非同期式制御回路設計における STG の CSC conflict 解消のための効率的な因果関係追加に関する一考察

山藤 友紀 † 桑子 雅史 † 新家 稔央 †
武蔵工業大学 知識工学部 情報科学科 †

1 はじめに

現在, デジタルシステムにおける消費電力の増大などの問題を解消する手法として非同期式論理設計が注目されている [1]. 非同期式制御回路の制御仕様を表記する手法として STG(Signal Transition Graph) がある. STG は制御信号の遷移の因果関係を有向グラフで表現したものである. 設計者は STG を記述し, それを元に非同期式制御回路を設計する.

回路を生成できるために STG が満たさなければならない必要十分条件として,CSC(Complete State Coding) 条件が知られている CSC conflict を起こしている STG は回路を生成するために何らかの変更を加える必要がある. STG に新たな因果関係を追加し CSC conflict を解消する一手法 [3] が提案されているが, 最も効率的に解消する因果関係の追加箇所は明らかにされていない. 本研究では, 効率的に CSC conflict を解消するための因果関係の追加箇所を明らかにし, より少ない因果関係の追加で解消できる追加箇所を示す.

2 CSC conflict[2] の一解決手法

2.1 STG と CSC conflict

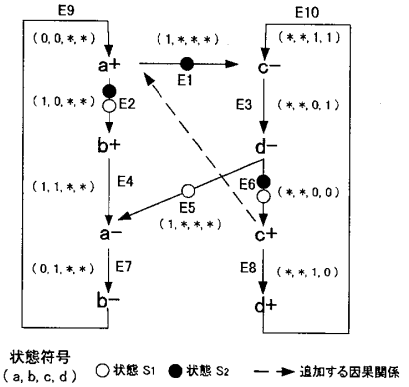

図 1: CSC conflict を起こしている STG に因果関係を追加する例

STG では制御信号 X が 0 → 1 に遷移することを X+と表し, 1 → 0 に遷移することを X− と表す. 遷移間の因果関係を有向枝で表す. 回路が現在どの状態にあるのかは有向枝上にトークンを置くことで表す. すなわち,STG における枝は状態に対応している. ある一状態の状態符号は, その状態においてトークンが置かれている枝すべての状態符号の合成となる. ある信号遷移 t に入力されている枝すべての上にトークンが乗ると, その遷移 t は発火することができる. t が発火すると入力枝上のトークンは消滅し, t のすべての出力枝上へトークンが移動する. 本研究においては, プレースは含まず, 同一信号の同一方向の遷移が 1 つだけ含まれる STG のみを対象とする.

STG から回路を生成するための条件である CSC 条件は,

- 全ての到達可能状態が異なる 2 進コードを持つ
- 2 つ以上の状態が同一の 2 進コードを持つ場合, それらの状態において遷移可能な内部信号と出力信号は等しい

のどちらかを満たすことである. CSC 条件を満たしていない STG は「CSC conflict が生じている」と言われる.

図 1 は CSC conflict が生じている STG の一例である. 図 1 において, 「○」で表したトークン配置である状態 S_1 の状態符号は E2∩E6∩E5=(10**)∩(**00)∩(1***)=(1000) である. 同様に, 「●」で表したトークン配置である状態 S_2 の状態符号は E2∩E6∩E1=(10**)∩(**00)∩(1***)=(1000) であり, 状態符号の重複 (以下では「符号重複」と略記する) が生じている。従って, この STG は CSC conflict が生じている.

STG の CSC conflict を解消する方法には, 制御信号を追加する手法と, 因果関係を追加する手法 [3] がある. 制御信号を追加する手法は常に CSC conflict を解消できるのに対し, 因果関係を追加する手法では解消できない STG も存在する. しかし因果関係を追加する手法は, 制御信号を追加する手法よりも回路速度・回路量の点で優れた回路が得られる傾向にある. 従って, 因果関係を追加する手法によって CSC conflict を解消できる STG に対しては, この手法を用いるほうが良いと言える.

2.2 因果関係の追加によって解消する手法 [3]

図 1 は CSC conflict を起こしている STG に対し, 因果関係を追加して解消する例を示す.

既存の枝に振られた状態符号中の信号のうち, 論理値が 0 または 1 に確定しているものについては, 因果関係の追加によってその論理値を反転させることはできない. 因果関係の追加では, 不定値となっている信号の論理値が 0 または 1 に確定する可能性のみである. 従って, 因果関係の追加によって CSC conflict を解消する場合, S_1 と S_2 を構成する枝の状態符号中の不定値が確定することにより, S_1 または S_2 が消滅することによって解消される可能性しかない [3].

CSC conflict を起こしている状態を消滅させるには, S_1(または S_2) を構成する枝の状態符号に含まれる信号の論理値が, S_1(または S_2) の状態符号と相反を起こすように因果関係の追加を行う. また, 論理値の相反を引き起こす際に着目する有向枝は S_1 および S_2 のどちらか一方にのみ含まれる枝とする. 図 1 において一方の状態にのみ含まれる枝は E1,E5 である.

