A-03

音声認識システムにおける低消費電力化設計

A power aware design for voice recognition system

重谷 歩美†　　神戸 尚志††
Ayumi Omotani　Takashi Kambe

1. はじめに

現在、システム LSI は集積技術の向上により高性能化が進んでおり、それに伴い消費電力の増大が問題となっている。低消費電力化が必要とされる理由の一つとしては携帯情報機器の出現があげられる。携帯情報機器ではマルチメディア処理や通信機能の実現の必要があるため高性能であること、携帯するため軽い必要があり、電池を大きくできないことから、バッテリーが長持ちすることが要求される。もうひとつの理由として、電力は熱の形で消費され、誤作動の原因となる可能性があり、低消費電力設計は極めて重要である。本研究では携帯機器への搭載を目的として大語彙連続音声認識における出力確率計算回路設計に対して C 言語設計による低消費電力 LSI 手法を提案し、消費電力・処理時間の両面から最適化を行う。

2. LSI の電力消費のメカニズム

システム LSI のような集積回路は一般的に CMOS 回路が用いられている。その平均電力は 3 つの要素からなり、消費電力は式(1)のように表される。

$$P=P_{dynamic}+P_{SC}+P_{leak} \quad (1)$$

$P_{dynamic}$ はダイナミック・パワーと呼ばれる電力であり、動作中の CMOS 回路の出力が 0→1 または 1→0 に遷移する際に消費される電力で、式(2)のように表される。

$$P_{dynamic}=C\cdot V_{DD}^{2}\cdot f_{CLK}\cdot p_t \quad (2)$$

ここで、C は負荷容量、V_{DD} は電源電圧、f_{CLK} は動作周波数、p_t はスイッチング確率である。

P_{SC} はショートサーキット・パワーと呼ばれる電力であり、電源 VDD と GND が導通した際に pMOS と nMOS に瞬間的に流れる貫通電流により消費される電力であり、式(3)のように表される。

$$P_{SC}=p_t\cdot I_{sc}\cdot \Delta t_{SC}\cdot V_{DD}\cdot f_{CLK} \quad (3)$$

ここで I_{SC} は貫通電流の平均値、Δt_{SC} は貫通電流の流れる時間である。

P_{leak} はリーケージ・パワーと呼ばれる電力であり、リーク電流(漏れ電流)によって消費される電力で、式(4)のように表される。

$$P_{leak}=I_{leak}\cdot V_{DD} \quad (4)$$

このリーク電流には PN 接合に逆バイアスが印加された場合に発生するジャンクション・リーク I_{junk}、MOSFET がオフの状態で Source- Drain 間に流れるサブスレッショルド・リーク I_{sub}、薄い Gate 酸化膜を通してトンネル効果によって電流が流れるゲート・リーク I_{gate}、や Gate 電極の Drain 端に高い電界がかかることによって Drain から基盤へ流れる GDIL(Gate Induced Drain Leakage)IGDIL などがある。

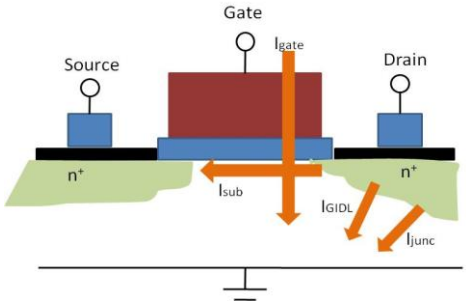

図 1　電力消費のメカニズム

3. 低消費電力設計手法

CMOS 回路が消費する 3 種類の電力の中でも、回路動作時にはダイナミック・パワーが最も大きく、全体の 85%～90%を占めている。ダイナミック・パワーの低減には式(2)から、電源電圧 V_{DD} の低減が最も効果的である。しかし、処理速度の低下につながる危険性がある。動作周波数についても同様で、処理速度の低下につながる場合がある。また、負荷容量については微細化によって効果的に低減することができるが、微細化には製造段階における大きなコストアップや技術力が必要となる。そのため負荷容量 C とスイッチング確率 p_t を考え、大きなスイッチング確率を持つ回路のスイッチング動作を削減することによって平均的に諸費電力を低減する方法が考案されており、ゲーテッド・クロック、オペランド・アイソレーション、プリ・コンピュテーションなどの手法がある。

