 5. Arithmetic and geometric means are given to show the area reduction in terms of number of 4-LUTs covered by both algorithms. Altera MAX+PLUSII 8.1 reads the *tdf* (text description file) after EMB_Syn has synthesized and mapped the circuit. It goes through placement and routing and actual critical delay is reported. Same number of EMBs are used for EMB_Pack and EMB_Syn. Similar to EMB_Pack, EMB_Syn considers timing slacks and will not increase the delays of the timing-critical paths during synthesis and mapping. Same set of MCNC benchmarks reported in [8] are used in the comparison. 45.06% improvement in area reduction is obtained for EMB_Syn over EMB_Pack while the circuit delays are almost the same. Note that we skip 9sym, 9symml, and rd84 because each of these circuits can be covered trivially by 1 EMB.

In Table 6, EMB_Syn is used to compare with SMAP[10]. Since the primary goal of SMAP is to reduce circuit area, EMB_Syn does not consider critical path delays during EMB candidate selection. Results using 1, 4, 8, or 16 EMBs are shown and EMB_Syn outperforms SMAP by up to 5.23%.

	# LUT	1 EMB	4 EMB	8 EMB	16 EMB
C5315	596	30.3	33.4	35.7	37.4
C7552	679	37.5	39.2	40.1	41.6
i10	994	41.0	42.5	43.2	44.4
apex3	867	40.2	41.3	42.4	43.6
apex4	1262	56.9	57.0	58.2	60.7
pd	4575	200.8	202.2	205.5	207.4

Table 4: Sample CPU run times in seconds for running EMB_Syn on a Sun Ultra-5/10 330MHz dual CPUs with 1 gigabytes of memory.

5. CONCLUSION

In this paper we have presented a new FPGA synthesis approach for the FPGAs with EMBs. It combines functional decomposition and structural clustering to map subcircuits into EMBs to get maximum area reduction. The structural

	Cutmap	CutMap+EMB_Pack				CutMap+EMB_Syn			
	# LUT	# CLUT	Depth	# EMB	Delay(ns)	# CLUT	Depth	# EMB	Delay(ns)
C5315	673	48	9	3	59.9	53	9	3	59.1
C6288	555	22	25	3	102.4	46	25	3	105.2
C7552	850	56	8	5	67.7	74	8	5	67.9
C880	142	28	13	3	66.8	32	13	3	64.2
9sym	90	90	3	1	25.5	90	3	1	25.6
9symml	94	94	3	1	25.5	94	3	1	25.3
alu2	192	18	11	3	61.1	128	11	3	62.3
alu4	326	22	13	3	67.2	104	13	3	66.5
apex6	300	21	6	3	44.6	30	6	3	45.2
apex7	88	14	5	3	34.6	20	5	3	34.4
des	1367	38	6	5	77.6	44	6	5	77.9
i10	1166	84	13	8	163.6	116	13	8	163.3
pair	641	70	7	3	47.6	84	7	3	48.1
rd84	83	83	3	1	25.5	83	3	1	25.7
total	6567	688	125	45	869.6	998	125	45	870.7
Ar Mean	469.07	49.14	8.92	3.21	62.11	71.28	8.92	3.21	62.19
Ar Ratio	N/A	1	1	1	1	+45.06%	1	1	+0.13%
Geo Mean	302.38	40.34	7.40	2.73	53.93	62.50	7.40	2.73	53.96
Geo Ratio	N/A	1	1	1	1	+54.93%	1	1	+0.05%

Table 5: Comparison between EMB.Pack and EMB.Syn with delay constraint.

