並列実行される動作におけるデータ代入の衝突の判定

4H－1

北道淳司　森岡澄夫　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

ハードウェア記述言語では，複数の動作の並列実行を自然に記述できるものが多い．しかし，複数の動作が同じレジスタ，バス，端子等に異なるデータを転送する（データ代入の衝突[1]）かどうかを判断することは難しい．高信頼という観点からは，データ代入の衝突が起きないことを保証し，その記述から高位合成することが望ましい．

本論文では，２．において，同期式順序回路を対象とし，複数動作が実行される順序回路の一モデルを示し，そのモデル上でのデータ代入の衝突を定義する．３．において，衝突が起きないことを検証するアルゴリズムを与える．

2．同期式順序回路とデータ代入の衝突

同期式順序回路Mの記述は，制御系とデータパス系からなる．回路はレジスタの集合Ｆおよび制御レジスタの集合Ｃをもつ．ここでは，簡単のためレジスタFi∈Ｆは同じレンジを持つ整数型とし，制御レジスタCj∈ＣはSTATEjk∈Sj（状態名の集合）を値に持つ．回路には，外部からのパラメータINITおよび入力INPUTが与えられる．

[制御系]

実行制御は，複数の有限制御部により行う．各制御レジスタCi∈Ｃによる制御を以下のように書く．

```
initi: if boolinit1 then <Tinit1,STATEinit1>;        (C1)
       if boolinit2 then <Tinti2,STATEinit2>;        (C2)
          ...
STATEi1:if boolstatei1 then <Ti1,STATEi1'>;          (C3)
          if...
```

initiはCiの初期値である．状態STATEijにおいて，if文の条件を表す論理式は，Fi∈Ｆ，Cj∈Ｃ，inputk∈INPUT，＋，－，if関数，＜，＝，and，not，or，定数等からなり，同時に複数のものが真にならないものとする．(C3)は，CiがSTATEi1で，かつ式boolstatei1が成り立つとき，動作Ti1を実行し，Ciの値はSTATEi1'となることを表す．

[データパス系]

動作Tiを実行するときに，レジスタへのデータの代入を以下のように書く．

```
INIT: Fj  ←expFinitj                    (I1)
Ti:   Fj  ←expFij                       (D1)
```

但し，expFinitjは，init-dataj∈INIT，＋，－，if関数，＜，＝，and，not，or，定数等からなる整数型の関数，expFijはFi∈Ｆ，Cj∈Ｃ，inputk∈INPUT，＋，－，if関数，＜，＝，and，not，or，定数等からなる関数である．

(I1)は，初期状態<init1,...initn>での各レジスタの初期値の指定を表す．(D1)は，Ti実行後のレジスタFjの値が，expFijの値となることを表す．□

回路全体は，初期状態<init1,...initn>から動作しはじめる．状態<STATE1,...,STATEn>において制御系のif文における論理式が真となる箇所の動作が並列に実行され，各制御レジスタの値はそれぞれ次の状態名になる．レジスタに複数のデータ代入があるとき，それらのデータの値が全て等しいときその値が代入され，そうでないときは衝突が生じる．以下，これらを形式的に定義する．

定義１（並列実行におけるデータ代入とデータ代入の衝突）

状態<S1,...,Sn>，各レジスタFiの値がdi，各入力データinputkの値がekの時，実行すべき動作集合を[Ti,...Tj]とする．各レジスタFnへのデータの代入

```
Ti:Fn←expi
  :
Tj:Fn←expj
```

に対し，expi,...,expjの値が全て等しいとき，次の状態でのFnの値は，expiとなる．expi,...,expjの値のうち一つでも異なるものがあるとき，動作[Ti,...Tj]はデータ代入の衝突を起こすという．衝突を起こした時の次の状態でのFnの値は定義されない．また，実行すべき動作集合にデータ代入の指定がない時は，現状態の値が保存される．□

A Decision Problem of Collision of Parallel Data Transfer
Junji Kitamichi　Sumio Morioka　Teruo Higashino　Kenichi Taniguchi
Department of Information and Computer Sciences,
Faculty of Engineering Science, Osaka University
Toyonaka-shi ,560 Japan

定義2　回路Mが与えられたとき，任意の外部からのパラメータinit-data，任意の外部からの入力系列inputに対し，初期状態から到達する任意の状態において，実行する並列動作がデータ転送の衝突を起こさないとき，回路はデータ転送の衝突を起こさずに動作し続けるという．　□

3．データ代入の衝突が起きないことの検証方法

本論文で提案する検証方法では，まず，各レジスタや各制御レジスタの値の間で成り立つべき不変式Qを検証者が考案する．次に，回路が，Qがいつも成り立つように動作し続けること，Qのもとでデータ代入の衝突が起きないことを示す．