ゲーテッド・クロックとはクロック信号に AND ゲート、または OR ゲートを挿入することによってクロックの遷移が不要な場合にクロックを積極的に停止させる方法である。実際に設計された動画処理に用いられる MPEG4 チップを解析した例では、クロックとフリップフロップ(F/F)による消費電力が 48%を占めていると報告されている [1]。このことからクロックとフリップフロップの消費電力を低減する方法としてゲーテッド・クロックは有効である。

本文では C 言語設計におけるゲーテッド・クロックによる低消費電力化について考える。

4. C 言語設計

C 言語設計では動作合成によって RT レベル回路を生成するため、回路のハードウェアアルゴリズムのみを記述し、タイミングなど回路動作の詳細を記述する必要がない。このため、システム LSI のアルゴリズムを自然に記述するこ

†近畿大学大学院　総合理工学研究科　エレクトロニクス系工学専攻
††近畿大学　理工学部　電気電子工学科

とができ、高位合成により様々なアーキテクチャを短時間に生成するので、ハードウェア設計を大幅に効率化できる。また、従来の C 言語で記述されていたアルゴリズム設計資産の再利用も容易となっている。

C 言語設計システムは一般にハードウェア記述用に拡張された C 言語と、その記述から RT レベル回路を自動生成する動作合成ツール、検証・デバッグを補助するツールなどからなり、実用的な LSI 設計が可能となっている。また、ハードウェア記述用に拡張された C 言語は変数宣言時に変数ビット幅を指定することや、並列動作を明示的に記述することができる。本文ではシャープ株式会社の提供のもと、Bach システムを用いる。

5. 消費電力解析手法

消費電力の解析には RT レベル、または、ゲートレベルでの解析を行う Synopsys 社の Power Compiler を使用する。

5.1 電力解析フロー

Power Compiler による電力解析フローを図 2 に示す。

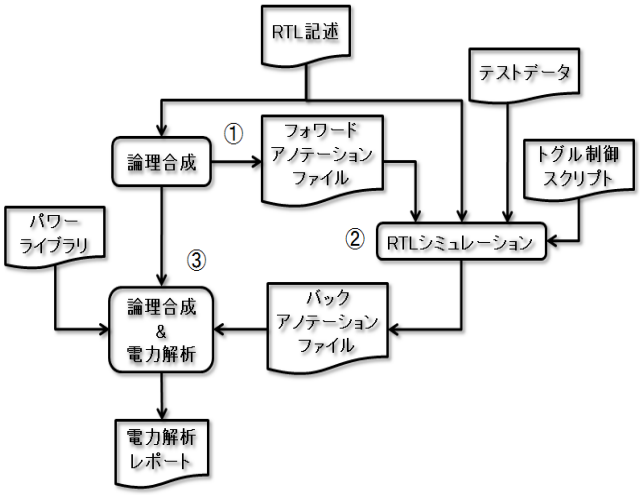

図 2　Power Compiler による電力解析

図 2 の①の部分では RTL 記述から論理合成を行い、フォワードアノテーションファイルを生成する。フォワードアノテーションには作成条件、環境などのファイル情報とインスタンス回路ごとのポート名が記載されている。

RTL シミュレーション②にフォワードアノテーションファイルをロードすることで、トグル制御スクリプトに従いシミュレーションを行い、バックアノテーションファイルに出力する 。バックアノテーションファイルではシミュレーションによって検出された論理値 0 の期間(T0)、論理値 1 の期間(T1)、不定”X”の期間(TX)、信号の立ち上がり(0→1)および立ち下り(1→0)の合計回数(TC)、グリッジ(0→X→1 or 1→X→0)の回数(IG)が各ポートごとに記述されている。

最後にパワーライブラリの特性と、バックアノテーションファイル中のスイッチング情報により消費電力計算③を行い、電力解析レポートを出力する。

5.2 消費電力計算

Power Compiler で求めるダイナミック・パワーはスイッチングパワーとインターナルパワーの 2 種類がある。

5.2.1 スイッチングパワー

スイッチングパワーとは動作セルの出力が遷移した際にその負荷容量が充電と放電をすることによって消費される電力を表しており、その計算式 P_c を式(5)に示す。

$$P_c = \frac{V_{DD}^2}{2} \sum_{\forall nets(i)} (C_{Loadi} \times TR_i) \qquad (5)$$

