

イベントの2項関係に基づくハードウェア仕様記述

手嶋 茂晴, 平石 裕実, 矢島 脩三

Shigeharu TESHIMA, Hiromi HIRAISHI and Shuzo YAJIMA

(京都大学 工学部)

1. はじめに

時間やデータの抽象度の高いレベルで、ハードウェアの設計を支援する検証システム・DAシステムの開発が望まれており、そのためには仕様記述体系が不可欠である。本論文では、仕様はハードウェアの満たすべき要件を記述するものと考え、ハードウェアの機能的な動作についての形式的な仕様記述体系を提案する。提案する記述体系では、ハードウェアのプリミティブな動作をイベントと考え、時間やデータ値などをイベントの2項関係により表現する。また、その他にイベントの発生に関する因果関係を含めて動作を議論する。

時間と因果は半順序関係、値は同値関係によって表現する。時間を半順序関係によって表すことはハードウェアの並列動作を議論する上で有効である。イベント自体の抽象度は自由に設定できるので、記述対象のハードウェアの記述レベルに独立な記述が可能である。システムの処理対象の相対的な関係に数学的な基礎をもつ仕様記述の意味論は抽象データ型において広く研究されている^{[1][2][4]}。抽象データ型ではデータの同値関係のみによってデータ型を定義する。データ型の議論はシステムの取り扱うデータに対象が限定させる。そこで本論文では抽象データ型の仕様記述体系を拡張し、ハードウェアの動的な振る舞いを取り扱うことができる記述体系を構築する。

2.では基本的な用語を定義する。3.ではハードウェアのモデル化およびの動作履歴の表現について考え、4.では仕様記述の構文と意味について議論する。

2. 諸定義

ここでは3.以後に用いる多ソート代数^{[3][5]}と第一階述語論理^{[6][7]}に関する用語を定義する。

[定義1] 集合名の集合を $\mathbf{So}$, 関数記号の集合 $\mathbf{Fs}$ とする。ただし $\mathbf{Fs} = \bigcup_{w \in \mathbf{So}^*, s \in \mathbf{So}} \mathbf{Fs}_{w,s}$ $w \neq w'$ または $s \neq s'$ のとき, $\mathbf{Fs}_{w,s} \cap \mathbf{Fs}_{w',s'} = \emptyset$ である。このとき, $\langle \mathbf{So}, \mathbf{Fs} \rangle$ をシグニチャという。 $\mathbf{Fs}_{s_1, \dots, s_n, s}$ は $s_1, \dots, s_n$ を名前に持つ集合の積集合を定義域, s を名前に持つ集合を値域にもつ関数の記号より成る集合である。□

[定義2] $\mathbf{Sig} = \langle \mathbf{So}, \mathbf{Fs} \rangle$ をシグニチャとするととき, $\mathbf{Sig}$ をシグニチャにもつ多ソート代数 $\mathbf{A}$ は,

i) $s \in \mathbf{So}$ を集合名にもつ非空集合 $\mathbf{A}_s$,

ii) $f \in \mathbf{Fs}_{s_1, \dots, s_n, s}$ ($n > 0$) を関数記号にもつ関数 $f_{\mathbf{A}}$,

$$f_{\mathbf{A}}: \mathbf{A}_{s_1} \times \mathbf{A}_{s_2} \times \dots \times \mathbf{A}_{s_n} \rightarrow \mathbf{A}_s,$$

iii) $f \in \mathbf{Fs}_{\epsilon, s}$ (ϵ は空系列) に対応する $\mathbf{A}_s$ の要素, から構成される。多ソート代数を以下では単に代数という。□

以下に定義する第一階述語論理は一般の述語論理に型の概念を導入したものである。

第一階述語論理は以下の記号をアルファベットにもつ言語のクラスである。有限集合 S (型の名前の集合) に対して、以下のように記号を定める：

- i) 変数記号 変数記号の集合(可算無限)を X とする。ただし $X = \bigcup_{s \in S} X_s$, $s \neq s'$ のとき $X_s \cap X_{s'} = \emptyset$, X_s は s に対応する集合の要素をとる変数の集合である。
- ii) 関数記号 関数記号の集合を F とする。ただし, $F = \bigcup_{w \in S^*, s \in S} F_{w,s}$, $w \neq w'$ または $s \neq s'$ のとき $F_{w,s} \cap F_{w',s'} = \emptyset$ 。
- iii) 述語記号 述語記号の集合を P とする。ただし $P = \bigcup_{i=1,2,\dots} P_i$, $i \neq j$ のとき $P_i \cap P_j = \emptyset$, P_i は i -引数述語記号の集合である。
- iv) 論理記号 $\neg, \vee, \exists$
- v) 補助記号 $(,), ,$

S に対して, X を変数記号の集合, F を関数記号の集合, P を述語記号の集合にもつ言語を $L(S, X, F, P)$ と書く。 $L(S, X, F, P)$ には項と論理式の2種類の表現が定義される。

[定義3] $L(S, X, F, P)$ において, 以下の i)~iii) を有限回繰り返して得られるものだけが項である: ここに, $T_s (s \in S)$ は以下の i)~iii) を有限回繰り返して得られる要素だけの集合であり, $T = \bigcup_{s \in S} T_s$ とする。

- i) $a \in F_{\varepsilon,s}$ は T_s に含まれる項である。
- ii) $x \in X_s$ は T_s に含まれる項である。
- iii) $t_1 \in T_{s_1}, t_2 \in T_{s_2}, \dots, t_n \in T_{s_n}, f \in F_{s_1 s_2 \dots s_n, s}$ のとき,

$f(t_1, t_2, \dots, t_n)$ は T_s に含まれる項である。 $\square$

[定義4] $L(S, X, F, P)$ において, 以下の i)~iv) を有限回繰り返して得られるものだけが論理式である:

