一つのEFSMの複数EFSMによる実現の正しさの一証明法

1K－1
森岡 澄夫　北嶋 暁　島谷 肇　東野 輝夫　谷口 健一
大阪大学 基礎工学部 情報工学科

1. まえがき

同期式回路の設計では，回路を簡単化する（回路のゲート数を削減する）ため，回路のデータパス中の部品の制御部を，並列動作する複数個のステートマシンを組合せて実現することが多い．そのような制御部の実現の正しさを保証できることが望ましい．

本稿では，一つの拡張有限状態機械（レジスタ付きの有限状態機械．EFSMと呼ぶ）を，処理方式（レジスタ間で行う演算・データ転送の内容や，それらの順序）は変えずに，複数個のEFSMを組み合わせて実現したとき，その実現が正しく行われていることを証明する一つの方法について述べる．

その証明は，基本的には，実現側における複数のEFSMの直積マシンを考え，そのEFSMのうち初期状態から到達可能な部分と，仕様側のEFSMとが等価であることを示すことによって行う．しかし，二つのEFSMの等価性は一般には決定不能である．レジスタの取り得る値の範囲を限るなどすれば決定可能問題になるが，それでも，各レジスタの値も状態に含めた全状態 total state の上で等価性の証明を行うことにすると，計算時間が非常にかかる [1]．

そこで本稿では，証明を少ない計算量で行えるようにするため，実現を行ううえでの幾つかの現実的な設計制約を設け，そのもとで，証明を，全状態の上でなく，EFSM中の有限状態制御部の状態の上で行うことにした．

また，本稿で扱う証明対象では，仕様や実現の直積マシンの状態遷移が，複雑かつ比較的大規模な（数十状態程度）場合が多いと考えられる．そのような場合，一般には証明作業に手間がかかる．そこで，作業にかかる労力をなるべく減らすため，そのもとで証明を自動化できるような設計制約を設けることにした．

2. 証明の対象

本稿では，以下を証明の対象とする．実現は幾つかの設計制約のもとで行うが，それらの制約については **3.** でまとめて述べる．

[仕様：一つの **EFSM**] 具体的には，データパスと，それを制御する一つのステートマシンからなる，RTレベルの同期式順序回路（回路全体で一つのEFSMとみなせる）を想定している．

[実現：データパス（レジスタ）を共有する複数の **EFSM** が並列動作する回路] 各EFSMは，それぞれ同一のクロックで並列動作する．データパス上のレジスタは全てのEFSMが共有し，どのEFSMも，同じレジスタの読み書きを行える．各EFSMはクロックごと，現在のレジスタ値に関する条件（大小比較など）や，他のEFSMがどの状態にいるかの条件により，どのような演算やデータ転送を行って，どの状態へ遷移するかを決める．

上では，次のように実現する場合を想定している：仕様のEFSMと処理方式は同じままで，部分処理ごと，あるいはデータパス上のレジスタごとに，別々のEFSMで制御を行うように実現する．

以下，仕様側のEFSMをEFSM/S，実現側の各EFSMをEFSM/Ii（$1 \leq i \leq$ EFSMの総数）と呼ぶ．EFSM/S，および各EFSM/Iiの記述は，それぞれ，遷移の実行指定（どの状態でどのような条件が成り立つとき，どの遷移を実行してどの状態へ移るか）と，遷移の動作内容の指定（各レジスタに，どのような演算の結果を代入するかの指定）からなる．一つの遷移の動作内容は，（各レジスタごと）$REG := expr$の形をした代入文の集合により指定する．

3. 実現のうえでの制約

証明を効率よく行えるようにするため，以下に示す制約1～3のもとで，実現の各EFSM/Iiを設計することにする．制約の目的については **5.** で述べる．

(制約 **1**) 実現のレジスタは，仕様のレジスタをそのまま用いること（レジスタの名前や役割を変えない）．

(制約 **2**) 各EFSM/Iiの各遷移の動作内容の指定（各レジスタの代入文の右辺 $expr$）には，EFSM/Sの各遷移の動作内容中に出現する，いずれかの代入文の右辺の式を，そのまま用いること．

(制約 **3**) 各EFSM/Iiの各遷移の実行条件は，他のEFSM/Ijがどの状態にいるかの条件式と，EFSM/S中のいずれかの遷移の実行条件との，論理結合で与えられること．

2. で述べたような，部分処理ごと（あるいは機能部品ごと）にEFSMを割り当てる場合には，上記の制約のもとで設計を行えることが多いと思われる．

[制約のもとでの設計例] 文献 [2] における，標準的な5段パイプラインCPUを例として用いる．このパイプラインCPUの状態図（仕様）を図1の左上に示す．命令パイプラインの各ステージごと，それを実行する時（および，しない時）の動作内容が指定されている（if,id,･･･,no-if,･･･とラベル付けされている）．図1で，例えば遷移$T1$では，ステージifを実行し，他のステージは実行しない．各ステージの実行は，CPU中の各レジスタの現在値に関する条件（簡単のため，図1中では *stall-i* として示す）により，適宜ストールされる．

