ELectronics EXpress

IEICE Electronics Express, Vol.9, No.17, 1414–1422

MH^4 : multiple-supply-voltages aware high-level synthesis for high-integrated and high-frequency circuits for HDR architectures

Shin-ya Abe[1a)], **Youhua Shi**[2], **Masao Yanagisawa**[3], **and Nozomu Togawa**[1b)]

[1] *Dept. of Computer Science and Engineering, Waseda University*

[2] *Waseda Institute for Advanced Study, Waseda University*

[3] *Dept. of Electronic and Photonic Systems, Waseda University*

3–4–1 Okubo, Shinjuku-ku, Tokyo 169–8555, Japan

a) *shinya.abe@togawa.cs.waseda.ac.jp*

b) *togawa@togawa.cs.waseda.ac.jp*

Abstract: In this paper, we propose multiple-supply-voltages aware high-level synthesis algorithm for *HDR architectures* which realizes high-speed and high-efficient circuits. We propose three new techniques: *virtual area estimation*, *virtual area adaptation*, and *floorplanning-directed huddling*, and integrate them into our HDR architecture synthesis algorithm. Virtual area estimation/adaptation effectively estimates a huddle area by gradually reducing it during iterations, which improves the convergence of our algorithm. Floorplanning-directed huddling determines huddle composition very effectively by performing floorplanning and functional unit assignment inside huddles simultaneously. Experimental results show that our algorithm achieves about 29% run-time-saving compared with the conventional algorithms, and obtains a solution which cannot be obtained by our original algorithm even if a very tight clock constraint is given.

Keywords: high-level synthesis, energy-optimization, interconnection delay, multiple supply voltages, distributed-register architecture

Classification: Integrated circuits

References

[1] S. Abe, M. Yanagisawa, and N. Togawa, "An energy-efficient high-level synthesis algorithm for huddle-based distributed-register architectures," *Proc. ISCAS 2012*, pp. 576–579, 2012.

[2] J. Cong, Y. Fan, G. Han, X. Yang, and Z. Zhang, "Architecture and synthesis for on-chip multicycle communication," *IEEE Trans. Computer-*

DOI: 10.1587/elex.9.1414
Received August 09, 2012
Accepted August 20, 2012
Published September 12, 2012

Aided Design Integr. Circuits Syst., vol. 23, no. 4, pp. 550–564, 2004.
[3] J. Cong, Y. Fan, and J. Xu, "Simultaneous resource binding and interconnection optimization based on a distributed register-file microarchitecture," *ACM Trans. Design Automation of Electronic Systems*, vol. 14, no. 3, pp. 1–31, 2009.
[4] J. Jeon, D. Kim, D. Shin, and K. Choi, "High-level synthesis under multicycle interconnect delay," *Proc. ASP-DAC '01*, pp. 662–667, 2001.
[5] D. Kim, J. Jung, S. Lee, J. Jeon, and K. Choi, "Behavior-to-placed rtl synthesis with performance-driven placement," *Proc. ICCAD '01*, pp. 320–325, 2001.
[6] A. Manzak and C. Chakrabarti, "A low power scheduling scheme with resources operating at multiple voltages," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 10, no. 1, pp. 6–14, 2002.
[7] A. Ohchi, S. Kohara, N. Togawa, M. Yanagisawa, and T. Ohtsuki, "Floorplan-driven high-level synthesis for distributed/shared-register architectures," *IPSJ Trans. System LSI Design Methodology*, vol. 1, pp. 78–90, 2008.
[8] A. Ohchi, N. Togawa, M. Yanagisawa, and T. Ohtsuki, "Floorplan-aware high-level synthesis for generalized distributed-register architectures," *IEICE Trans. Fundamentals*, vol. 92, no. 12, pp. 3169–3179, 2009.
[9] S. Tanaka, M. Yanagisawa, T. Ohtsuki, and N. Togawa, "A fault-secure high-level synthesis algorithm for rdr architectures," *IPSJ Trans. System LSI Design Methodology*, vol. 4, pp. 150–165, 2011.
[10] C. Tran, H. Kawaguchi, and T. Sakurai, "Low-power high-speed level shifter design for block-level dynamic voltage scaling environment," *Proc. ICICDT 2005*, pp. 229–232, 2005.
[11] K. Usami, M. Igarashi, F. Minami, T. Ishikawa, M. Kanzawa, M. Ichida, and K. Nogami, "Automated low-power technique exploiting multiple supply voltages applied to a media processor," *IEEE J. Solid-State Circuits*, vol. 33, no. 3, pp. 463–472, 1998.
[12] H. Yang and L. Dung, "On multiple-voltage high-level synthesis using algorithmic transformations," *Proc. ASP-DAC '05*, pp. 872–876, 2005.

