システムＬＳＩ設計技術　108－21
(2003. 1. 29)

FPGAを用いた細胞シミュレーションシステムの構築

長名　保範†　　福島　知紀†　　天野　英晴†

†慶應義塾大学大学院理工学研究科
〒223–8522 横浜市港北区日吉 3–14–1
E-mail: †bio@am.ics.keio.ac.jp

あらまし　近年の急速な分子生物学の進展に伴い、細胞内の代謝活動を定量的なモデルで表現し、シミュレーションを行い、その結果と実験結果を比較することでより詳細に細胞のはたらきを理解するという研究方法が可能になりつつある。しかしながら、細胞シミュレーションに必要な計算量は大きく、将来、より精密なシミュレーションが行われるようになった場合には計算時間が現在よりもさらに深刻な問題になると考えられる。本研究報告では、FPGAとホストPCの協調処理によって高速なシミュレーションを行うための基盤システムであるReCSiPを提案する。初期評価の結果、ReCSiPは、ホストPC上のソフトウェアによる処理に比べて約8倍の速度向上を達成した。
キーワード　細胞シミュレーション, 代謝系モデル, reconfigurable hardware, ReCSiP

Design of a Cell Simulation System with FPGA

Yasunori OSANA†, Tomonori FUKUSHIMA†, and Hideharu AMANO†

† Graduate School of Science and Technology, Keio University
Hiyoshi 3–14–1, Yokohama-shi, Kanagawa, 223–8522 Japan
E-mail: †bio@am.ics.keio.ac.jp

Abstract　The rapid progress of molecular biology in 1990s made simulations of metabolism possible. By modeling and simulating metabolism in living cells, biologists will be able to understand more details of metabolic processes. However, the computational cost for cell simulation often becomes large, and simulations tend to take much longer time than experiments. In this research, a simulation accelerator with FPGA is proposed and evaluated with a simple example. The accelerator system "ReCSiP" consists of a PCI board with FPGA and software library for controling the board. Initial evaluation results show that the performance of ReCSiP is about 8 times that of the software on the host PC.
Key words　Cell simulation, metabolic model, reconfigurable hardware, ReCSiP

1. はじめに

80年代後半から90年代のはじめにかけて、Gepasi [1], DBsolve [2] をはじめとして、代謝系のモデルを解析するためのソフトウェアがいくつか開発されてきた。90年代後半に入ると、計算機の急速な性能向上と、分子生物学の進展によってさまざまな定量的な測定が可能になったことから、実験データを用いて、細胞内の代謝システム全体を計算機上で再現しようとする試みが行われるようになった。これらの細胞シミュレータの例としては、E-Cell [3] や The Virtual Cell [4] を挙げることができる。

細胞をひとつのシステムとして捉え、シミュレーションする場合には、いくつもの問題点がある。たとえば、

（1）　数値モデル化するのに充分な実験データがない
（2）　計算精度と計算時間のトレードオフ
（3）　計算精度を落しても、必要な計算時間はかなり長い

といった点である。(1) は、生物学実験の進展を待つほかないが、(2) および (3) に関しては、計算機システムの改良により、現状を打開することが可能であると考えられる。

現在、細胞内の空間的広がりと物質の拡散を考慮するような、精密なモデルでのシミュレーションは実時間の数千から数万倍を所要しており、計算機の性能が数十倍になったとしても、あまり大きな効果が出ないということができる。一方で、分子動力学などの分野では、専用計算機が大きな成果をあげており、細胞シミュレーションの分野においても、何らかの計算用ハードウェアによって大きな計算能力の向上を果たすことの可能性は充分にあるといえよう。

しかしながら、細胞のシステムは、従来の計算科学が対象としてきた電子回路、流体力学、分子動力学などと比較するとはるかに計算対象が複雑かつ巨大 [5] であることと、シミュレー

ション対象の性質によって様々なアルゴリズムを用いるため、専用ハードウェアの開発は困難な面も存在する。

そこで我々はFPGAを用いた細胞シミュレーション支援のためのプラットフォームであるReCSiP(REconfigurable Cell SImulation Platform)を開発した。ReCSiPは、FPGAを搭載するPCIボードとそのデバイスドライバ、及びユーザプログラムからのボード利用を支援するためのツール群から構成され、ホストCPUとFPGA上の専用ハードウェアの協調処理により、複雑なアルゴリズムの高速な実行を実現する。