ここで、C_{Load} は回路内のネットに接続されるピンの寄生容量、ゲート容量、ドレイン容量などを含んだ総負荷容量であり、TR_i はネット i のトグル率(秒あたりの遷移率)である。

5.2.2 インターナルパワー

インターナルパワーとはセルの内部で消費される電力である。セルの内部に存在しているキャパシタンスがスイッチングの際に充電、放電をすることによって電力を消費する。また、ショートサーキット・パワーもこのインターナルパワーに含まれている。インターナルパワーの計算式を式(6)に示す。

$$P_{Int} = E_Z \times TR_Z \qquad (6)$$

ここで、E_Z は、入力遷移や出力負荷などで求められるセルの出力 Z の内部エネルギー、TR_Z はセルの出力ピン Z のトグル率である。

6. 音声認識システム

我々は、携帯機器への搭載を目的に音声認識システムの低消費電力化を図っている。本文では特に大語彙連続音声認識システム Julius における出力確率計算部についての低消費電力化を考える。

6.1 大語彙連続音声認識システム Julius

Julius とは京都大学、奈良先端科学技術大学院大学等で開発された音声認識システムである。以下に Julius の概要を示す。

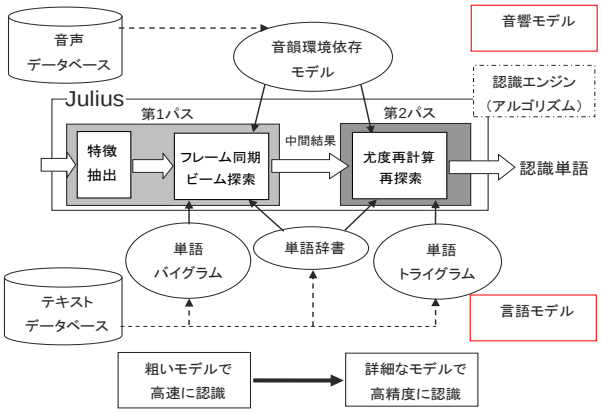

図 3　大語彙連続音声認識システム Julius

第一パス処理は、音素 HMM から状態遷移・出力確率値を求め、候補文を求める。第二パス処理は、第一パス処理で得た文の候補を、再探索・再評価し最終的な文の候補が出力する。

6.2　出力確率計算

Julius で使用する HMM(隠れマルコフモデル)は、信号系列に対して定常と見なす「状態」を定義し、状態から状態へ遷移する確率（状態遷移確率）と状態ごとに信号が観測される確率（出力確率）が計算する。音声認識におけるフレーム単位の音響特徴量は、p 次元ベクトルで表現され、音響特徴量の確率分布は複数のガウス分布を組み合わせた混合ガウス分布で表現され、出力確率は以下の計算で求める。p 次元、t 番目フレームの音響特徴量ベクトルを o_{tp} とし、i 状態 m 混合目のガウス分布計算 b_{im} は、式(7)で表される。

$$\ln b_{im}(o_t) = \omega_t - \frac{1}{2}\sum_{p=1}^{P}\frac{1}{\sigma_{imp}^2}(o_{tp} - \mu_{imp})^2 \qquad (7)$$

ここで、ω_i を混合重み値, μ_{imp} を平均ベクトル, σ^2_{imp} を分散ベクトルとし、HMM の学習時に求める。M 混合ガウス分布(指数対数計算)の出力確率計算値 b_i は、式(8)で表される。

$$\log b_i(o_t) = \log\sum_{p=1}^{P} b_{im}(o_t) \qquad (8)$$

この 2 つの式を使用して、出力確率計算を行う。

出力確率計算は計算処理が多く、並列化による効率の上昇が望め、ソフトウェア部との通信量が少ないことから出力確率計算部のハードウェア化を行なう。

7.　出力確率計算回路

ハードウェアは演算回路を並列に動作させることによって高速に動かすことができる。並列動作を行うためには回路の内部でおいて同期をとる必要がある。回路を逐次的に動作させる場合においても外部との通信によって同期を取り、値の受け渡しを保証する。出力確率計算回路は逐次処理で、値を読み込み、計算を行った後に結果が出力される。