1 Introduction

Low-power and energy-efficiency are the important factors in recent LSI design. On the other hand, interconnection delays become dominant in the total circuit delays as device feature size decreases. High-level synthesis, which is one of the LSI design automation techniques, should deal with both energy-efficiency and interconnection delays.

Several energy-aware high-level synthesis algorithms have been proposed which utilize multiple supply voltages [6, 12]. Several interconnection delay aware high-level synthesis algorithms have been also proposed which target distributed-register architectures [2, 3, 4, 5, 7, 8, 9]. However, these algorithms only consider either of energy-efficiency or interconnection delays.

A high-level synthesis algorithm targeting huddle-based distributed-register architectures (HDR architectures) has been proposed in [1] where energy-efficiency and interconnection delays are considered simultaneously. As shown in Fig. 1 (a), an HDR architecture is one of the distributed-register

DOI: 10.1587/elex.9.1414
Received August 09, 2012
Accepted August 20, 2012
Published September 12, 2012

architectures, which divides chip area into several partitions called *huddles*. Inside each huddle, interconnection delays can be small enough and can be ignored, which means that we can only consider inter-huddle interconnection delays in high-level synthesis. Moreover, we can assign high supply voltages to critical huddles and low supply voltages to non-critical huddles. We can easily realize multiple-supply-voltages aware high-level synthesis. In [1], scheduling/binding as well as floorplanning are simultaneously optimized by using iterative synthesis flow (Fig. 1 (b)). The algorithm [1], however, has the two severe problems: (A) the huddle-area and interconnection-delay oscilla-