2. 細胞シミュレーション

細胞シミュレータには様々なものがあるが、多くは代謝反応のモデル化とシミュレーションを行い、細胞内に存在する各種の代謝物質個々について、濃度の時間変化を調べるためのソフトウェアである。

ここでは、代謝反応のモデルの例を挙げ、シミュレーションの高速化を実現する方式について考察する。

2.1 Michaelis-Menten モデル

Michaelis-Mentenモデルは、酵素反応を質量作用則を用いて反応速度式の形に表現したものである。このモデルでは、、基質S、酵素E、酵素-基質複合体SE、生成物Pの反応を式1のように定義する。k_1, k_2, k_3 は各反応の反応速度定数である。

$$S + E \underset{k_2}{\overset{k_1}{\rightleftarrows}} S \cdot E \xrightarrow{k_3} P + E \tag{1}$$

この式1を、個々の反応速度式に展開すると式2〜式5のようになる。

$$\frac{d[S]}{dt} = -k_1[S][E] + k_2[SE] \tag{2}$$

$$\frac{d[P]}{dt} = -k_3[SE] \tag{3}$$

$$\frac{d[E]}{dt} = -k_1[S][E] + (k_2 + k_3)[SE] \tag{4}$$

$$\frac{d[SE]}{dt} = k_1[S][E] + (k_2 + k_3)[SE] \tag{5}$$

これら4つの式を、タイムステップ毎に計算することで、各物質濃度の時間変化を追跡することができる。なお、式4および5からわかるように、$d[E] = -d[SE]$ であるので、この計算は片方だけでよい。

なお、ここに挙げた式は、定常状態付近での挙動を近似したものであり、必ずしも厳密ではないが、広く用いられている。

2.2 物質の移動モデル

シミュレータによっては、細胞の立体構造モデルを持ち、空間内での物質の移動を含めた計算を行う。これらのシミュレータは、細胞内の場所ごとに前述のMichaelis-Mentenモデルなどによって物質濃度の計算を行い、それを元にして濃度勾配や電位勾配を計算する。濃度や電位の勾配から、物質の種類ごとに拡散の方向と量を求めることで、細胞内の物質の移動をシミュレートすることが可能になる。

物質の移動を表す式を次に示す。

$$\vec{J} = -D\nabla C - z\mu C\nabla V, \quad \mu = \frac{DF}{RT} \tag{6}$$

D は拡散定数、C は濃度、z はイオン価数、μ はmobilityであり、それぞれ物質の種類ごとに異なる。右辺第1項は拡散による移動、第2項は電位勾配による移動である。

2.3 高速化手法の考察

現在、バイオインフォマティクスの分野では遺伝子配列の解析が盛んであり、さまざまな研究機関で大規模なPCクラスタが稼働し、成果をあげている。これは、遺伝子配列の解析という問題は主にsequence matchingであり、容易に並列化し、かつ高い並列度で実行することが可能であるためである。

しかしながら、代謝系のシミュレーションにおいては、

- 複数の反応に関与する物質が多く、その依存関係が極めて複雑であること
- 式6のように、計算の途中でシミュレーションする系全体に関するデータが必要となること

などから、並列化した場合には通信がボトルネックになってしまうことが明白であり、精密なシミュレーションではクラスタなどを用いた大規模な並列化は困難であることが予想される。

しかしながら、化学反応は互いに独立な事象として取り扱うことができ、シミュレーションを行うタイムステップごとにデータの同期がとれればよいので、問題そのものの並列性は充分に確保することができる。したがって、FPGAなどに複数の演算器やメモリを並べて処理することにより、高速化が見込めると考えられる。

3. シミュレーションプラットフォーム ReCSiP

3.1 ReCSiP の概観

現在多くのシミュレータは、Unix上でC/C++を用いて実装されており、SMPやクラスタなどを用いた並列処理によって高速化するアプローチが考えられる。しかし、既に述べたように細胞シミュレーションにおいては、タイムステップ毎にグローバルな濃度勾配や電位勾配を算出する必要があるために、クラスタなどを用いた場合には頻繁な同期操作が必要であり、細胞シミュレーションは並列化に速度向上が単純な方法では実現しにくいアプリケーションであるということができる。また、クラスタはコスト的にも設置場所的にも大きいため、どのような研究者でも気軽に利用できる性質の計算資源ではない。