Qを用いた検証アルゴリズム

与えられた回路Mに対して，Fi∈F，Cj∈Cを変数とし，＋，－，if関数，＜，＝，and，not，or，定数等からなる論理式をQとする．

(1) ∀init-data　Q'　が成立つことを示す．但し，Q'は，Qにおける各Fiを初期状態でのレジスタへの代入（I1)で，Cjをinitjで，それぞれ置き換えた式である．Q到達可能状態集合RSの初期値を{<init1,...,initn>}とする．

(2)状態<S1,..,Sn>∈RSにおいて

∃F∃input　(boolstate1i∧...∧boolstatenj∧Q')

が真となる並列動作[Ti,...,Tj]を実行可能な動作と呼ぶ．但し，boolstatepqは，状態Spにおけるif文の条件式であり，動作Tqを実行した後，制御レジスタCqはSq'となる．また，Q'は，Qにおける各CjをSjに置き換えた式である．

(2-1)状態<S1,..,Sn>における，実行可能な並列動作[Ti,...,Tj]に対し，

∀F∀input　(boolstate1i∧...∧boolstatenj∧Q'→
　expF1i=...=expF1n　　(F1への各代入文の右辺)
and...
and expFmi=...=expFmn)　　(Fmへの各代入文の右辺)

が成り立つことを示す．Q'は，Qにおける各Cjを，Sjに置き換えた式である．成り立たないときは，失敗である．

(2-2)さらに，状態<S1,..,Sn>における，実行可能な並列動作[Ti,...,Tj]に対し，

∀F∀input　(boolstate1i∧...∧boolstatenj∧Q'→Q")

が成り立つことを示す．Q"は，Qにおける各Fiをレジスタの代入(D1)で，Cjを動作後の状態名Sj'で置き換えた式である．成り立たないときは失敗である．

(2-3)手順(2-1)(2-2)の各論理式が成り立つとき，集合RSに，実行可能な動作[Ti,...,Tj]後の状態<S1',...,Sn'>を加える．新たな状態が付け加えられる間，手順(2)を繰り返す．

(3)手順(2)で失敗することなく，新たにRSに追加する要素がないならば，終了する(検証に成功)．□

アルゴリズムが(3)で終了すれば，初期状態より任意のパラメータや入力に対して，Qが成立し続け，かつ，データ転送の衝突を起こさずに動作し続ける．そのことは，初期状態からの動作の実行回数Nを用いた帰納法を用いて証明できる（証明略）．

アルゴリズムにおいて成り立つことを示すべき各論理式は，制御レジスタCjを整数変数，各状態名をそれぞれ異なる整数に置き換えることにより，整数上の制約論理となる．その論理式は，制約論理の恒真性判定アルゴリズム[2]を用いて成り立つことが判定できる．

[例] 検証に用いるQの例

図1は，制御部C1,C2，レジスタr1,r2,...からなる回路である．初期状態から，C1がS1r，C2がS2wまたはS2yに動作するまで，データ代入の衝突が起きないことを示すためのQの一例は以下のようになる．（C1=S1init and C2=S2initを<S1init,S2init>のように省略する）

Q= (<S1init,S2init> or <S1p,S2u> or
　<S1q,S2v> or <S1q,S2x) →r4=r2)
and (<S1p,S2u> or <S1q,S2v> or <S1q,S2x) →r3 = r1 - 4)
and C2=S2v → r1>0
and C2=S2x →not r1>0　)

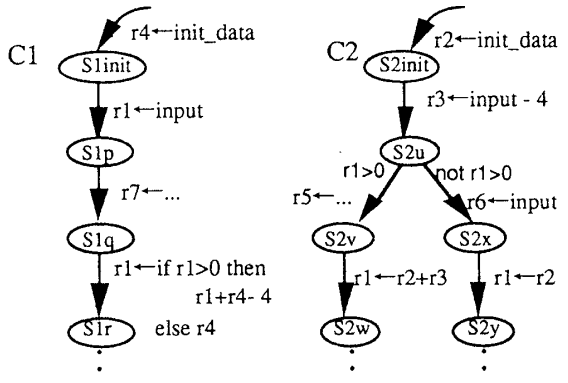

図1　データ代入の衝突を起こさない回路の例

4．まとめ

今後，本論文で提案したアルゴリズムを実現し，本手法の有効性を調べたい．

[参考文献]

[1]"UDL/1言語仕様",日本電子工業振興協会,1993.

[2]東野輝夫，北道淳司，谷口健一："整数上の線形制約の処理と応用",コンピュータソフトウエア,Vol.9, No.6,pp.31-39 (平4-11).