- i) $p \in P_n, t_1, t_2, \dots, t_n \in T$ のとき, $p(t_1, t_2, \dots, t_n)$ は論理式である。
- ii) l が論理式のとき, $(\neg l)$ は論理式である。
- iii) l_1, l_2 が論理式のとき, $(l_1 \vee l_2)$ は論理式である。
- iv) l が論理式, $x \in X$ のとき, $(\exists x l)$ は論理式である。

また, $p \in P_n, t_1, t_2, \dots, t_n \in T$ のとき, $p(t_1, t_2, \dots, t_n), \neg p(t_1, t_2, \dots, t_n)$ をリテラルという。 $\square$

$(\neg(l_1 \vee l_2))$ を $((\neg l_1) \wedge (\neg l_2))$, $(\neg(\exists x l))$ を $\forall x (\neg l)$, $(l \vee (\neg l))$ を $true$ と書くこともある。また $\exists, \neg, \wedge, \vee$ の順に演算子の優先順位を定め, 不要なカッコは省くことができるものとする。

[定義5] $L(S, X, F, P)$ の表現の有限集合を Ex に対して, Ex に現れる $F_{\varepsilon,s}$ の要素からなる集合を H^0_s とする。 Ex に $F_{\varepsilon,s}$ の要素が現れないときは適当な定数 a_s をとり, $H^0_s = \{a_s\}$ とする。そして, $H^i_s = H^{i-1}_s \cup \{f(t_1, t_2, \dots, t_n) \mid f \in F_{s_1 s_2 \dots s_n, s}, t_1 \in H^{i-1}_{s_1}, t_2 \in H^{i-1}_{s_2}, \dots, t_n \in H^{i-1}_{s_n}\}$ と定義するとき, $H = \bigcup_{s \in S} H^\infty_s$ を Ex の Herbrand 領域という。また, H^∞_s は H_s と略記される。 $\square$

[定義6] $L(S, X, F, P)$ の表現の有限集合を Ex , Ex のHerbrand領域を H とするとき, $HB = \{p(t_1, t_2, \dots, t_n) \mid p \in P_n, t_1, \dots, t_n \in H, n = 1, 2, \dots\}$ をHerbrand基底集合という. ここで, p は Ex に現れる述語記号である. HB の要素を基礎例という. また, HB の部分集合をHerbrand解釈という. $\square$

[定義7] I をHerbrand解釈とするとき, I による論理式の真偽を関数 m_I を用いて帰納的に定義する:

変数を含まないリテラル $p(t_1, t_2, \dots, t_n)$ に対して,

$$m_I(p(t_1, t_2, \dots, t_n)) = \begin{cases} 1, & p(t_1, t_2, \dots, t_n) \in I \text{ のとき,} \\ 0, & p(t_1, t_2, \dots, t_n) \notin I \text{ のとき.} \end{cases}$$

論理式 l に対して,

$$m_I(\neg l) = 1 - m_I(l).$$

論理式 l_1, l_2 に対して,

$$m_I(l_1 \wedge l_2) = \min(m_I(l_1), m_I(l_2)).$$

論理式 l と変数 $x \in X_s$ ($s \in S$)に対して,

$$m_I(\exists x l) = \max_{\sigma} (m_I(l\sigma)), \quad \sigma \text{ は変数 } x \text{ を Herbrand 領域の部分集合 } H_s \text{ の要素に置き換える置換操作.}$$

論理式 l は $m_I(l) = 1$ のとき, 解釈 I において真であり, $m_I(l) = 0$ のとき, 解釈 I において偽である. $\square$

3. ハードウェアモデル

3.1 イベントとキャリア

仕様記述体系の基となるハードウェアモデルについて述べる. ハードウェアは有限固定個のキャリアから構成され, イベントをキャリア上に発生させるものとする. イベントはプリミティブな動作である. イベントは発生後, ある期間キャリア上に存在し, そして消滅する. イベントは時間的には期間を表す. 信号線, レジスタなどがキャリアであり, 信号線が論理値1をとっていることなどがイベントである.

キャリア上に発生するイベントは, 1つ1つ異なるイベントとして識別される. イベントにはイベントの種類を示す値が定義される. イベントの値は従来のハードウェア記述におけるデータに対応する. 設計者は設計のレベルや設計の内容に応じて値の定義される集合を設定する. イベントの値はイベントの発生するキャリアによって型が決まる.

3.2 イベントの2項関係による動作履歴の表現

ハードウェアの動作履歴は動作における時間概念を直接に反映する. 前節の議論より履歴はどのキャリアでどのイベントがいつ発生し, いつ消滅したかを示す記録, つまりイベントの値, 時間の記録である. 本論文ではどのように, どうしてイベントが発生したかという因果関係も履歴に含める. 因果と時間はともにハードウェアの動的な振る舞いに関する概念であるが, 物理的な現象を表す時間と, 物理

的なハードウェアの構成に独立な概念である因果とを区別する．因果関係を用いることによってハードウェアの実現と独立で、抽象的なハードウェアの動作を記述することが可能になる．

イベントは動作の最小単位である．したがって、イベントより細かな単位を基準にして履歴を表現することはイベントがプリミティブな動作であることに反する．イベントの相対的な関係のみで表現しなければならない．イベントの関係は履歴を表現する十分な能力を持つ．

以下 (1)因果, (2)時間, (3)値の順に履歴の表現について述べる．

(1) 因果: 因果を表わす2項関係は反射律の成立を仮定すると、イベントの半順序関係である．イベント a がイベント b の発生の原因、イベント b がイベント a の結果であるとき、かつそのときにのみ $a \geq_c b$ とする．図1に示すDフリップ・フロップの場合、イベント e_1, e_2 は Q 上に発生するイベント e_3 の原因であるので、 $e_1 \geq_c e_3$ かつ $e_2 \geq_c e_3$ が成立する．