これに対して、本研究で提案するのは、FPGAとメモリから構成されるボードを、PCIバスに接続して利用するアクセラレータとホストCPUの協調システムである。FPGA上のハードウェアとホストCPUで、それぞれ計算負荷の大きく比較的単純な処理と、複雑で計算負荷の小さい処理を分担して処理することによって、効率のよい処理を目指す。高速化の手法としては、FPGA上の多数の演算器による並列処理であるが、クラスタに比べ安価であり、通信処理によるオーバヘッドは存在しない。

3.2 ReCSiP ボードの構成

図1にボードの構成を示す。中心になるFPGAはXilinx社のVirtex-II (XC2V6000)で、600万ゲート相当の回路が実装可能である。その周辺に、並列アクセス可能な同期SRAMを4セットと、大容量のSDRAMを実装している。そのほかに、PCIイ

ンターフェイスとして QuickLogic 社の QL5064 を実装しており、64bit/66MHz の PCI バスに接続することが可能である。

ボードはホストからはメモリデバイスとして見えるようになっており、ボード上のメモリや、FPGA 上に実装した任意のレジスタをメモリ空間上に割り当てることが可能である。また、メインメモリとボード上のメモリ間でデータ転送を行うためにボードがマスタとなって DMA 転送を行うことができる。PCI バスと、ボード上のローカルバスのクロックは分離されており、ハードウェア設計者の負担も軽減されるように配慮されている。

3.3 ソフトウェアの構成

図 2 に ReCSiP を利用したソフトウェアを実行するまでの手順を示す。

ユーザプログラムからボードを呼び出す場合、一般にはドライバから取得したアドレスをを用いて `read()`, `write()` などの関数によってボードと通信したり、割り込み処理ルーチンなどを自前で用意する必要があり、ユーザにとっての開発効率が悪い。

ReCSiP では、簡単なスクリプトを用いることでボードを利用する関数を作成することができる。このスクリプトは、ReCSiP プリプロセッサによって、C 言語のコードに変換され、C/C++ から関数として呼び出すことができるようになる。

4. PCI バスの通信ボトルネックの解決

FPGA 上に回路を構成する場合、ホスト CPU との通信は PCI バスなどを介して行うのがリーズナブルな解決策であるが、この場合は必然的に PCI バスの転送速度の制約を受ける。したがって、この転送速度がボトルネックになる可能性がある。

この通信ボトルネックを回避するためには、PCIx や Infiniband に代表される最新のバスを用いる必要があるが、多くの PC ではまだ搭載されていないのが現状である。また、メモリスロットにボードを挿す手法 [6] も提案されているが、ボードの AC 設計が困難であることと、近年の PC に採用されるメモリシステムの交代サイクルが短くなっていることから、現在のところ設計面と互換性の面から採用が困難である。