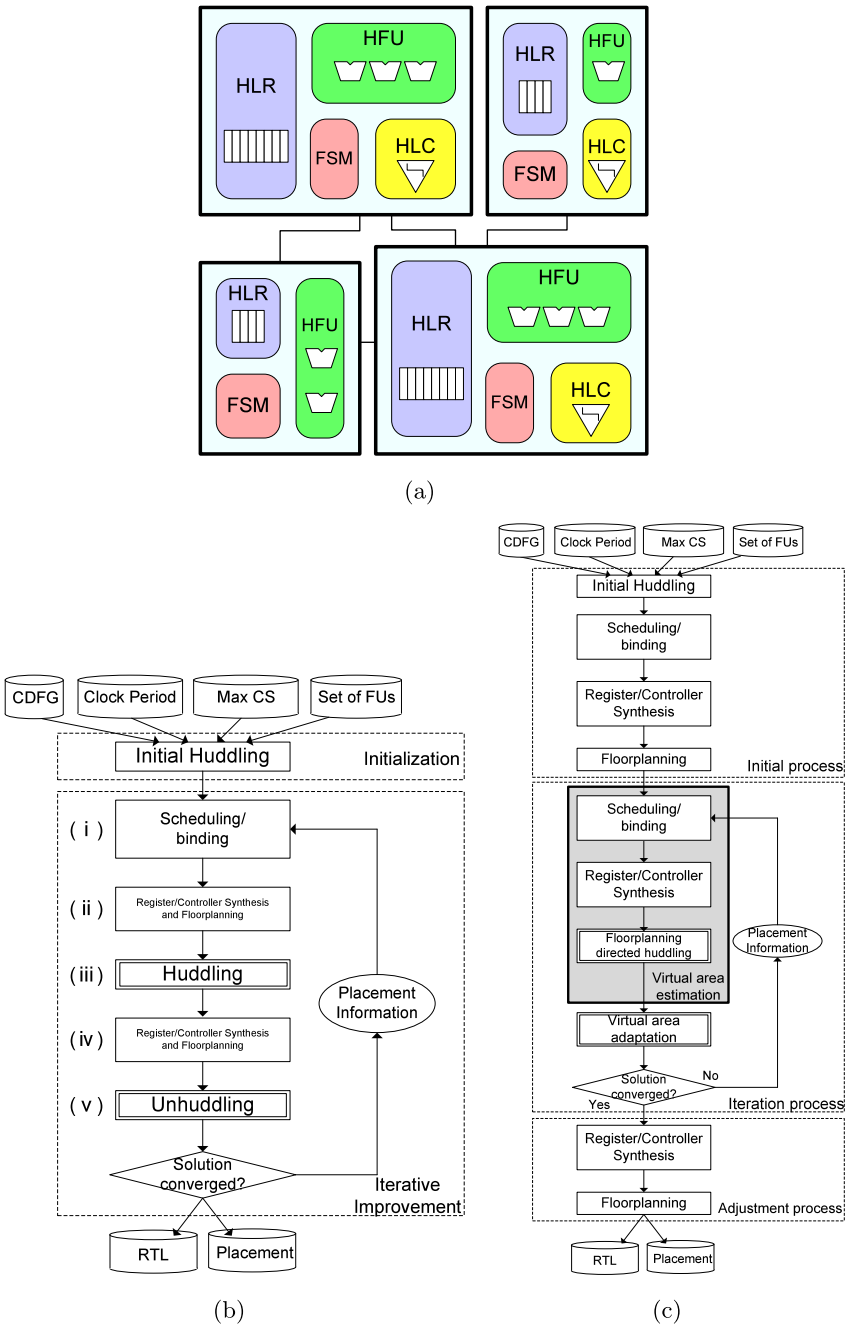

Fig. 1. (a) An HDR architecture. (b) The original algorithm targeting HDR architectures [1]. (c) Our proposed MH^4 algorithm.

DOI: 10.1587/elex.9.1414
Received August 09, 2012
Accepted August 20, 2012
Published September 12, 2012

tion during iterations and (B) the insufficient huddle construction methods.

In this paper, we propose three new techniques, *virtual area estimation*, *virtual area adaptation*, and *floorplanning-directed huddling* to resolve the problems (A) and (B) and then we propose a new multiple-supply-voltages aware high-speed and high-efficiency high-level synthesis algorithm for HDR architecture called MH^4 Experimental results show that our algorithm achieves about 29% run-time-saving compared with the conventional methods, and successfully obtains a solution which cannot obtained by our original algorithm.

2 The MH^4 Algorithm

Our proposed algorithm targets HDR architectures (Fig. 1 (a)). Our high-level synthesis problem is, for a given control data flow graph (CDFG), a clock cycle constraint, a control step constraint, and a set of functional units, to assign each operation node to a control step and a functional unit, to bind each functional unit to each huddle, and to assign a supply voltage to each huddle so that the given CDFG is executed correctly considering multi-cycle interconnect communications. The objective is to minimize the total energy consumption. See [1] in detail.

In [1], we have proposed a high-level synthesis algorithm for HDR architectures where scheduling/binding as well as floorplanning are simultaneously optimized by using iterative synthesis flow (Fig. 1 (b)). The original algorithm [1] has the two problems:

(A) Huddle-area and interconnection-delay oscillation:

In our original algorithm, huddle areas and interconnection delays may be oscillated during iterations. For example, Figs. 2 (a), (b), (c), and (d) show this problem.

(B) The insufficient huddle construction methods:

In our original algorithm, huddles are generated by the two steps *huddling* and *unhuddling*. Since they are not much dependent on each other, we may have poor huddle construction finally. Moreover, huddle construction is composed of *merge*, *partition*, and *transfer* but the original algorithm only considers merge and partition.

In order to resolve the above problems, we propose three new techniques as follows:

1. **Virtual area estimation:**

   We introduce *a virtual area* into each huddle. Virtual areas of huddles do not oscillate in our iterations.

2. **Virtual area adaptation:**

   Virtual area estimation above may have some area and interconnection delay overheads. Our virtual area adaptation relaxes these overheads as the iterations proceed.

DOI: 10.1587/elex.9.1414
Received August 09, 2012
Accepted August 20, 2012
Published September 12, 2012

3. Floorplanning-directed huddling:

In our original algorithm, the two steps, huddling and unhuddling, are executed based on floorplanning results but we embed them into floorplanning as *floorplanning-directed huddling*.

Based on these three techniques, we propose a new multiple-supply-voltages aware high-speed and high-efficiency high-level synthesis algorithm for HDR architectures called MH^4 (Fig. 1 (c)). MH^4 is mainly composed of the three processes: initial process, iteration process, and adjustment process. In the initial process, initial huddle placement is determined. In the iteration process, we perform scheduling/binding and floorplanning repeatedly based on *virtual area estimation/adaptation*, where huddles are constructed by *floorplanning-directed huddling*. When no timing violations occur in floorplanning-directed huddling, the iteration is finished and we go to the adjustment process. In the adjustment process, real area of each huddle is estimated by the scheduling/binding result obtained in the iteration process. Huddles which no functional unit is assigned to are eliminated in the adjustment process.