この処理において、演算がおこなわれる前に音声パラメータはレジスタに格納されている。出力確率計算のソフト・ハード構成、通信方法を図 4 に示す。

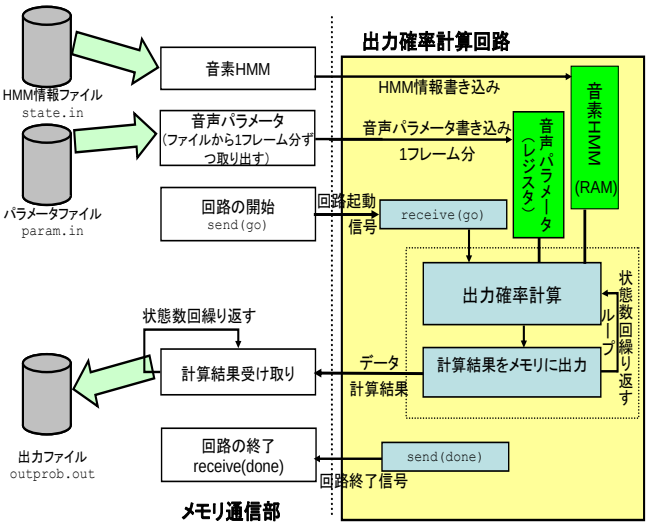

図 4　出力確率計算回路

本文ではこの出力確率計算回路に対して並列化、パイプライン化とその低消費電力化を行なう。

7.1　並列処理

HMM 情報の異なる状態における処理は独立しているため、複数の状態を並列に動作させることが出来る。出力確率計算回路には 4 つの混合が存在するため、それらを並列処理する（図 5）。指数対数計算は 4 混合分のガウス分布計算結果が必要となるためシーケンシャルに処理する。