そこでここでは、必要なデータ転送量について検証し、続いて PCI バスの転送能力を有効に利用する方法について考察する。

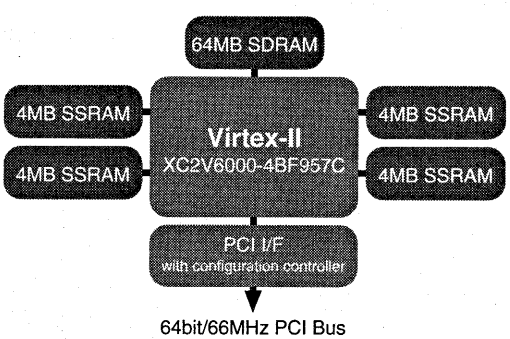

図 1 ReCSiP ボードの構成

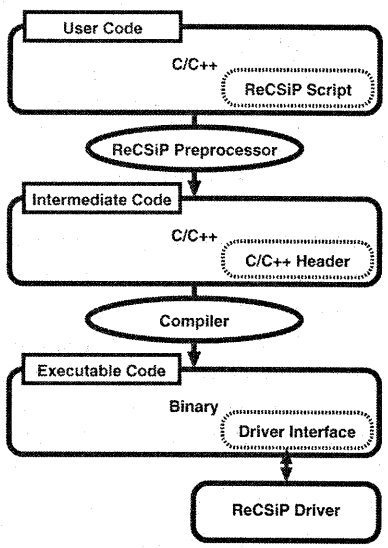

図 2 ReCSiP のソフトウェア構成

4.1 ソフトウェアでの計算能力

まず、高速化にあたって要求される性能向上比について計算するために、ソフトウェアで代謝シミュレーションを行った場合の処理能力について評価を行った。

表 1 に、ソフトウェアによって Michaelis-Menten 型の反応モデルを、1 次の Euler 法によって解いた場合の、1 秒間あたりの処理能力を示す。この表から、PentiumIII 1.13GHz では、1 秒間に 1015 万の反応についての処理が可能であることがわかる。

4.2 必要なデータ転送量

このモデルでは、各タイムステップにおける物質の増減を計算するために、ひとつの反応に関与する物質 4 つの時刻 t における濃度を入力し、時刻 $t+1$ までの増減 4 つを得ることにな

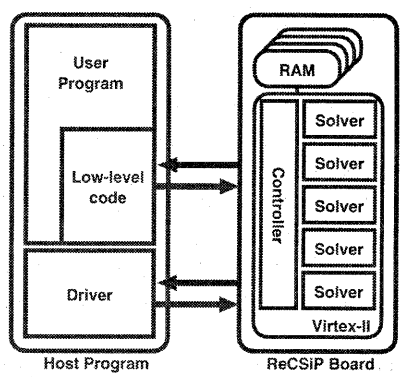

図 3 ReCSiP のシステム構成

表 1 ソフトウェアでの処理能力

プロセッサ	ops/sec.
PentiumIII 1.13GHz	10.15M
PentiumIII 800MHz	6.25M
UltraSPARCII 300MHz	5.21M

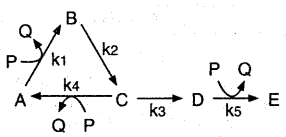

図4　反応経路の例

る。各物質の濃度、及びその差分を32bitの単精度浮動小数点型で表現するとした場合、1反応あたり(4+4)×32[bit] = 256[bit]のデータ転送が必要である。

ここで、PCIバスの転送容量は、64[bit] × 66[MHz] = 4.2[Gbit/s]であり、1反応あたり256bitのデータ転送を必要とする場合には、4.2[Gbit/s] ÷ 256[bit/op.] = 16[Mop./s]となり、1秒間に1600万の反応についてのデータ転送が限度であることがわかる。

これでは、PentiumIII 1.13GHzに対する性能向上は1.6倍に過ぎず、大幅な性能向上を果たすためにはデータ転送量の大幅な削減が必須事項であることがわかる。

4.3　反応経路マップの埋め込み

データ転送量の削減の例として図4のような反応経路を考えてみる。この一連の反応に寄与する物質はA, B, C, D, E, P, Qの7種類であるが、個々の反応速度式を考えると、