このパイプラインCPUを，制約1～3のもとで，図1の左下に示す5つのEFSMで実現した．用いるレジスタは仕様と全く同じである．各EFSM/Iiは，それぞれパイプラインの1つのステージの処理を担当し，「そのステージの実行がストールしている」状態と，「ストールしていない」状態を持つ．各EFSM/Iiの遷移の動作内容には，仕様中のif,id,･･･,no-if,･･･の動作内容をそのまま割り付けた．各EFSM/Iiの遷移の実行条件は，他のEFSMの状態に関する条件と *stall-i* との論理結合からなる．

このようにEFSMを分解することで，制御部のゲート総数が，100程度から20程度にまで減った．より大規模・複雑なEFSMを分解する場合，ゲート数が相当に減少し，回路の速度も速くなることが多い．

4. 実現の正しさの定義

本稿では，EFSM/Iiの組合せでEFSM/Sが正しく実現されていることを，以下のように定義する．

[実現の正しさの定義] EFSM/Sを初期状態から実行した時の遷移系列A（無限長．初期化の動作 $init$ も系列に含める）と，各EFSM/Iiをそれぞれ初期状態から並列実行した時の複合遷移系列B（無限長）について，任意のレジスタ初期値と，任意の入力系列のもとで，以下が満たされること：

任意の $k \geq 1$ について，(i)Bの kステップ目で，相異なるEFSM/Iiによる同じレジスタへの代入の衝突が起こらず（同じレジスタに対する代入文が複数あれば，それらの右辺の式 $expr$ が全て記号として同じ），かつ (ii)Aの kステップ目の動作内容（各レジスタ代入文）と，Bの kステップ目の動作内容が，記号として同じである（よって各レジスタに同じ値が転送される）こと．

5. 実現の正しさの自動証明手続き

ここでは，**3.** の制約のもとで，**4.** で定義した実現の正しさが満たされることを自動証明する手続きについて述べる．仕

A Technique to Prove Correctness of Implementation of an EFSM by multiple EFSMs
Sumio Morioka, Akira Kitajima, Hajime Shimatani, Teruo Higashino and Kenichi Taniguchi
Deperment of Information and Computer Sciences, Faculty of Engineering Science, Osaka University
Toyonaka-shi, Osaka 560, Japan

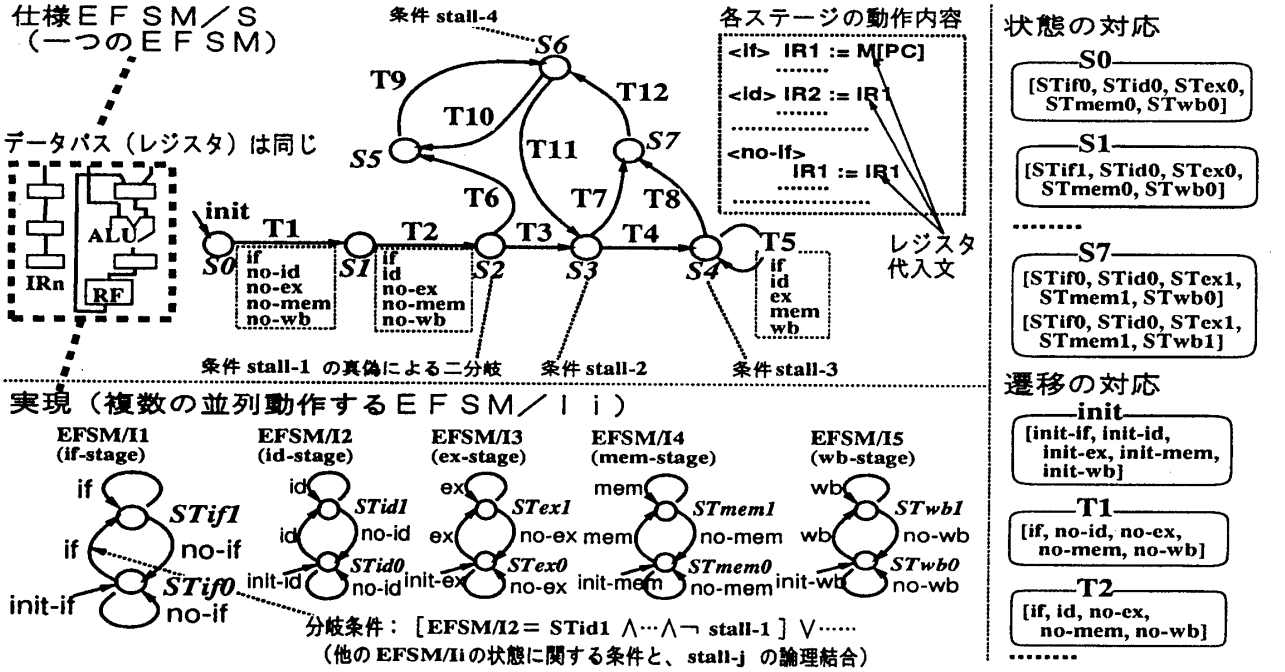

図 1: 5段パイプライン CPU を複数 EFSM で実現した例

様と実現で処理方式が全く同じであれば，多くの場合，以下の手続きで証明できると思われる．

(手順 1)EFSM/Sの実行と各 EFSM/Ii の並列実行を，それぞれの初期状態から以下のように行って，EFSM/Sの遷移（および状態）と各 EFSM/Ii の複合遷移（状態対）の対応をつける．