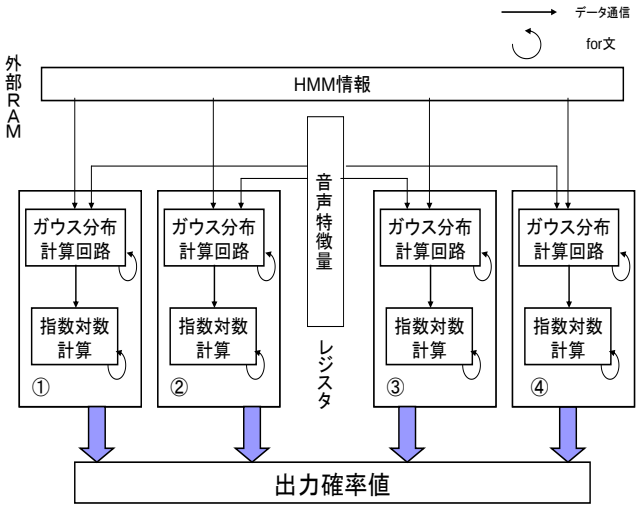

図 5　4 並列処理

また、この 4 並列処理のタイミングチャートを図 6 に示す。

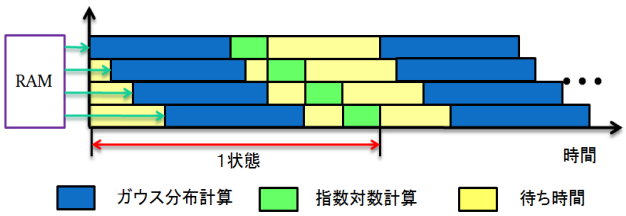

図 6　4 並列処理タイミングチャート

7.2　パイプライン処理

ガウス分布計算回路と指数対数計算回路のパイプラインを混合数分用意することで並列パイプライン処理を行う。4 混合の場合を図 7 に表す。

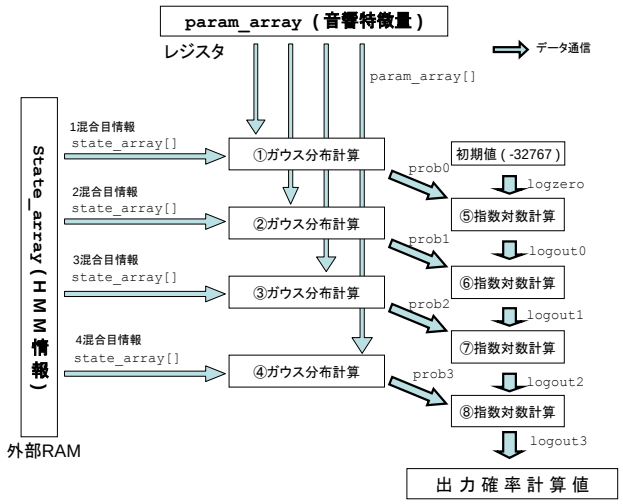

図 7　パイプライン処理

また、このパイプライン処理のタイミングチャートを図 8 に示す。

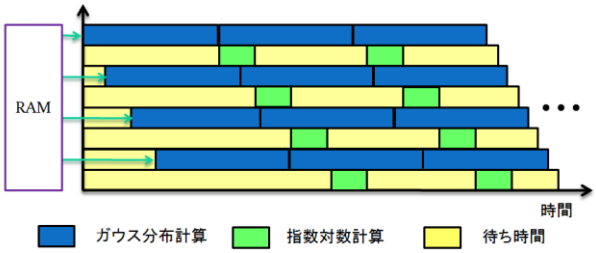

図 8　パイプライン処理タイミングチャート

7.3　メモリ分割

図 6、図 8 からわかるように、HMM 情報のメモリからの読み出しに待ち時間が生じている。メモリを 4 つに分割し、HMM 情報を読み込むための待ち時間を解消する。タイミングチャートを図 9 に示す。

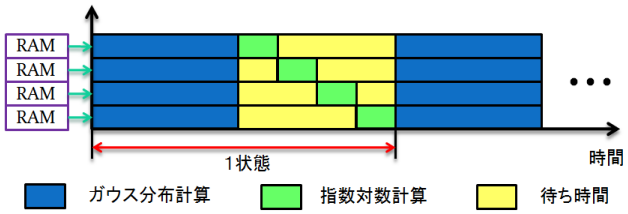

図 9　メモリ分割の 4 並列回路のタイミングチャート

7.4　ループ展開

次元数分の計算には for 文を用いている。これを展開することによる計算の並列を図る（図 10）。しかし、この並列化は回路の規模が増大するデメリットも持つ。

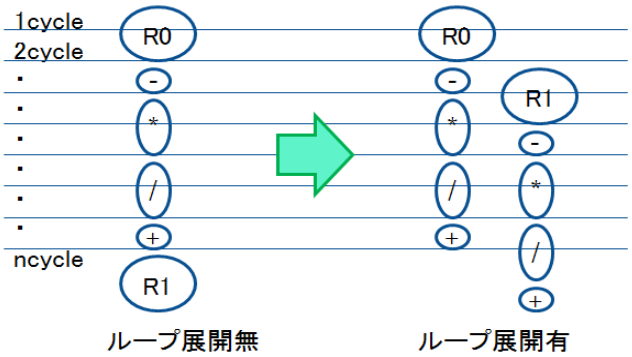

図 10　ループ展開によるサイクルタイミング

7.5　ガウス分布計算式変形

ガウス分布の式(7)を下式(9)のように、回路規模増大や計算時間増の原因となる除算をなくすよう変形する。乗算は除算に比べて回路ゲートが 4000 ゲート少なく、処理時間も約 11cycle 早く処理できる。ガウス分布計算における必要なパラメータは予め学習データにより求められるため、ガウス分布計算の分散値としてその逆数を用いる。

$$\ln b_{im}(o_t) = \omega_t - \frac{1}{2}\sum_{p=1}^{P} \sigma'^{2}_{imp}\left(o_{tp} - \mu_{imp}\right)^2 \qquad (9)$$

ただし

$$\left(\sigma'^{2}_{imp} = \frac{1}{\sigma^{2}_{imp}}\right)$$

8. 出力確率計算回路の低消費電力設計結果

これまで述べた、出力確率計算回路を 4 並列処理 RAM 分割＋ループ展開＋ガウス分布計算式変形したものと、パイプライン処理 RAM 分割＋ループ展開＋ガウス分布計算式変形したものの回路規模、処理時間、消費電力を表 1 に示す。

表 1　ゲーテッド・クロック非搭載回路の設計結果

回路名	回路規模 (gate)	処理時間 (ns)	消費電力(mW)		
			インターナルパワー	スイッチングパワー	合計
4並列処理	213,917	683,290	79.624	155.511	235.135
パイプライン処理	247,475	607,450	73.126	143.486	216.612