$$\frac{d[A]}{dt} = -k_1[A][P] + k_4[C][P] \quad (7)$$

$$\frac{d[B]}{dt} = k_1[A][P] - k_2[B] \quad (8)$$

$$\frac{d[C]}{dt} = k_2[B] - k_3[C] - k_4[C] \quad (9)$$

$$\frac{d[D]}{dt} = k_3[C] - k_5[D][P] \quad (10)$$

$$\frac{d[E]}{dt} = k_5[D][P] \quad (11)$$

$$\frac{d[P]}{dt} = -k_1[A][P] - k_4[C][P] - k_5[D][P] \quad (12)$$

$$\frac{d[Q]}{dt} = k_1[A][P] + k_4[C][P] + k_5[D][P] \quad (13)$$

となる(注1)。

この一連の反応速度式を解く場合に、

（1）　個々の反応速度式についてホスト・アクセラレータ間でデータの受渡しを行う

（2）　予めアクセラレータに与えた反応経路マップにしたがって、一連の反応速度式をまとめて解く

（3）　2の場合に、アクセラレータ側でデータを保持し、タイムステップ毎の転送はアクセラレータからホスト方向のみで済ませる

（4）　3の場合に、ホスト側ですべての物質の濃度変化を把握する必要がなく、$d[P]/dt, d[E]/dt$だけをホストに転送すればよい場合

というケースのそれぞれについて、1タイムステップあたりのデータ転送量から、PCIバスの性能を最大限に使った場合の最大性能を計算する。

(1)では、式7で[A], [C], [P]、式8で[A], [B], [P]、式9で[B], [C]、式10で[C], [D], [P]、式11で[D], [P]、式12で[A], [C], [D], [P]、式13で[A], [C], [D], [P]というように、20のデータをホストからアクセラレータに渡す必要がある。受け取るデータ数は7であり、各々32bitであるので、この反応系を処理する場合のPCIのバンド幅から計算される最大処理能力は、4.2[Gbps] ÷ (32[bit] × 27) = 4.9[Mtimesteps/s]となる。この場合には同一タイムステップで同一データを複数回転送しており、バンド幅の浪費が明らかである。

(2)では、(1)の場合に存在する冗長な転送がなくなり、A, B, C, D, Pの5物質の濃度をアクセラレータに転送し、A〜Qの7物質の濃度をホストが受け取ることになる。(1)と同様に最大処理能力を計算すると、4.2[Gbps] ÷ (32[bit] × 12) = 11[Mtimesteps/s]となり、倍以上の性能向上が見込める。

(3)の場合は、ホストからアクセラレータへのデータ転送がなくなるため、4.2[Gbps] ÷ (32[bit] × 7) = 18.9[Mtimesteps/s]と、さらなる性能向上が可能になる。

さらに(4)では、4.2[Gbps] ÷ (32[bit] × 2) = 66[Mtimesteps/s]と、(1)の場合の約13倍の性能が得られる。

以上のことから、アクセラレータには反応速度式を解く回路だけでなく、反応経路を解釈し、反応速度式を解く回路に必要な値を入力するための制御回路を実装することが有効であるということができる。

以下で、Michaelis-Menten型の反応モデルに基づいて表現された複数の反応で構成される反応系のシミュレーションを行うためのシステムの実装例とその評価について述べる。

5.　Michaelis-Menten solverの実装

ReCSiPボード上のFPGAにはユーザが自由に回路を実装することができるが、回路設計にはHDLなどの知識が必要になるため、多くのシミュレータ開発者には困難である。そのため、いくつかの反応モデルやアルゴリズムに基づいた回路を予め作っておき、ライブラリ化しておく必要がある。今回は、Michaelis-Menten型の反応モデルを、1次のEuler法で解く回路を実装した。

5.1　実装したアルゴリズム

実装は1次のEuler法であるので、式2〜式5に示した導関

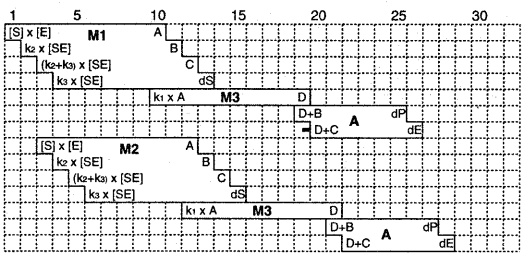

図5　パイプラインの構成

(注1)：この反応系をMichaelis-Mentenモデルに沿って記述すると、8つの反応として扱うことになるが、反応速度式の記述が繁雑になるため、ここでは質量作用則に基づいた反応式を立てて考える。

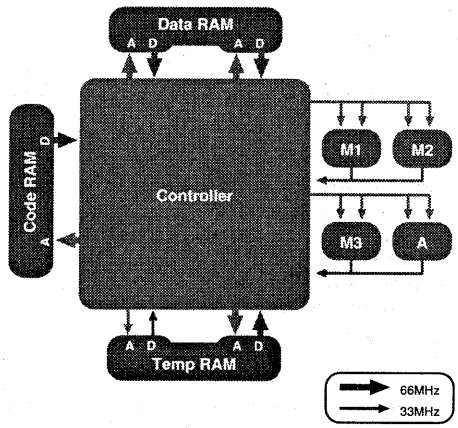

図 6　Michaelis-Menten solver の全体構成

数の値を順次積算していけばよい。また、本来は積分が必要であるが、これもタイムステップに基づく定数を導関数の値に掛けるだけであるので、反応速度定数 k_1〜k_3 に予め掛けておくことで計算を省略することができる。今回実装した回路では、これらの式の値を、それぞれパイプライン化された 3 つの乗算器 M1, M2, M3 と、加算器 A を用いて解く。各々の演算器は IEEE754 準拠の単精度浮動小数点型である。