	Mapped circuit+SMAP					Mapped circuit+EMB_Syn			
		1 EMB	4 EMB	8 EMB	16 EMB	1 EMB	4 EMB	8 EMB	16 EMB
	# LUT	# CLUT	# CLUT	# CLUT	# CLUT	# CLUT	# CLUT	# CLUT	# CLUT
C5315	596	12	42	76	141	14	47	83	160
C6288	527	19	61	93	140	21	66	97	157
C7552	679	15	57	94	163	19	69	99	171
apex1	696	14	47	87	159	14	53	91	168
apex3	867	11	75	119	199	17	69	125	210
apex4	1262	319	1205	1261	1261	327	1213	1262	1262
cps	749	46	117	173	249	53	122	173	250
ex5p	1064	198	810	1043	1056	207	823	1052	1077
i10	994	18	55	94	170	23	55	94	182
pair	641	13	45	81	148	17	47	82	161
pdc	4575	88	277	480	816	91	283	482	824
spla	3690	67	200	349	569	72	202	353	571
total	16340	820	2991	3950	5071	875	3051	3993	5193
Ar Mean	1361.66	68.33	249.25	329.16	422.58	71.91	254.25	332.75	432.75
Ar Ratio	N/A	1	1	1	1	+5.23%	+2.00%	+1.09%	+2.40%
Geo Mean	1027.95	33.65	118.99	188.37	292.63	38.95	124.95	193.35	307.32
Geo Ratio	N/A	1	1	1	1	+15.75%	+5.00%	+2.64%	+5.02%

Table 6: Comparison between SMAP and EMB_Syn without delay constraint.

clustering is based on MFFC/MFFS and the root set selection takes the bound set and the feasibility for EMB mapping into consideration. Both encoding functions or base functions are considered for EMB mapping for area reduction. In addition, multiple EMBs are considered for individual large cluster to better explore the solution space. The EMB_Syn algorithm can be extended in several directions. We like to create new root_select heuristic that can lead to simpler logic in the encoding functions and base functions for EMB mapping. In addition, routability should be considered during EMB mapping since routing structure tends to be overloaded, leading to sub-optimal performance and difficulties to implement incremental design changes.

6. ACKNOWLEDGEMENT

The authors would like to thank Yean-Yow Hwang, Chang Wu, and Songjie Xu for their helpful discussions. This work is partially supported by Altera Corporation and Vantis Corporation under the California MICRO Program.

7. REFERENCES

- [1] Altera, "Programmable Logic Devices Data Book", Altera Corp., San Jose, CA 1996.
- [2] J. Cong and Y. Ding, "An Optimal Technology Mapping Algorithm for Delay Optimization in Lookup-Table Based FPGA Design", Proc. of International conference on Computer-Aided Design, November 1992, pp. 48-52.
- [3] J. Cong and Y.-Y. Hwang, "Simultaneous Depth and Area Minimization in LUT-based FPGA Mapping", Proc. of International Symposium of FPGA, pp. 68-74.
- [4] J. Cong and Y. Ding, "Combinational Logic Synthesis for LUT Based Field Programmable Gate Arrays", ACM TODAE April 1996, Vol. 1, No. 2, pp. 145-204.
- [5] J. Cong, P. Li, S.-K. Lim, T. Shibuya, and D. Xu, "Large Scale Circuit Partitioning with Loose/Stable

Net Removal and Signal Flow Based Clustering”, Proc. of International Conference on Computer-Aided Design, November 1997, pp. 441-446.

- [6] J. Cong, J. Peck and Y. Ding, “RASP: A General Logic Synthesis System for SRAM-Based FPGAs”, Proc. of International Symposium of FPGA 1996, pp. 137-143.
- [7] J. Cong, C. Wu, and E. Ding “Cut Ranking and Pruning: Enabling A General And Efficient FPGA Mapping Solution”, Proc. of International Symposium on FPGAs 1999, pp. 29-35.
- [8] J. Cong and S. Xu, “Technology Mapping for FPGAs with Embedded Memory Blocks”, Proc. of International Symposium on FPGAs 1998, pp. 179-188.
- [9] Y.T. Lai, K.R.R. Pan, and M. Pedram, “OBDD-Based Function Decomposition: Algorithms and Implementation”, IEEE Transaction on Computer-Aided Design of Integrated Circuits and Systems, vol. 15, No. 8, August 1996, pp. 977-990.
- [10] S.J.E. Wilton, “SMAP:Heterogeneous Technology Mapping for Area Reduction in FPGAs with Embedded Memory Arrays”, Proc. of International Symposium on FPGAs 1998, pp.171-178.
- [11] B. Wurth, K. Eckl, and K. Antreich, “Functional Multiple-Output Decomposition: Theory and an Implicit Algorithm”, Proc. of Design Automation Conference 1995, pp. 54-59.