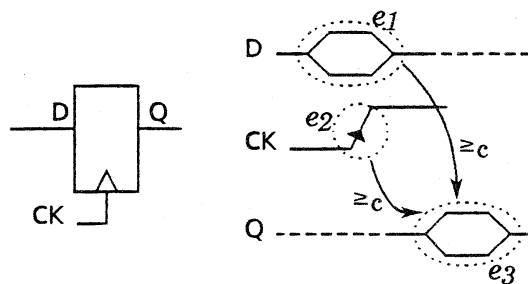


図1 D-フリップ・フロップの動作

(2) 時間: 『イベント a の消滅後にイベント b が発生する』という2項関係のみで時間を考える．時間を表す2項関係は、時間が過去から未来へ流れることに対応して、因果関係と同様に半順序関係である．イベント a の消滅後にイベント b が発生するとき、かつそのときにのみ $a \geq_t b$ とする．図2に示すようにイベントが発生するとき、図3に示す関係が成立する．また、 $a \nlessdot_t b$ かつ $a \nlessdot_t b$ なるイベント a, b は並列に処理される可能性をもつ．しかし、一般に $a \nlessdot_t b$ かつ $a \nlessdot_t b$ なる2つのイベントの組がすべて並列に処理できるとは限らない．

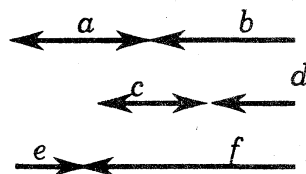


図2 図式的な時間表現

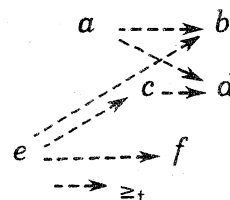


図3 2項関係による時間表現

(3) 値: 値には型が定義されたとした．型の概念は多ソート代数によって形式化される．本モデルでは、イベントの相対的な関係を議論の対象としているので、代数の具体的な構成を知る必要はない．したがって、値は抽象データ型と考える．そこで

代数を構成する集合において要素 d_1, d_2 が同一の要素であるとき、かつそのときのみ $d_1 =_v d_2$ が成立する同値関係を用いて値を表現する。イベントからイベントの値を求める関数を val とし、 val を用いて $=_v$ をイベントと値の集合との和集合における同値関係に拡張する。イベント a の値が d であるとき、つまり $val(a)=d$ のとき、かつそのときにのみ $a=_v d$ 、またイベント a とイベント b が同じ値をもつとき、つまり $val(a)=val(b)$ のとき、かつそのときにのみ $a=_v b$ とする。

図1において、フリップ・フロップを機能させるキャリア ck 上のイベント(信号値の立ち上がり)の値を p -edgeとすると、 $e_1 =_v e_3$ かつ $e_2 =_v p$ -edgeが成立する。

(1)~(3)の議論により、履歴の形式的な表現を定義する。

[定義8] C をキャリアの名前の集合、また $Sig = \langle So, Fs \rangle$ をシグニチャとして、関数 $v: C \rightarrow So$ が定義されるとき、 $\langle C, So, Fs, v \rangle$ をスキーマという。□

C はハードウェアを構成するキャリア名の宣言である。 Sig はイベントの値がつくる代数のシグニチャである。また、 v はキャリア上に発生するイベントの値の型を決める関数である。スキーマはハードウェアの静的構成を明らかにする。

[定義9] 2項関係 R_1, R_2 をもつイベントの集合 E 、 $Sc = \langle C, So, Fs, v \rangle$ なるスキーマに対して、 $Sig = \langle So, Fs \rangle$ をシグニチャにもつ代数を A とし、関数 car, val を $car: E \rightarrow C$ 、 $val: E \rightarrow \bigcup_{s \in So} A_s$ とすると、 $\langle Sc, E, R_1, R_2, car, val, A \rangle$ なる数学的構造が以下の条件を満たすならば、 $\langle Sc, E, R_1, R_2, car, val, A \rangle$ をスキーマ Sc に対する履歴という：

- i) $e \in E$ ならば $val(e) \in A_{v(car(e))}$ 。
- ii) R_1, R_2 は半順序関係で、 E は R_1, R_2 に関して共通の最大元をもつ。
- iii) $e_1, e_2 \in E$ 、 $e_1 \neq e_2$ 、 $e_1 R_1 e_2$ ならば $e_2 \not R_2 e_1$ 。□

履歴 $\langle Sc, E, R_1, R_2, car, val, A \rangle$ において、 R_1 は因果関係 $\geq_c$ 、 R_2 は時間関係 $\geq_t$ に対応し、 R_1 と R_2 の最大元は処理の開始を表わす仮想的イベントである。以下では最大となるイベントを e_{beg} と書くことにする。

4. 仕様記述 構文と意味-

4.1 第一階述語論理による関係の記述

仕様はハードウェアの満たすべき要件である。本論文では機能的動作を対象にし、その構文と意味について考察する。モデルにおいて動作に関する仕様は履歴において成立すべき因果、時間、値についての性質である。つまり履歴において、

- 1) 値のつくる代数
- 2) イベント発生
- 3) 時間的制約 について成立すべき性質が記述される。

因果、時間、値の2項関係に2引数述語を対応させ1)~3)の性質について第一階述語論理を用いて記述する。

$Sc = \langle C, So, Fs, v \rangle$ をもつハードウェアの履歴 $\langle Sc, E, R_1, R_2, car, val, A \rangle$ に対して、 $L_{Spec} = L(C \cup So, X, F, P)$ の論理式を用いる。ここで $Fs_{w,s} \subseteq F_{w,s}$ ($w \in So^*$, $s \in So$)、 $\{val_c\} \subseteq F_{c,v(c)}$ ($c \in C$)、 $\{beg\} \subseteq F_{e,car(e_{beg})}$ 、 $\{Eqv, Cas, Bfr\} \subseteq P_2$ である。 Eqv は、代