Since the processes in MH^4 other than *virtual area estimation*, *virtual area adaptation*, and *floorplanning-directed huddling* are the same as the ones in [1], we explain here each of the three new techniques.

2.1 Virtual area estimation

If we employ the *maximum area* obtained so far in each iteration as an estimated huddle area, the huddle area estimation cannot oscillate and we can expect that the solution will converge very fast without oscillating. The estimated area here is called *virtual area*.

Let $A_{real}(h_j)$ be the original area estimation of the huddle h_j in [1] and $A_{virtual}(h_j)$ be our *virtual area* of huddle h_j. $A_{real}(h_j)$ is called a *real area*. Initial value of $A_{virtual}(h_j)$ is set to be the real area obtained by the initial process in MH^4. In each iteration, the $A_{virtual}(h_j)$ is updated if we have larger real area for the huddle h_j. However the $A_{virtual}(h_j)$ is not updated if we have the same or smaller real area for the huddle h_j. Figs. 2 (e) and (f) show how to calculate virtual area.

Overall, the virtual area of the huddle h_j is estimated in the iteration process when its huddle construction is changed as follows:

1. $A_{real}(h_j)$ is calculated by summng up the areas of functional units, registers, a controller, and level converters inside h_j.
2. If $A_{virtual}(h_j) \geq A_{real}(h_j)$, $A_{virtual}(h_j)$ is not updated.
3. If $A_{virtual}(h_j) < A_{real}(h_j)$, we set $A_{virtual}(h_j) = A_{real}(h_j)$.

2.2 Virtual area adaptation

Virtual area estimation, however, may increase interconnection delays between huddles as the iterations proceed. To solve this problem, we should decrease the difference between virtual area and real area.

DOI: 10.1587/elex.9.1414
Received August 09, 2012
Accepted August 20, 2012
Published September 12, 2012