図 1 では状態 S_2 を構成する枝 E1 において信号 c の論理値を 1 に確定させるように, c+から a+へと因果関係の追加を行っている. E1 における c の論理値を 1 に確定させると,E1 の状態符号は (1*1*) となる. E1 の状態符号が S_2 の状態符号 (1000) との間で相反するようになるため, 状態 S_1 は消滅する. こうして,S_1 と S_2 との間の符号重複を解消することができる.

2.3 因果関係追加の追加によって解消する手法の問題点

[3] において, 効率的に CSC conflict を解消するために枝の状態符号を構成する信号のうちどれに着目すべきであるかという指針は明確化されていない. そのため着目する信号の選び方によっては, 必要以上の因果関係の追加を行ってしまい, 生成される回路の速度低下・回路量増大につながる危険がある.

ここで, 状態符号が重複している状態の組 S_x,S_y のことを conflict-pair と呼び, P{S_x,S_y}と表記するものとする. 図 1 において「○」,「●」で表したトークン配置の状態 S_1,S_2 は前述のとおり状態符号が重複している。「○」,「●」のトークン配置の状態において b+が発火すると, E_2 上のトークンは E_4 上へ移動する. そのときの状態は S_3=E_4∩E_6∩E_5=(1100), S_4=E_4∩E_6∩E_1=(1100) となり, 状態符号の重複が起きている. 同様に, 図 1 において E_1 を含む状態と,E_5 を含む状態との間で生じる状態符号の重複は少なくとも S_5=E_4∩E_8∩E_5=(1110) と S_6=E_4∩E_8∩E_1=(1110), S_7=E_4∩E_{10}∩E_5=(1111) と S_8=E_4∩E_{10}∩E_1=(1111) があり, conflict-pair としては P{S_1,S_2}=(1000), P{S_3,S_4}=(1100), P{S_5,S_6}=(1110),

---

An efficient addition of causality for solving CSC-conflict of STGs in designing of asynchronous controller

† Tomoki YAMAFUJI
† Masashi KUWAKO
† Toshihiro NIINOMI
† Faculty of Knowledge Engineering, Musashi Institute of Technology

$P\{S_7,S_8\}=(1111)$ の 4 つは存在すると言える．手法 [3] ではある一組の conflict-pair に着目し，それらの状態を構成する枝の状態符号中で不定値となっている信号の論理値を確定させる．ここで,$P\{S_1,S_2\}$に着目して E_1 において不定である信号 c の論理値を 1 に確定させ, E_1=(1*1*) としたとする．これにより状態 S_2 と S_4 は消滅し,$P\{S_1,S_2\}$と $P\{S_3,S_4\}$の状態符号の重複は解消される．続いて,$P\{S_5,S_6\}$に着目して E_5 において不定である信号 d の論理値を 1 に確定させ, E_5=(1**1) としたとする．これにより状態 S_5 は消滅し,$P\{S_5,S_6\}$は解消される．しかし,$P\{S_7,S_8\}$は解消できておらず，すべてを解消するには更に因果関係の追加を行う必要がある．

対して,E_1 において不定である信号の c の論理値を 1 に確定させ, E_1=(1*1*) としたとする．これにより状態 S_2,S_4 は消滅し,$P\{S_1,S_2\}$, $P\{S_3,S_4\}$は解消される．続いて,E_5 において不定である信号 c の論理値を 0 に確定させ, E_5=(1*0*) としたとする．これにより状態 S_5,S_7 は消滅し, $P\{S_5,S_6\}$, $P\{S_7,S_8\}$ は解消される．よって E1 と E5 が原因で起こる状態重複を解消することができる．

すなわち，状態重複を解消する際にどの信号に着目するかによって，ある STG の CSC conflict を解消するために追加する因果関係が，必要以上の数になる場合がある．

3 因果関係の追加箇所の効率的な選択方法

手法 [3] では，符号重複が起きている状態を構成する有向枝において，不定となっている信号の論理値を因果関係の追加によって確定させる．有向枝 E の状態符号において論理値が確定する信号は，E を含む閉ループ中に+,− の両方の遷移が含まれるものである [3]．

従って，図 1 において 1 つの因果関係を追加して E_1 または E_5 において不定値である信号 X の論理値を確定しようとする際，E_1 を含む閉ループと E_5 を含む閉ループに X の+ または − のどちらか一方の遷移が含まれないと因果関係を追加しても X の論理値が確定することはない．よって，不定である信号の論理値を確定させる条件として，E_1 を含む閉ループと E_5 を含む閉ループに+または − のどちらか一方の遷移のみ含まれるような信号でなければならない．

図 1 の E_1 と E_5 の状態符号において不定である信号は，E_1 を含む閉ループと E_5 を含む閉ループに，+または − のどちらか一方の遷移，もしくは両方の遷移が含まれていない信号である．