数と付値関数 val によって定義されるイベントの同値関係 $=_v$ に対応し, Cas は因果関係 $\geq_c (x \geq_c y : Cas(x, y), x \text{ causes } y)$, Bfr は時間関係 $\geq_t (x \geq_t y : Bfr(x, y), x \text{ before } y)$ にそれぞれ対応する述語記号で, val_c は val の定義域を $car(e) = c (c \in C)$ となるイベントに制限した関数の関数記号であり, $beg \in F_{e, car(e_{beg})}$ は開始のイベント e_{beg} に対応する定数である. しかし, 一般の L_{Spec} の論理式は履歴を解釈にして, 真偽が判定できない. 履歴は動作過程において発生したイベントについて値, 因果, 時間の関係を表現するものであり, 発生していないイベントについては記述されていない. そこで履歴に対して L_{Spec} の論理式が意味をもつように構文に制限を加える.

[定義8] L_{Spec} の論理式 l に出現する拘束変数 x が 項 t 以前(以降)に制限されている ということを次のように定義する:

i) x が l の冠頭標準形において v で拘束されているとき, 以下の条件を満たす冠頭和積標準形 l' が存在する:

x の出現するすべての和項において, $\neg Cas(t_i, t_{i+1}) (\neg Cas(t_{i+1}, t_i)) \ i=1, \dots, n-1, t_1=x, t_n=t$ なるリテラルがすべて, または $\neg Cas(x, y) (\neg Cas(y, x))$ なるリテラルがその和項に出現する. ここで, $t_1, t_2, \dots, t_n$ は項であり, y は項 t 以前(以降)に制限されている変数である. これらは和項ごとに定まる.

ii) x が l の冠頭標準形において $\exists$ で拘束されるとき, 以下の条件を満たす冠頭積和標準形 l' が存在する:

x が出現するすべての積項において, $Cas(t_i, t_{i+1}) (Cas(t_{i+1}, t_i)) \ i=1, \dots, n-1, t_1=x, t_n=t$ なるリテラルがすべて, または $Cas(x, y) (Cas(y, x))$ なるリテラルがその積項に出現する. ここで, $t_1, t_2, \dots, t_n, y$ は i) と同様である.

また, l の自由変数については, i) の条件 (v で拘束されることは除く) を満たすとき v の意味で項 t 以前(以降)に制限されるといい, また ii) の条件(同様)を満たすとき $\exists$ の意味で項 t 以前(以降)に制限されるという. $\square$

変数 x が項 t 以前に制限されるとき, 項 t の原因でないイベントへの変数 x の置換は論理式の真偽に影響しない.

スキーマを $Sc = \langle C, So, Fs, v \rangle$ とするとき, Sc をスキーマにもつハードウェアの動作に対する仕様は, L_{Spec} の論理式と $\Rightarrow_c, \Rightarrow_t$ の記号からなる $\text{syn1}) \sim \text{syn3})$ の構文をもつ表現の集合である:

$\text{syn1}) \text{ Eqv}(t, t'), t, t' \in T_s (s \in So)$

$\text{syn2}) l_1 \Rightarrow_c l_2 \quad l_1, l_2 \text{ は } L_{Spec} \text{ の論理式.}$

l_1 は次の条件を満たす:

i) 自由変数は $\exists$ の意味で beg 以降に制限される.

ii) 拘束変数はいずれかの自由変数以後に制限され, かつ, いずれかの自由変数以前に制限される.

iii) 冠頭積和(和積)標準形で $\neg \text{Eqv}(t, t') \ t, t' \in T$ なるリテラルを含まない.

iv) $t \in T_s (s \in So)$ は Cas, Bfr の引数ではない.

l_2 は $\text{Eqv}(x, t)$ なる論理式である. ここで x は $x \in X_c, c \in C$ なる l_1 の自由変数で, t は l_1 の自由変数のみを変数に含む項である.

syn3) $l_3 \Rightarrow_t l_4$ l_3, l_4 は L_{Spec} の論理式.

l_3 は次の条件を満たす:

- i) 変数はすべて自由変数で $\exists$ の意味で *beg* 以降に制限されている.
- ii) 正リテラルの積である.

l_4 は $Bfr(x, x')$ なる論理式である. x, x' は l_3 に出現する自由変数であり, l_3 の自由変数は x または x' の以前に制限される. $\square$

syn1) の表現 $Eqv(t, t')$ は直観的には, 項 t, t' を評価するとき, 値が等しいことを意味する. syn2) の表現 $l_1 \Rightarrow_c l_2$ は l_1 の自由変数を履歴のイベントに置き換えたとき, l_1 が真ならば, 値について l_2 を満たすイベントが発生することを意味する. つまり l_1 の自由変数に割りあてられたイベントに対して, l_2 を満たすイベントが存在し, その間に因果関係が成り立つ. syn3) の表現 $l_3 \Rightarrow_t l_4$ は l_3 の自由変数を履歴のイベントに置き換えたとき, l_3 が真ならば l_4 で表わされる時間的制約が成り立つことを意味する.

4.2 仕様記述の不動点意味

仕様記述の意味を入力および初期値から動作した履歴を求める関数として定義する. 本論文では入力・初期値は履歴の中であらかじめわかっている有限個のイベントの関係と考える. 入力と初期値をあわせて入力パターンということにする. 仕様を公理と考え, そこからイベント間の関係を導出することによって, ハードウェアの動作をシミュレートする. 導出可能なすべての関係が入力パターンに対する履歴である. 我々は入力パターンを Herbrand 解釈に対応づけ, Herbrand 領域で意味を議論する. まずそのための Herbrand 領域を設定する.