全体のスケジュールを図 5 に示す。M1 と M2 を 2 クロックずらして、別々の反応を解くことで、後段の M3 は演算器の稼働率が 50%、A は 100%となり、有効に資源を使うことができる。M1 と M2 の稼働率は 100%であり、M3 や A に M1, M2 からのデータを入力するために数本のシフトレジスタを用いている。

5.2　Solver の全体構成

図 6 に Michaelis-Menten solver の構成を示す。データを格納しておくための data RAM, temp RAM と、動作をプログラムするためのコードが格納される code RAM、および各演算器は controller モジュールに接続されている。各メモリのデータ幅は 32bit、アドレス幅は 10bit である。Code RAM には、0 番地から順番に、controller に対する指示が書き込まれており、controller はこれにしたがって data RAM からデータを取得し、演算器に渡し、結果を temp RAM にストアする。演算器のスループットに追従するため、各メモリは演算器の倍のクロック周波数で動作し、data RAM と temp RAM は dual port 構成である。

5.3　Solver の動作

図 7 に、反応系の例を示す。この場合、data RAM の内容は表 2 のようになっている。

図 7　反応系の例

Code RAM の各番地の上位 16bit と下位 16bit には、それぞれ data RAM へのポインタが格納されている (Data RAM のアドレスは 14bit 幅であり、残りの 2bit は処理の終了や、特別な処理を指示するために用いられる)。

表 2　Data RAM の内容

A	D	A	D	A	D	A	D
0	k_{11}	4	k_{21}	8	$[S]$	c	$[SE_2]$
1	k_{12}	5	k_{22}	9	$[E_1]$	d	$[P]$
2	k_{13}	6	k_{23}	a	$[E_2]$		
3	$k_{12}+k_{13}$	7	$k_{22}+k_{23}$	b	$[SE_1]$		

Controller は、code RAM に格納されているポインタの列に従って data RAM からデータを読みだし、一旦バッファに格納したあと、演算器の動作クロックに合わせて演算器に送り込む。たとえば、図 7 の反応を解く場合に、code RAM の各番地が指す data RAM の内容は表 3 のようになる。

表 3　Code RAM の内容

A	D(High)	D(Low)	A	D(High)	D(Low)
0	$[S]$	$[E_1]$	4	$[S]$	$[E_2]$
1	k_{12}	$[SE_1]$	5	k_{22}	$[SE_2]$
2	$k_{12}+k_{13}$	k_{11}	6	$k_{22}+k_{13}$	k_{21}
3	k_{13}	—	7	k_{23}	—

演算器間のデータの受渡しは controller が自動的に行い、最終的に算出される $dS/dt, dE/dt, dP/dt$ は、controller が temp RAM の 0 番地から順番に格納する。

ここまでで述べたように、タイムステップ毎に、code RAM の 0 番地から一連の操作を繰り返すことにより、各タイムステップにおける各物質濃度の増減を計算することができ、各物質濃度の時間変化を追跡することができる。

5.4　性能評価

現時点では PCI 経由でのホストとの通信部分がデバッグ中のため、solver のみで、合成／配置配線を行った。solver1 つと 2 つのリソース使用量及び演算器部分の動作周波数を表 4 に示す。solver1 つあたりのロジック、メモリともに FPGA 全体のサイズに対する割合は 10%程度以下になっており、複数の solver を並べることによる高性能化の余地があることがわかる。

表 4　合成結果

実装数	スライス数　(%)	メモリ　(%)	動作周波数
1	3,713　(10%)	6　(4%)	48MHz
2	7,449　(12%)	12　(8%)	44MHz
3	11,163　(33%)	18　(12%)	43MHz
4	14,897　(44%)	24　(16%)	42MHz

今回実装した solver は、2 クロック毎に別の反応を解くことができるため、2 つ実装した場合の最大スループットは 44Mop./s となる。

ここで、図 4 の反応系を Michaelis-Menten モデルを用いて表現し、ホスト PC 上のソフトウェアで解く場合と、ReCSiP ボード上のハードウェアを利用する場合の処理時間を考える。この反