前節で行った E_1 の状態符号中の不定である信号 c を 1 に，E_5 の状態符号中の不定である信号 d を 1 に確定させたとき，conflict-pair の 1 つである $P\{S_7,S_8\}$は解消されなかった．これは，$P\{S_1,S_2\}$=(1000), $P\{S_3,S_4\}$=(1100) において E_1=(1*1*) と信号 c の論理値が相反しているため，$P\{S_1,S_2\}$, $P\{S_3,S_4\}$の状態重複は解消される．また $P\{S_5,S_6\}$=(1110) において E_5=(1**1) と信号 d の論理値が相反しているため，$P\{S_5,S_6\}$の状態重複は解消される．しかし,$P\{S_7,S_8\}$=(1111) において E_1=(1*1*) と E_5=(1**1) は信号 c,d で相反が起きていないので $P\{S_7,S_8\}$の状態重複は解消できない．これを解消するには,E_1 または E_5 の状態符号中で不定な b の信号を 0 に確定させる因果関係の追加を行わなければならない．よって,E1,E5 の不定である信号 c の論理値を E_1=(1*1*), E_5=(1*0*) と相反するように確定させた方が符号重複を少ない因果関係の追加で解消できる．

以上を考慮すると，符号重複を効率的に解消する因果関係の追加箇所は以下のとおりであると考えられる．

図 2 において s_x は信号，t_x は信号遷移，E_{cx} は符号重複を起こしている 2 状態のうちどちらか一方の状態にのみ含まれる枝，L_x は E_{cx} で繋がれている閉ループを表している．

前述したように同一の信号を同一の論理値で確定させても符号重複をすべて解消することはできない．よって因果関係の追加で効率的に解消するには E_{c1}, E_{c2} において E_{c1}=(...,1,...,*,...,*,...), E_{c2}=(...,0,...,*,...,*,...) のように同一の信号の論理値を 1 と 0 にそれぞれ確定させる．これにより,E_{c1} と E_{c2} が同一の状態符号を持つことはなくなる．よって,E_{c1} と E_{c2} が原因で起こる符号重複はすべて解消できる．これにより，[3] より少ない因果関係の追加で済む場合が考えられる．

また，確定させる信号の条件は

- 範囲 t_3 から t_5 において+(または −) の一方の遷移のみが存在する
- −(または+) の反対遷移は同じ L_1 内の範囲 t_7 から t_1 に存在する

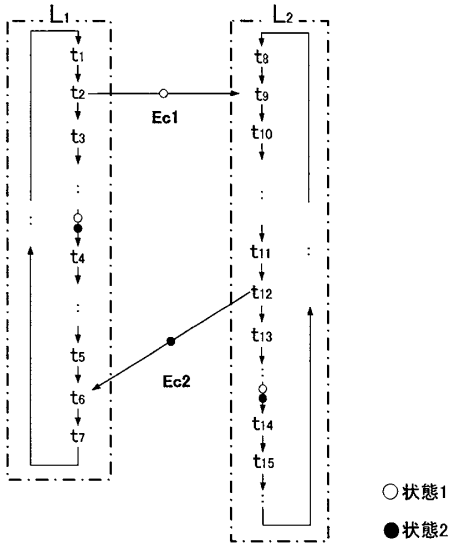

図 2: 因果関係の追加箇所の効率的な選択方法

または，

- 範囲 t_{13} から t_8 において+(または −) の一方の遷移のみが存在する
- −(または+) の反対遷移は同じ L_2 内の範囲 t_{10} から t_{11} に存在する

を満たすものであると言える．もし，この条件を満たす信号が STG 内に存在しない場合，因果関係の追加で符号重複を解消することはできない．

図 2 において上記の条件を満たす信号が L_1 内の t_4 であった場合，より少ない因果関係の追加で解消できる箇所は

1. E_{c1} の終点にある t_9 から t_4
2. t_4 から E_{c2} の始点にある t_{12}

また，条件を満たす信号が L_2 内の t_{14} となった場合より少ない因果関係の追加で解消できる箇所は

1. t_{14} から E_{c1} の始点にある t_2
2. E_{c2} の終点にある t_6 から t_{14}

これにより,E_{c1} および E_{c2} が原因で起こる符号重複はすべて解消される．

4 まとめ

STG に対して因果関係を追加することによって CSC conflict を解消する一手法 [3] において，より効率的に CSC conflict を解消するための因果関係の追加箇所の選択方法を提案した．本手法により，回路を生成するために必要な因果関係の追加がより少なくなり，実現される回路の速度性能の向上および回路量の低減が期待できる．

本稿でも述べたとおり,CSC conflict は因果関係の追加だけでは解消できない場合がある．実際には従来手法である制御信号の追加による解消方法を併用する必要がある．因果関係の追加と制御信号の追加を融合させた解消方法を提案することは今後の課題である．

参考文献

[1] 南谷 崇, “非同期式マイクロプロセッサの動向”, 情報処理, Vol. 39, No. 3, pp.181–186 (1998).

[2] Moon, C.W., “Synthesis and Verification of Asynchronous Circuits from Graphical Specification”, PhD Thesis, Univ. of California at Berkeley (1992).

[3] 木村 威暁, “STG からの非同期式制御回路設計における CSC conflict の一解決手法” 情報処理学会第 70 回全国大会講演論文集 (1), pp.177–178 (2008)