[定義10] スキーマ $Sc = \langle C, So, Fs, v \rangle$ の仕様 $Spec$ に対して, 関数 $cond : Spec \rightarrow L_{Spec}$ の論理式の集合, と関数 $lit : Spec \rightarrow$ 正リテラルの集合 を次のように定義する: $exp \in Spec$ に対して,

- i) exp が $Eqv(t, t')$ なる表現(syn1)) のとき,

$$\begin{aligned} cond(exp) &= true, \\ lit(exp) &= \{Eqv(t, t')\}. \end{aligned}$$

- ii) exp が $l \Rightarrow_c Eqv(x, t)$ なる表現(syn2)) のとき,

$$\begin{aligned} cond(exp) &= l, \\ lit(exp) &= \{Cas(x_1, f_{exp}(x_1, \dots, x_n)), Cas(x_2, f_{exp}(x_1, \dots, x_n)), \dots, Cas(x_n, f_{exp}(x_1, \dots, x_n)), \\ &\quad Eqv(f_{exp}(x_1, \dots, x_n), t), Bfr(beg, f_{exp}(x_1, \dots, x_n))\} \end{aligned}$$

ここで, $x_1, \dots, x_n$ は l の自由変数, f_{exp} は exp に対してあらかじめ決められている関数記号で, $Fs_{c_1 c_2 \dots c_n, c} (x \in X_c, x_1 \in X_{c_1}, x_2 \in X_{c_2}, \dots, x_n \in X_{c_n}, c, c_1, c_2, \dots, c_n \in C)$ の要素である. また $exp_1 \neq exp_2$ ならば $f_{exp_1} \neq f_{exp_2}$ とする.

- iii) exp が $l \Rightarrow_t Bfr(x, x')$ なる表現(syn3)) のとき,

$$\begin{aligned} cond(exp) &= l, \\ lit(exp) &= \{Bfr(x, x')\}. \quad \square \end{aligned}$$

キャリアごとに $f_c \in F_{c,c}, a_c \in F_{e,c} (c \in C)$ なる関数記号を選ぶとき, 表現の集合 $\cup_{exp \in Spec} \{cond(exp)\} \cup lit(exp) \cup c \in C \{f_c(a_c)\}$ の Herbrand 領域 H , Herbrand 基底集合

HB は仕様 $Spec$ の意味を議論するための十分な能力をもつ。以後これらを単に H , HB と記す。

仕様の表現は動作の立場から見ると, $\text{syn1}) \sim \text{syn3})$ のいずれの表現も, 「途中までの履歴が条件を満たせば, 新しくイベントに関係が生じる」ということを記述している。このことを Herbrand 解釈上で考えると, 表現は Herbrand 解釈で条件が満たされるとき, 新しく生じる関係に対応する基礎例を解釈に追加する手続きを意味する。関数 cond は表現の意味する手続きが適用される条件, 関数 lit は条件が成立するとき, Herbrand 解釈に追加される基礎例をリテラルで表す。 $\text{syn2})$ の場合, 新しく発生するイベントを $f_{\text{exp}}(x_1, \dots, x_n)$ によって表している。 f_{exp} はイベントを返す関数の関数記号であるが, ここでは関数の内容は考えず, イベントを識別するために名前をつける働きをする。しかし, 定義9よりすべての Herbrand 解釈が履歴を表現するわけではない。 Cas , Bfr , Eqv によって表される述語を Herbrand 領域上の2項関係と見なしたとき, Cas , Bfr が半順序関係に, Eqv が同値関係にならなければならない。また, Eqv による同値類分割が代数を構成しなければならない。これらの条件を満たす Herbrand 解釈の集合を U_{his} とする。

任意の解釈 $I \in U_{\text{his}}$ は以下のように履歴 $\langle Sc, E, \geq_c, \geq_t, \text{car}, \text{val}, A \rangle$ に対応づけることができる:

- 1) $\text{Cas}(\text{beg}, t) \in I$ となる項 t のみを含む集合 E , またそのとき, $t \in H_c$ ($c \in C$) ならば, $\text{car}(t) = c$ と定義される関数 car .
- 2) E の要素 t, t' に対して, Cas , Bfr に対応する $\geq_c, \geq_t$.
- 3) Eqv に対応する同値関係 $=_v$ による H_s ($s \in So$) の同値類分割 $H_s / =_v$ と, $f \in F_s$ に対応して同値類分割上に定義される定数, 関数から構成される代数 A .
- 4) E の要素 t に対して, $\text{Eqv}(t, t') \in I$ かつ $t' \in H_s$ ($s \in So$) のとき, t' が含まれる $H_s / =_v$ の同値類 d とすると, $\text{val}(t) = d$ により定義される関数 val .

U_{his} の要素は履歴 $\langle Sc, E, \geq_c, \geq_t, \text{car}, \text{val}, A \rangle$ に現れるイベントおよび代数の要素に Herbrand 領域の要素によって名前をつけたものである。

一方, 入力パターン $\langle Sc, E_{\text{IP}}, \geq_c, \geq_t, \text{car}, \text{val}, A_{\text{IP}} \rangle$ は次のような U_{his} の要素 I に対応づけることができる。入力パターンにおいてキャリア $c \in C$ 上に発生するイベントに対して項 $a_c, f_c(a_c), f_c(f_c(a_c)), \dots \in H_c$ を割り当てる。 I は以下のような Herbrand 解釈である:

任意のイベント $e, e' \in E_{\text{IP}}$ に対して,

- 1) $e \geq_c e'$ のとき, かつそのときにのみ $\text{Cas}(t_e, t_{e'}) \in I$.

$e \geq_t e'$ のとき, かつそのときにのみ $\text{Bfr}(t_e, t_{e'}) \in I$.