(1) EFSM/Sの初期状態と各 EFSM/Ii の初期状態対を，対応づける．

(2) EFSM/Sと各 EFSM/Ii をそれぞれ 1 回ずつ順次実行して，到達した（実行した）EFSM/Sの状態（遷移）と EFSM/Ii の状態対（複合遷移）を対応づける．

EFSM/Sに遷移の分岐があれば，各遷移ごと再帰的に実行をトレースして対応をつける．トレースは，EFSM/S上で，（そこに到達したときの EFSM/Ii の状態対が）既に対応づけられている状態に到達するまで続ける．

各 EFSM/Ii を動かすためには，それらの遷移のうち，どの遷移の実行条件が成立するか（どの遷移が実行されるか）を調べる必要がある．これは，論理式

∀∀∀... [他の EFSM/Ij がどの状態にいるかの条件式
→ { EFSM/Sでこれから実行する遷移の実行条件
＝ （判定する EFSM/Ii の）遷移の実行条件 }]

の真偽を，整数上の論理式（プレスブルガー文）の真偽判定アルゴリズム [3] などで調べることにより行う．同じ EFSM/Ii で，どの遷移の実行条件も真と判定されなかった場合は，証明失敗とする．

(手順 2)EFSM/Sの遷移 Ti と，それに対応する各 EFSM/Ii の複合遷移 ti（一般に複数個）の組 $< Ti, ti >$ごと，(i) ti の動作内容に，同じレジスタに対する代入文が複数個含まれているなら，それら全ての右辺（$expr$）が記号として同じであることと，(ii) Ti の各レジスタ代入文 α ごと，ti（の動作内容）に同じレジスタに対する代入文 β が存在し，かつ，α の右辺と β の右辺が記号として同じであることを示す（(i),(ii) のいずれも，記号が同じかどうかは文字列比較により調べる）．以上を示すことができれば証明は成功とし，そうでなければ証明失敗とする．

以上の手続きでは，仕様と実現で処理方式が少しでも違う場合，証明に失敗する．例えば，実現の方で，参照するレジスタを仕様とは別の（ただし実行時には値が同じになる）レジスタにした場合，実際には正しく動作するにも関わらず，証明は失敗してしまう．

しかし，上記のパイプライン CPU の例では，処理方式を全く変えていないため，上の方法で証明できる．手順 1 により，図 1の右側に示すように状態・遷移の対応が求まる．手順 2 の (i),(ii) の判定は，ごく短時間で行える．

以下，3. の設計制約 1〜3 の目的について説明する．これらの制約の目的は，証明の際，レジスタの実際の値を考えなくてもすむようにすることである．これにより，遷移の動作内容の等価性や，データ代入の衝突の有無を，単純に記号（文字列）の比較だけで調べることができる．

もしこれらの制約がなければ，記号実行などにより，各状態でレジスタ間にどのような関係が成り立つかを調べなければならなくなる．そのためには，例えば，EFSM/Sや EFSM/Ii のループ上の状態に対し，そこでレジスタ値に関して成り立つと思われる不変表明を検証者が与えて，それが実際に成り立つことを証明するなどしなければならない [4]．このため，証明にかかる手間（労力）が大幅に増え，証明自動化も困難になってしまう．

6. あとがき

本稿では，一つの EFSM を，並列動作する複数の EFSM を組み合せて実現したときに，その実現の正しさを効率よく証明できるようにするための幾つかの設計制約を示し，実際に自動で証明を行う方法についても述べた．

今後の課題としては，本稿で提案した制約のもとで実用的な例題の設計が行えるかを調べることや，本証明法の処理系を試作して，大規模な例題でも本手法が有効に働くか等を評価することが挙げられる．

参考文献

[1] Jerry R. Burch, Edmund M. Clarke, David E. Long, Kenneth L. McMillan, and David L. Dill: "Symbolic Model Checking for Sequential Circuit Verification", IEEE transactions on computer aided designs of integrated circuits and systems, Vol 13, No. 4, pp.401−424 (1994−04).

[2] 島谷肇, 森岡澄夫, 北嶋暁, 東野輝夫, 谷口健一: "代数的手法を用いたパイプライン方式 CPU の設計検証", 信学技報, DA96−02 (1996−02).

[3] 森岡澄夫, 東野輝夫, 谷口健一: "全ての変数が存在記号で束縛された冠頭標準形プレスブルガー文の真偽判定プログラム", 信学技報, SS95−18, pp.63−70(1995−07).

[4] 北道淳司, 森岡澄夫, 東野輝夫, 谷口健一: "並列実行される動作におけるデータ代入の衝突の判定", 第 47 回情処全大, 4H−01 (1993−10).