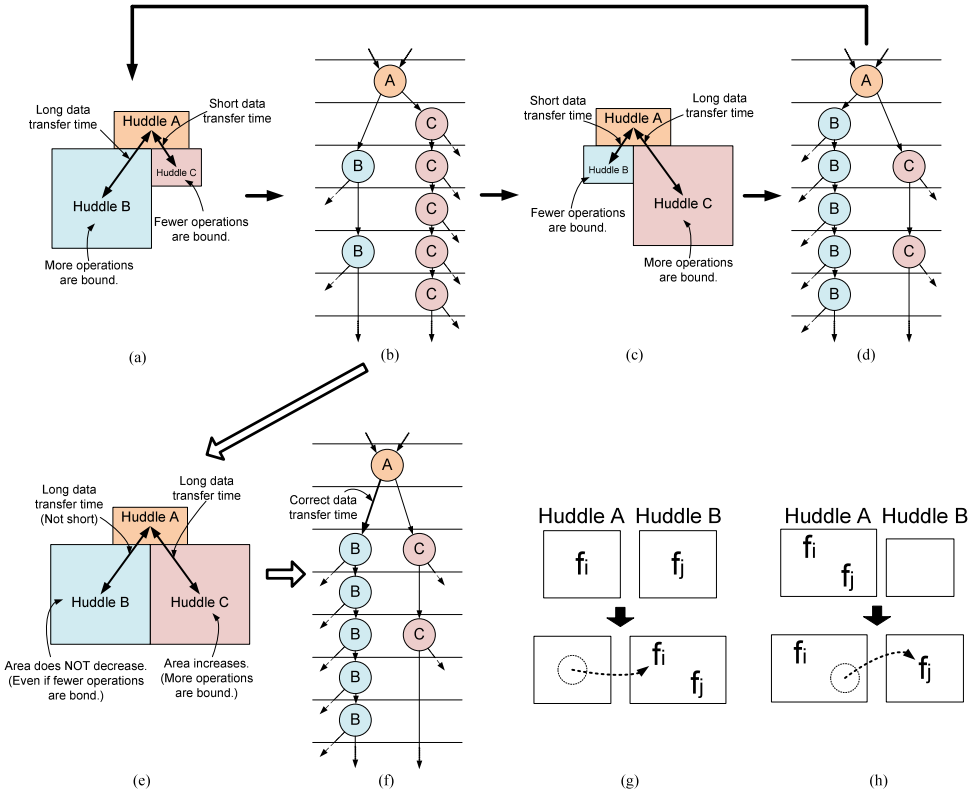

Fig. 2. Suppose that we have the floorplan (a) at the $(i-1)$-th iteration, where too many operations are bound to the huddle B and we require too much data transfer time between the huddles A and B. Based on (a), we can have the scheduling/binding (b) at the i-th iteration. Since the data transfer between A and C requires shorter time in (a), many operations are bound to C this time. Based on (b), we can have the floorplan (c) at the i-th iteration. It can not satisfy the timing constraint between A and C since too many operations are bound to C and C becomes too large instead of B. Based on (c), we can have the scheduling/binding (d) at the $(i+1)$-th iteration. When we have the floorplanning result based on (d), we will go back to (a) and these steps may be repeated.

On the other hand, MH[4] obtains the floorplan (e) at the i-th iteration based on (b). Area estimation $A_{virtual}(C)$ of C is updated but area estimation $A_{virtual}(B)$ of B is not. Based on (e), we can have the scheduling/binding (f) at the $(i+1)$-th iteration in MH[4]. Based on (f), we can also have the same floorplanning result as (e) which means the convergence of our algorithm.

In floorplanning-directed huddling, we only consider huddle *transfer*. In (g), huddle *merge* is realized by transferring the functional unit f_i from the huddle A to the huddle B. In (h), huddle *partition* is realized by transferring the functional unit f_j from the huddle A to the huddle B.

We execute *virtual area adaptation* after floorplanning-directed huddling. Because this step is just before scheduling/binding at the next iteration, we can use virtual areas closer to real areas at the next iteration.

DOI: 10.1587/elex.9.1414
Received August 09, 2012
Accepted August 20, 2012
Published September 12, 2012

Virtual area adjustment is executed as follows:

1. Let $A_{dif}(h_j) = A_{virtual}(h_j) - A_{real}(h_j)$ be the difference between real area $A_{real}(h_j)$ and virtual area $A_{virtual}(h_j)$ of the huddle h_j.
2. We set $A_{virtual}(h_j) = A_{real}(h_j) + \phi \cdot A_{dist}(h_j)$.

where ϕ is an adaptation parameter. In order to decrease ϕ as the iterations proceed, we set $\phi = 1 - 0.09i$ at the i-th iteration.

Note that the virtual area estimations are returned to their real area in the adjustment process.

2.3 Floorplanning-directed huddling

Virtual area estimation may cause vacant huddles which no functional unit is assigned to but has a virtual area. By effectively using vacant huddles, all of the three huddle construction methods *merge*, *partition*, and *transfer* can be represented by just using *transfer* (Figs. 2 (g) and (h)).

As pointed out in [1], huddle construction correlates with floorplanning. It is better for us to integrate huddle construction methods into floorplanning. In floorplanning, huddle placement as well as its height and width is optimized by using a simulated annealing (SA) strategy based on a sequence-pair representation. In this step, we consider the four moves as follows:

Move 1: Select two elements and exchange them in Γ_+.

Move 2: Select two elements and exchange them in Γ_+ and Γ_-.

Move 3: Select one element and change its aspect ratio.

Move 4: Select functional unit f_i and transfer it from the huddle h_j to the huddle $h_k (\neq h_j)$.

In the SA optimization, its cost function is the same as [1].

3 Experimental results

We have implemented the proposed algorithm in C++. The algorithm has been applied to DCT (48 nodes), EWF3 (102 nodes), FIR filter (75 nodes), and COPY (378 nodes, including conditional branches) where COPY is a practical application example. We used the same functional units and level converters as in [1]. Selectable voltages were assumed to $v_l = 0.8\,\mathrm{V}$, $v_m = 1.0\,\mathrm{V}$, and $v_h = 1.2\,\mathrm{V}$. Controllers inside huddles were synthesized by Synopsys Design Compiler in each iteration. The interconnection delays were assumed to be a proportion to square of the wiring length and an interconnection delay is set to be 1 ns when wiring length is 250 μm [1]. Energy consumption is obtained using Synopsys Design Compiler.

We have compared our proposed algorithm ("MH^4" in Table I) to the GDR architecture synthesis algorithm [8] ("GDR" in Table I), MCAS for RDR architectures [2] ("RDR" in Table I), our original algorithm targeting

DOI: 10.1587/elex.9.1414
Received August 09, 2012
Accepted August 20, 2012
Published September 12, 2012

Table I. Experimental results.