ただし, $t_e, t_{e'}$ はそれぞれイベント e, e' に割り当てられた項である。

- 2) $e =_v e'$, $\text{car}(e) = c$ かつ $\text{car}(e') = c'$ のとき, かつそのときにのみ

$\text{Eqv}(t_e, t_{e'}), \text{Eqv}(\text{val}_c(t_e), t_{e'}), \text{Eqv}(t_e, \text{val}_{c'}(t_{e'})), \text{Eqv}(\text{val}_c(t_e), \text{val}_{c'}(t_{e'})) \in I$.

ただし, $=_v$ は代数 A_{IP} と関数 val から得られる同値関係である。

- 3) $\text{val}(e) = d$ かつ $\text{car}(e) = c$ のとき, かつそのときにのみ

$Eqv(t_e, t_d), Eqv(val_c(t_e), t_d) \in P$. ただし, d は代数 A_{IP} の要素であり, t_d は代数 A_{IP} において d と評価される項である.

4) 項 $t, t' \in H_s$ ($s \in So$) が代数 A_{IP} において同一の要素に評価されるとき, かつそのときにのみ $Eqv(t, t') \in P$.

[定理1] T を任意の解釈 $I \in U_{his}$ に対して $T \supseteq I$ となる Herbrand 解釈とし, $U = U_{his} \cup \{T\}$ とするとき, U は $\supseteq$ に関して完備束である.

証明は自明である. $\square$

次に関数 $lit, cond$ を用いて仕様と入力パターンからイベントの関係を導き出す関数を定義する.

[定義11] スキーマ $Sc = \langle C, So, Fs, v \rangle$ の仕様 $Spec$ に含まれる $syn1) syn2)$ の形の表現すべてから成る部分集合を $Spec^V, C, syn3)$ の形の表現すべてから成る部分集合を $Spec^T$ とするとき, 次のように関数 $\tau_1, \tau_2, \tau_3: 2HB \rightarrow 2HB$ を定義する:

$I \in 2HB$ に対して,

$\tau_1(I) = I \cup \{L\sigma \mid L \in lit(exp)\}$ $exp \in Spec^V, C, m_I(cond(exp)\sigma) = 1$ のとき,
 I 任意の $exp \in Spec^V, C$, 任意の置換 σ に対して, $m_I(cond(exp)\sigma) = 0$ のとき.

$\tau_2(I) = I \cup \{L\sigma \mid L \in lit(exp)\}$ $exp \in Spec^T, m_I(cond(exp)\sigma) = 1$ のとき,
 I 任意の $exp \in Spec^T$, 任意の置換 σ に対して, $m_I(cond(exp)\sigma) = 0$ のとき.

$\tau_3(I) = I \cup \{Cas(x, x)\sigma \mid m_I(Cas(beg, x)\sigma) = 1\}$
 $\cup \{Cas(x, z)\sigma \mid m_I(Cas(x, y) \wedge Cas(y, z) \wedge Cas(beg, x)\sigma) = 1\}$
 $\cup \{Bfr(x, x)\sigma \mid m_I(Cas(beg, x)\sigma) = 1\}$
 $\cup \{Bfr(x, z)\sigma \mid m_I(Bfr(x, y) \wedge Bfr(y, z) \wedge Cas(beg, x) \wedge Cas(beg, y) \wedge Cas(beg, z)\sigma) = 1\}$
 $\cup \{Bfr(beg, x)\sigma \mid m_I(Cas(beg, x)\sigma) = 1\} \cup \{Eqv(w, w)\sigma\}$
 $\cup \{Eqv(w, v)\sigma \mid m_I(Eqv(v, w)\sigma) = 1\} \cup \{Eqv(w, u)\sigma \mid m_I(Eqv(w, v) \wedge Eqv(v, u)\sigma) = 1\}$
 $\cup \{Eqv(t'' t_1/x_1 t_2/x_2 \dots t_n/x_n, t'' t'_1/x_1 t'_2/x_2 \dots t'_n/x_n) \mid t'' \in T,$
 $Eqv(t_1, t'_1), Eqv(t_2, t'_2), \dots, Eqv(t_n, t'_n) \in I, x_1, x_2, \dots, x_n \text{ は } t'' \text{ に現れる変数}\}$
 $\cup \{Eqv(x, w)\sigma \mid m_I(Eqv(val_c(x), w)\sigma) = 1\} \cup \{Eqv(val_c(x), w)\sigma \mid m_I(Eqv(x, w)\sigma) = 1\}$

ここで, $x, y, z \in \bigcup_{c \in C} X_c$ $w, v, u \in X$, ただし, σ は論理式の自由変数 $x \in X_s$ ($s \in C \cup So$) の Herbrand 領域の部分集合 H_s の要素への置換である. $\square$

τ_1 は $syn1), syn2)$ の形の表現に対応する関数であり, τ_2 は $syn3)$ に対応する関数である. τ_3 は解釈に履歴の条件を満足させるようとする関数である. ただし, 半順序関係の反対称律の成立は保証されない. τ_1, τ_2, τ_3 は σ の選び方について定義されていないが, σ の選び方は以後の議論において本質でない. τ_2, τ_3 についてはチャーチ・ロッサ性が成り立つ. τ_2, τ_3 はリテラル追加の条件が正リテラルのみによって表さ

れていることにより容易に理解される. また, τ_1 についてもチャーチ・ロッサ性が保証される範囲で用いられる.

τ_2, τ_3 に関して次の定理が成り立つ.

[定理2] 任意の解釈 $I \in 2HB$ に対して, $X = \{I, \tau_3 \circ \tau_2(I), (\tau_3 \circ \tau_2)^2(I), \dots\}$ とするとき, $\sup X$ (X の上限) は一意に決まり, かつ $\tau_3 \circ \tau_2$ の不動点である.