応系は、Michaelis-Menten モデルでは 8 つの Michaelis-Menten 型反応として扱うことができる。表 1 より、PC 上のソフトウェアでは 1 秒間に 10.15M 回 Michaelis-Menten 型の反応式を解くことができることから、1 秒間にシミュレーション可能なタイムステップ数は $10.15 \div 8 = 1.27$[Mtimesteps] である。

一方で、1 つの solver は 2 クロックごとに 1 つの反応速度式を解くことが可能であり、4 つの solver を用いれば 1 クロックごとに 2 つの反応速度式を解くことが可能になるので、4 つの solver を 42MHz で動作させた場合には、1 秒間に 84M 回反応式を解くことが可能である。したがって、ハードウェアで 1 秒間にシミュレーション可能なタイムステップ数は $84 \div 8 = 10.5$[Mtimesteps] であり、ソフトウェア処理の約 8 倍となる。

ここで、4. 節で述べたように、各タイムステップにおいて PCI バス経由ですべての代謝物質の濃度をホストに返すことのできる 1 秒あたりのスループットは最大 11[Mtimesteps] であるから、PCI バスのバンド幅は充分であることがわかる。

5.5 考　察

今回の実装と例では、FPGA による処理がソフトウェア処理の 4 倍程度の性能を示すことがわかった。Michaelis-Menten モデルを 1 次の Euler 法で解くという処理は比較的簡単な処理であるため、ソフトウェアでも高速に解くことができるが、より精密なモデルと高次の解法を用いた場合には、ソフトウェア処理によるスループットは低下する。それに対して、FPGA の面積にはまだかなりの余裕があるため、より高度な解法を用いた場合にもスループットを維持できる可能性があると考えられる。

6. ま と め

本研究報告ではまず、現在開発中の FPGA を用いた細胞シミュレーション支援プラットフォームである ReCSiP の構成について述べ、PCI カードとして実装された代謝系シミュレーションの高速化に際しての、データ転送量削減の重要性を示し、アクセレラレータ側での反応経路およびデータの保持によるバス帯域幅の有効利用方法を提案した。

続いて、Michaelis-Menten 型の反応モデルを用いたシミュレーションのための回路の実装について述べ、アクセラレータへ反応経路とデータを与える具体的な方法と実装例を示した。Xilinx 社の Virtex-II シリーズである XC2V6000-4 に 2 つの solver を実装した場合の動作周波数は約 44MHz で、例として示した反応経路のシミュレーションを行う場合のスループットは PentiumIII 1.13GHz の 8 倍であることがわかった。

また、より精密な反応モデルや高次の解法を実装した場合には、ソフトウェアのスループットは低下する一方であるのに対し、FPGA はスループットを維持できる可能性があり、その場合のソフトウェア処理に対するアドバンテージはより大きくなるものと考えられる。

今後は、安定性の高い解法を用いて微分方程式を解くアルゴリズムを実装するなどして性能評価を行うとともに、ソフトウェアの整備とあわせて、実際のシミュレータへの組み込みを目指した実装を行っていく予定である。

謝　辞

本研究は農林水産省 21 世紀グリーンフロンティア研究「イネ・ゲノムシミュレータの開発」による。

文　献

[1] Mendes P. Gepasi: a software package for modelling the dynamics, steady states and control of biochemical and other systems. *Computer Applications in the Biosciences*, Vol. 9, No. 5, pp. 563–571, Oct. 1993.

[2] I Goryanin, *et al.* Mathematical simulation and analysis of cellular metabolism and regulation. *Bioinformatics*, Vol. 15, No. 9, pp. 749–758, Sep. 1999.

[3] Masaru Tomita, *et al.* E-cell: software environment for whole-cell simulation. *Bioinformatics*, Vol. 15, No. 1, pp. 72–84, Jan. 1999.

[4] J. Schaff, *et al.* A general computational framework for modeling cellular struc ture and function. *Biophysical Journal*, Vol. 73, pp. 1135–1146, Sep. 1997.

[5] Kouichi Takahashi, *et al.* Computational challenges in cell simulation: A software engineering approach. *IEEE Intelligent Systems*, pp. 64–71, Oct. 2002.

[6] N.Tanabe et. al. Memonet: Network interface plugged into a memory slot. Technical report, Prof. of CLUSTER 2000, 2000.