ゲーテッド・クロック化は図 9 のようにシーケンシャルで処理を行う指数対数計算の前後に待ち時間が存在することから、4 並列回路については 4 並列すべてにゲーテッド・クロック化を適用し、パイプライン回路については指数対数部分にのみに適用した。表 2 にゲーテッド・クロック搭載回路の回路規模、処理時間、消費電力を示す。

表 2　ゲーテッド・クロック搭載回路の設計結果

回路名	回路規模 (gate)	処理時間 (ns)	消費電力(mW)		
			インターナルパワー	スイッチングパワー	合計
4並列処理	208,766	683,290	72.508	61.205	133.713
パイプライン処理	205,085	607,450	66.602	52.891	119.493

また、表 3、4 に 4 並列処理についてゲーテッド・クロック搭載のものと、非搭載のものの回路規模と消費電力について内部回路ごとの内訳を示す。

表 3　4 並列処理回路の回路規模内訳

	回路規模 (gate)	
ゲーテッド・クロック	非搭載	搭載
同期通信	6600	7301
TH1	52265	50843
TH2	51496	50194
TH3	51692	50221
TH4	51865	50207
gclk_1		9
gclk_2		7
gclk_3		7
gclk_4		7
合計	213917	208766

表 4　4 並列処理回路の消費電力内訳

	消費電力(mW)					
ゲーテッド・クロック	非搭載			搭載		
	インターナル	スイッチング	合計	インターナル	スイッチング	合計
同期通信	0.373	1.440	1.863	0.385	0.394	0.778
TH1	19.605	38.286	59.847	17.848	13.910	31.758
TH2	19.559	37.878	58.147	17.862	13.899	31.761
TH3	19.780	38.366	57.437	17.666	13.757	31.423
TH4	20.307	39.541	57.841	18.640	14.506	33.146
gclk_1				0.025	1.226	1.251
gclk_2				0.028	1.180	1.208
gclk_3				0.028	1.171	1.199
gclk_4				0.028	1.162	1.190
合計	79.624	155.511	235.135	72.508	61.205	133.713

また、パイプライン回路についてゲーテッド・クロック搭載のものと、非搭載のものの回路規模と消費電力内訳を表 5、6 に示す。TH1 はパイプライン処理を行うために、タイミングを図っている。ここで、TH2 は混合数分、ガウス分布計算とその結果の加算を行う回路である。TH3、TH5、TH7、TH9 はガウス分布計算、TH4、TH6、TH8、TH10 は指数対数計算の回路である。

表 5　パイプライン処理の回路規模内訳

	回路規模 (gate)	
ゲーテッド・クロック	非搭載	搭載
同期通信	5,982	7,020
TH1	178	177
TH2	2,192	2,173
TH3	31,033	28,644
TH4	28,524	20,274
TH5	31,803	28,644
TH6	27,863	20,262
TH7	31,731	28,644
TH8	28,175	20,267
TH9	31,630	28,643
TH10	28,365	20,289
gclk4		12
gclk6		12
gclk8		12
gclk10		12
合計	247,475	208,766

表 6　パイプライン処理の回路消費電力内訳

	消費電力(mW)					
ゲーテッド・クロック	非搭載			搭載		
	インターナル	スイッチング	合計	インターナル	スイッチング	合計
同期通信	0.726	2.084	2.855	0.437	0.714	1.152
TH1	0.003	0.003	0.009	0.002	0.002	0.004
TH2	0.293	0.446	0.739	0.290	0.167	0.458
TH3	17.622	34.325	51.827	16.043	12.684	28.726
TH4	0.293	0.767	1.143	0.274	0.232	0.506
TH5	17.622	34.277	51.827	16.113	12.721	28.834
TH6	0.376	0.860	1.284	0.313	0.259	0.571
TH7	17.699	34.404	52.104	16.180	12.745	28.925
TH8	0.433	0.884	1.327	0.325	0.269	0.594
TH9	17.608	34.519	52.128	16.179	12.751	28.930
TH10	0.451	0.917	1.369	0.335	0.276	0.611
gclk_4				0.028	0.018	0.046
gclk_6				0.028	0.018	0.046
gclk_8				0.028	0.018	0.046
gclk_10				0.028	0.018	0.045
合計	73.126	143.486	216.612	66.602	52.891	119.493