(証明) $\tau_3 \circ \tau_2$ は基礎例を追加する関数であるから, $I \subseteq I'$ ならば, $\tau_3 \circ \tau_2(I) \subseteq \tau_3 \circ \tau_2(I')$ したがって, $I \subseteq \tau_3 \circ \tau_2(I) \subseteq (\tau_3 \circ \tau_2)^2(I) \subseteq \dots \subseteq \sup X$ であり, また $\sup X \subseteq \tau_3 \circ \tau_2(\sup X)$ も成立する. いま, $\tau_3 \circ \tau_2(\sup X) \supset \sup X$ と仮定し, $g \in \tau_3 \circ \tau_2(\sup X)$, $g \in \sup X$ とする. $\tau_3 \circ \tau_2$ の定義より, g が追加される条件は, 有限個の基礎例が解釈に含まれているか否かで真偽が決められる. $\sup X = \sup\{x \mid x \in X\}$ より, 真偽を決定する基礎例を含む解釈を $I_1, \dots, I_n \in X$ すると, $I_i \subseteq P'$ ($i=1, 2, \dots, n$) なる $P' \in X$ が存在し, P' においても g が追加される条件は真である. つまり $g \in \tau_3 \circ \tau_2(P') \subseteq \sup X$ であり, 矛盾する. また, $\sup X$ の一意性は τ_2, τ_3 のチャーチ・ロッサ性より自明である. $\square$

定理2より, $\text{fix}[\tau_3 \circ \tau_2](I) = \sup\{I, \tau_3 \circ \tau_2(I), (\tau_3 \circ \tau_2)^2(I), \dots\}$ なる関数 $\text{fix}[\tau_3 \circ \tau_2]$ を定義することができる.

[定義12] $\tau = \text{fix}[\tau_3 \circ \tau_2] \circ \tau_1$ とするとき, 関数 $\tau_U: U \rightarrow U$ を解釈 $I \in U$ に対して次のように定義する:

$\tau_U(I) = \begin{cases} \top & I = \top \text{ のとき, または } \tau(I) \text{ が履歴の条件を満たさないとき,} \\ \tau(I) & \text{その他のとき.} \end{cases} \quad \square$

τ_U はイベントを発生させ, そのイベントについての関係を導き出す関数である. τ_U を無限回適用することによってハードウェアの動作をシミュレートすることを考える. τ_U の性質を議論するまえに τ_1 において成り立ついくつかの性質を明らかにする.

[補題1] 解釈 $I \in 2HB$ において, Herbrand 領域 H の部分集合 H' を $H' = \bigcup_{s \in CUSo} \{t \mid t \in H_s, \text{Cas}(\text{beg}, t) \in I\}$ とする. 解釈 I が以下の条件(L1)を満たすとき, H' に含まれる任意の項 t_1, t_2 に対して, $\text{Cas}(t_1, t_2) \in I$ ならば $\text{Cas}(t_1, t_2) \in \tau_1(I)$ である.

ここで, 条件(L1)とは exp に現れる自由変数の H' の要素への任意の置換 σ と任意の $\text{exp} \in \text{SpecV, C}$ に対して, Cas の引数として $\text{lit}(\text{exp})\sigma$ に現れる項がすべて H' に含まれるならば $\text{lit}(\text{exp})\sigma \in I$ が成り立つことである.

(証明) 解釈 I が条件(L1)を満たし, $\text{Cas}(t_1, t_2) \in I$ ($t_1, t_2 \in H'$) かつ τ_1 により $\text{Cas}(t_1, t_2)$ なる基礎例が I に追加されたと仮定する. つまり, $m_I(\text{cond}(\text{exp})\sigma) = 1$, $\text{Cas}(t_1, t_2) \in \text{lit}(\text{exp})\sigma$ となる置換 σ が存在する. $\text{exp} \in \text{Spec}$ は syn2 の構文をとる表現である.

τ_1 の定義より, t_2 を除き Cas の引数として $\text{lit}(\text{exp})\sigma$ に現れる項はすべて H' に含まれる. また仮定より $t_2 \in H'$. したがって, 条件(L1)より $\text{lit}(\text{exp})\sigma \in I$. つまり $\text{Cas}(t_1, t_2) \in I$ となり仮定に反する. $\square$

[補題2] 補題1と同じ H' , 条件(L1)を満たす解釈 $I \in 2HB$ に対して, $\text{Bfr}(t_1, t_2) \in I$ ($t_1, t_2 \in H'$) かつ $\tau_3 \circ \tau_2(I) = I$ ならば $\text{Bfr}(t_1, t_2) \in \text{fix}[\tau_3 \circ \tau_2] \circ \tau_1(I)$ である.

(証明) $Bfr(t_1, t_2) \in fix[\tau_3 \circ \tau_2] \circ \tau_1(I)$ となるためには, τ_3 と τ_2 の定義より $Bfr(t_1, t_2)$ は $Cas(t_1', t_1), Cas(t_2', t_2)$ ($t_1', t_2' \in H'$) なる基礎例が $\tau_1(I)$ に含まれなければならないが, 補題1より τ_1 はそのような基礎例を I に追加しない. $\square$

[定理3] (L1)を満たす解釈 $I \in U$ に対して, τ_U はチャーチ・ロッサ性をもつ.

(証明) 表現 $exp \in Spec$ が自由変数に対する置換 σ によって, $m_I(cond(exp)\sigma) = 1$ であるとする. また σ により $cond(exp)$ の自由変数が置換されたHerbrand領域の項を $t_1, t_2, \dots, t_n$ とする. 補題1, 2より Cas, Bfr を述語記号として, 項 $t_1, t_2, \dots, t_n$ を引数にもつ基礎例は, τ_U によって I に追加されることはない. 一方 τ_U において基礎例追加の条件では, Eqv については負リテラルの出現を許さない. したがって, $m_{\tau_U(I)}(cond(exp)\sigma) = 1$ であり, チャーチ・ロッサ性が満たされる. $\square$

[定理4] 条件(L1)式を満たす解釈 $I \in U$ に対して $X = \{I, \tau_U(I), \tau_U^2(I), \dots\}$ とすると, $supX$ は一意に決まり, かつ τ_U の不動点である.