App.	FUs	Clock [ns]	Steps	Architecture and algorithm	ϕ	Rectangular area [μm^2]	All energy [pJ]	CPU time [sec]	Iterations
EWF3	Add×3 Mul×2	1.5	65	GDR	–	42432	476.70	830.43	24
				RDR	–	78400	537.74	105.35	1
				HDR [1]	–	50445	473.72	401.25	2
				MHDR [1]	–	47817	403.12	480.73	2
				MH^4 (Single)	$1-0.09i$	45034	487.60	344.74	2
				MH^4	$1-0.09i$	48399	404.36	357.50	2
FIR	Add×3 Mul×3	1.5	30	GDR	–	22165	198.52	2597.54	24
				RDR	–	99225	241.89	191.02	1
				HDR [1]	–	41856	247.04	635.23	2
				MHDR [1]	–	40040	178.34	725.91	2
				MH^4 (Single)	$1-0.09i$	25992	197.09	523.97	2
				MH^4	$1-0.09i$	37856	171.91	503.32	2
DCT	Add×4 Mul×4	1.5	15	GDR	–	64925	138.07	1338.14	24
				RDR	–	96800	181.25	191.41	1
				HDR [1]	–	60456	164.74	726.11	2
				MHDR [1]	–	65565	129.01	1372.02	4
				MH^4 (Single)	$1-0.09i$	57912	164.74	559.91	2
				MH^4	$1-0.09i$	60060	135.86	495.07	2
COPY	Add×3 Sub×1 Comp×1 Rshift×2 AND×1 Mul×5	1.5	170	HDR [1]	–	–	–	> 1 day	–
				MHDR [1]	–	–	–	> 1 day	–
				MH^4 (Single)	1	338976	8699.10	1900.19	7
					$1-0.09i$	433246	7805.45	2179.72	6
				MH^4	1	402992	3949.53	3618.05	9
					$1-0.09i$	325420	3307.97	2501.39	4
		5.5	90	HDR [1]	–	355320	5714.42	2283.05	4
				MHDR [1]	–	414080	2840.63	2531.08	4
				MH^4 (Single)	$1-0.09i$	374490	5542.46	1091.08	2
				MH^4	$1-0.09i$	336432	2804.23	1822.92	4

HDR architectures [1] with a single supply voltage ("HDR [1]" in Table I), our original algorithm targeting HDR architectures [1] with multiple supply voltages ("MHDR [1]" in Table I) and our proposed algorithm with a single supply voltage ("MH^4 (Single)" in Table I).

The experimental results show that all energy consumption of MH^4 is reduced by a maximum of 30.41% and an average of 18.24% compared with the other algorithms applied to single supply voltage. The CPU time of MH^4 and MH^4 (Single) are reduced by a maximum of 63.92% and an average of 29.11% compared with HDR [1] and MHDR [1]. MH^4 and MH^4 (Single) can obtain a feasible result for COPY which cannot be obtained by HDR [1] and MHDR [1]. All energy consumption in COPY using MH^4 is reduced by a maximum of 57.61% compared with MH^4 (Single). HDR [1] and MHDR [1] can obtain a feasible result for COPY when the clock period constraint was set to be 5.5 ns. In this case, however, all huddles are assigned to 0.8 V. Thus, there is no need to apply multiple supply voltages.

We verified the effectiveness of the adaption parameter ϕ in COPY because the results of them have several iterations and may have large virtual area overheads. The experimental results show that all energy consumption of $\phi = 1 - 0.09i$ is reduced by a maximum of 16.24% and an average of 13.26% compared with that of $\phi = 1.0$ which did not execute virtual area adaptation.

4 Conclusion

In this paper, we propose a multiple-supply-voltages aware high-speed and high-efficiency high-level synthesis algorithm for HDR architectures. Our proposed algorithm reduced energy consumption by an average of 18.24% compared with the single-supply-voltage aware algorithms and reduced CPU times by an average of 29.11% compared with our original algorithm targeting HDR architectures. Our proposed algorithm can successfully obtain high-

DOI: 10.1587/elex.9.1414
Received August 09, 2012
Accepted August 20, 2012
Published September 12, 2012

level synthesis solution which cannot be obtained by our original algorithm.

Acknowledgments

This work was supported partially by "Grant for Advanced Industrial Technology Development" from the New Energy and Industrial Technology Development Organization (NEDO) of Japan.

DOI: 10.1587/elex.9.1414
Received August 09, 2012
Accepted August 20, 2012
Published September 12, 2012