まず、回路規模については、表 3、表 5 で示すように、ゲーテッド・クロック回路が挿入されているにも関わらず、回路規模が減少した。4 並列処理回路では 5151[gate]、パイプライン処理回路では 38709[gate]の現象である。ゲーテッド・クロック搭載回路はゲーテッド・クロック用回路と同期通信リソースが増加したが、その他は減少した。これは Synopsys 社 Design Compiler の論理合成時に最適化され、非搭載の場合に存在していた MUX などがゲーテッド・クロック回路を搭載することによって削除されている可能性があることが原因として考えられる。

次に、処理時間に関して述べる。音声認識システムでは一定の時間幅で切り出したものをフレームとし、このフレームを 1 つの処理単位としている。ここでは 1 フレーム分

の処理時間を示す。ゲーテッド・クロックの搭載・非搭載に関わらず同じ処理時間となった。ゲーテッド・クロックは回路の性能に影響を及ぼすものではないことがわかる。

最後に消費電力について述べる。4 並列処理回路ではゲーテッド・クロックを搭載したものは非搭載のものと比べて 43%減少した。パイプライン処理回路では 44%減少した。表 4、表 6 から 4 並列処理回路のインターナルパワーは 8.9%、スイッチングパワーは 61%減少した。パイプライン処理回路ではインターナルパワーは 8.9%、スイッチングパワーは 63%減少した。スイッチングパワーの減少については、ゲーテッド・クロックによると考えられる。また、インターナルパワーや、ゲーテッド・クロック化を行っていないガウス分布計算の消費電力も減少している理由としては、不要な信号の伝播による無駄な信号の遷移が指数対数計算部分のゲーテッド・クロック化によって減少しているためと考えられる。

9. まとめ

本研究では高速指向の出力確率計算に対してゲーテッド・クロック化を施し、低消費電力化を行った。リアルタイム音声認識システムを実現するためには 10ms の処理速度が要求される。4 並列処理回路が 0.68ms、パイプライン処理回路が 0.60ms の処理時間となり、この要求を満たす可能性がある。消費電力については 4 並列処理回路では 43%の削減、パイプライン処理回路では 44%削減できたが、携帯端末搭載のためにさらなる低消費電力化の手法を考える。

参考文献

[1] 宇佐美公良: “STARC LSI 設計編(B コース) B8 章 低消費電力設計 1 ”、STARC,2008

[2] 鹿野清宏、伊藤克亘、河原達也、武田一哉、山本幹雄:“音声認識システム”、オーム社,2002

[3] 桜井貴康編: “低消費電力、高速 LSI 技術”、リアライズ社 ,1998

[4] K. Okada, A. Yamada, and T. Kambe: "Hardware Algorithm Optimization Using Bach C", IEICE Trans. Fundamentals vol.E85-A, No.4, pp835-841, 2002.

[5] “Power Compiler User Guide”、Synopsys 社提供 ,2006

[6] A.Eguchi, J.Hashimoto, M.Saituji, A.Yamada, T.Kambe; "A Hardware Design for the First Pass of A Large Vocabulary Continuous Speech Recognition System," The proceeding of 15th SASIMI, pp. 230 - 235,2009.

[7] A. Eguchi, K. Kishida, T. Kambe, M. Saituji,"An Application Specific Circuits Design for a LVCSR System," The proceeding of ECCTD09.

[8] T. Anantharaman and B. Bisiani, "A hardware accelerator for speech recognition algorithms," Proceedings of the 13th Annual Intl. Symposium on Computer Architecture,pages 216-223, 1986.

[9] S. Chatterjee and P. Agrawal, "Connected speech recognition on a multiple processor pipeline, " volume 2, pages 774-777, May 1989.

[10] H. Hon, "A survey of hardware architectures designed for speech recognition, "Technical Report CMU-CS-91-169,August 1991.

[11] S. Kaxiras, G. Narlikar, A. Berenbaum, and Z. Hu, "Comparing Power Consumption of an SMT and a CMP DSP for Mobile Phone Workloads, " International Conference on Compilers, Architectures, and Synthesis for Embedded Systems (CASES), November 2001.

[12] M. Ravishankar, "Parallel implementation of fast beam search for speaker-independent continuous speech recognition, "Computer Science and Automation, 1993.