(証明) τ_U は基礎例を追加する関数であるから, $supX \subseteq \tau_U(supX)$. また, 定理2より, $supX \neq \top$ のとき, $\tau_3 \circ \tau_2(supX) = supX$ が示される.

$supX \supseteq \tau_U(supX)$ を証明する. $supX = \top$ のとき, 明らかに成り立つ. $supX \neq \top$ のとき, $g \in \tau_U(supX)$ $g \in supX$ なる基礎例 g が存在すると仮定する. $\tau_1(supX) = supX$ とすると, $\tau_3 \circ \tau_2(supX) = supX$ となり仮定に反するに反する. したがって, τ_1 によって基礎例が $supX$ に追加されたことになる. つまり $supX$ において, $cond(exp)\sigma$ が真となり, $g \in lit(exp)\sigma$ である置換 σ が存在する. exp の自由変数が y_i ($i = 1, 2, \dots, n$) であるとき, $I' = \{Cas(beg, y_i)\sigma, Eqv(y_i, t_i)\sigma \mid i = 1, 2, \dots, n. Eqv(y_i, t_i) \text{ は } cond(exp) \text{ に現れるリテラルのうち述語記号 } Eqv \text{ をもつもの}\}$ ($I' \subseteq supX$) は要素数有限だから, $I' \subseteq I''$, $I'' \in X$ となる I'' が存在する. 補題1, 2より $supX$ において $cond(exp)\sigma$ が真ならば, I'' においても, $cond(exp)\sigma$ が真となり $g \in supX$ に反する.

また解釈 I が (L1) を満たすとき, $\tau_U(I)$ も (L1) を満たす. したがって, 任意の X の要素も同様に (L1) を満たす. 定理3より $supX$ の一意性が示される. $\square$

[定義13] 仕様 $Spec$ の意味を $fix[\tau_U] \circ fix[\tau_3 \circ \tau_2]$ と定義する. $\square$

入力パターンから変換されるHerbrand解釈 I_p に対しては $fix[\tau_3 \circ \tau_2](I_p)$ が条件 (L1) を満たすので, $fix[\tau_U] \circ fix[\tau_3 \circ \tau_2](I_p)$ の存在は定理4により保証される. τ_U は τ_1 によってイベントを発生させ, τ_2 と τ_3 によって制約条件, 履歴としての条件から生じる関係をすべて求める. τ_U をくり返すことは, ハードウェアが動作することに対応し, τ_U の不動点はハードウェアが無限に動くことに対応する.

5. むすび

本論文ではハードウェアの機能的な動作を対象とした仕様記述体系を提案した. 提案する仕様記述では, ハードウェアの並列動作を考慮せず, 直列的に行わなければならない処理のみを記述する. 直列処理の制約が満たされる範囲で最大の並列度をもつハードウェアの動作が記述から得られる履歴である. また, 動作履歴は時系列によって表現されることが多い. 時系列において, 時間は系列を構成する要素間の

全順序関係によって表現される。本論文のモデルでは時間を半順序関係に拡張し、より柔軟な記述を可能にしている。

Top-Down設計において、仕様から得られる実現はそれ自身以後の設計段階での仕様である。したがって、実現もハードウェアモデルに従いモデル化され、仕様とそれから得られた実現は本論文の提案する同一の記述体系で記述可能である。実現の仕様に対する正当性は仕様で成立するイベント間の関係が実現においても成立することである。

意味となる関数の定義は、記述の記号シミュレーションの手続きとみることができる。記号シミュレーションでは、データに記号値を許すが、本論文での入力パターンの考え方に従えば、記号値を含む入力パターンは、値についての関係の一部しか含まない入力パターンであり、手続きはそのまま対象を記号値に拡張することができる。

謝辞 日頃から御議論いただく京都大学矢島研究室の諸氏に感謝いたします。

参考文献

- (1) J. V. Guttag: "Abstract Data Types and the Development of Data Structure", CACM 20, 6, pp. 396-404 (June 1977).
- (2) J. A. Goguen, J. W. Thatcher, E. G. Wagner: "Initial Algebra Approach to the Specification, Correctness and Implementation of Abstract Data Types", IBM Research Report RC 6487 (Oct. 1976).
- (3) 稲垣, 坂部: "多ソート代数と等式論理", 情報処理, 25, 1, pp.47-53 (1984-01).
- (4) 稲垣, 坂部: "抽象データタイプ", 情報処理, 25, 5, pp.491-501 (1984-05)
- (5) H. Ehrig, B. Mahr: "Fundamentals of Algebraic Specification 1", Springer-Verlag (1985).
- (6) 長尾, 淵: "論理と意味", 岩波書店 (1983).
- (7) C. Chang, R. C. Lee: "Symbolic Logic and Mechanical Theorem Proving", Academic Press (1973).
- (8) D. Bjørner, C. Jones: "Formal Specification & Software Development", Prentice-Hall (1982).
- (9) C. A. R. Hoare: "Communicating Sequential Processes", CACM, 21, 8 (Aug. 1978).
- (10) C. Hewitt: "Laws for Communicating Parallel Processes", Proc. IFIP Congress, Toronto (1977).
- (11) D. Scott: "Outline of a Mathematical Theory of Computation", Tech. Mono. PRG-2, Oxford Univ. Comp. Lab (1970).
- (12) 中島玲二: "数理情報学入門", 朝倉書店 (1982).
- (13) M. H. Emden, R. A. Kowalski: "The Semantics of Predicate Logic as a Programming Language", J. ACM, 23, 4, pp. 733-742 (Oct. 1976).
- (14) J. Staples, V. L. Nguyen: "A Fixpoint Semantics for Nondeterministic Data Flow", J. ACM, 32, 2, pp. 411-444 (April 1985).