設 計 自 動 化 61－2
（1992. 2. 21）

機能図データの多元版数管理法

小林一夫　若林春夫
NTT LSI 研究所

本論文は、CAD フレームワークの代表的サービスの１つである版数管理に関して、機能図データに適用する際の新手法について論じる。機能設計段階の仕様を図的に表現する機能図は、広範な抽象度をカバーする。そのため、１つの機能名で呼ばれる機能であっても、異なる設計制約に対応して構成の異なる論理で実現されたり、用途に応じて異なる機能図の記法で表現されることが多い。しかし、これまで発表されているシステムないし手法は、このような機能名の持つ多義性を扱う上での統一的な管理手段としては充分ではなかった。そこで、従来の版数の概念を拡張して複数の属性で機能図データの版数を規定し、任意の版を動的に選択可能な版数管理手法を開発した。

Multidimensional Version Management of Design Data for Schematic Functions

Kazuo KOBAYASHI and Haruo WAKABAYASHI

NTT LSI Laboratories

3-1, Morinosato Wakamiya, Atsugi-Shi Kanagawa Pref., 243-01, JAPAN

This paper describes a version control technique for manipulating design objects specified by schematic diagrams in the functional design stage. These schematic diagrams cover the design objects at extensive levels of abstraction. Some functions referred by the same identification name may therefore be implemented as variable logic configurations corresponding to different design restrictions. Functions are also represented by description styles suitable for each purpose. Several version management systems and methods have been reported, but they do not have enough facilities to manage the aforementioned diverse meanings of the identification name. This paper therefore proposes a version management technique that supports an open configuration based on an extended version concept.

1 はじめに

ハードウェアの機能仕様を入力とする論理合成技術の実用化が進められるのに合わせて、機能仕様を図的に表現して設計ドキュメント（機能図）の作成と計算機入力を同時に可能とする図式表現技術の研究・開発も盛んに行なわれるようになってきた。計算機入力されるデータ*を、機能設計支援システムの共通の設計データとして管理するには、それらの識別名である機能名の持つ多義性を考慮する必要がある。

すなわち、機能図は、制御（データ）フロー図のように方式設計仕様から書き下ろされた直後の抽象度の高いものから、論理回路図に近い機能ブロック図のようにセルライブラリの素子と1対1に対応付けられるような抽象度の低いものまでの広範な抽象度を対象とする。そのため、1つの機能名で呼ばれる機能であっても、異なる設計制約に対応して構成の異なる論理で実現されたり、用途に応じて異なる記法で表現されることが多い。従来の版数も、同一機能名で呼ばれる設計オブジェクト同士を管理する役割を持っている。しかし、その管理機能は修正された論理構成を個別に識別することに限定されており、この機能名の持つ多義性全体をカバーするものではなかった。

そのため、ここでは、従来の版数の概念を拡張した多元版数をもとに、複数の設計オブジェクトを矛盾なく管理する手法を開発した。

本論文は、この新しい版数管理法についての基本概念を報告するものである。以下、2章では、まず、版数管理の必要性を整理し、機能図データ管理上の課題を明らかにする。次に、3章では、その課題を解決するために開発した版数管理手法を述べる。最後に、4章で、版数に関する情報を保持する設計オブジェクトの構成法を概説する。

2 背景

2.1 版数の定義

ある機能を実現する回路構成の仕様を例にとると、従来の版数は、その仕様が修正される毎に、元の仕様と修正後の仕様とを識別するために付与される識別名として扱われれてきた。この従来の概念の版数を付与した設計オブジェクトは、1つの機能名で呼ばれるが異なる情報を持つ設計オブジェクトの一例と見ることができる。そこで、版数の概念を”同一の機能名を持つが、異なる設計オブジェクトの識別名”に拡張すると、機能図で記述された設計オブジェクトを識別する主要な版数には以下がある†。

- Revision: 設計オブジェクトの更新結果を識別する。狭義の版数が、これに該当する。図1は、機能名 ADDER に機能を使って版数の概念を説明するものである。図中の Revision 配下で、上の矩形内は、ADDER の詳細仕様として、フルアダーを必要なビット数だけ並べて構成したものを表す。下の矩形内は、それを修正して桁上げのための接続を追加したものを表す。
- View: 抽象度の異なる仕様の表現を識別する。図1の View の例の2つの矩形内は、ADDER を機能ブロック図と設計記述言語 HDL で表したものを示している。この例のように、見た目が明らかに異なる記法の場合、異なる View として識別されるのが分かり易いが、一般的には、View と記法とは等価とは言えない。
- Alternative: 1つの仕様に対する異なる実現結果を識別する。図1の Alternative の例は、2つの矩形内ともフルアダーを並べて構成するところまでは同じであるが、桁上げの方法が、一方は、直列に上位の桁方向に伝搬させるのに対して、他方は桁上げ回路によって実現している。このような Alternative で識別される設計オブジェクトは、各々固有の設計トレードオフを持つ。

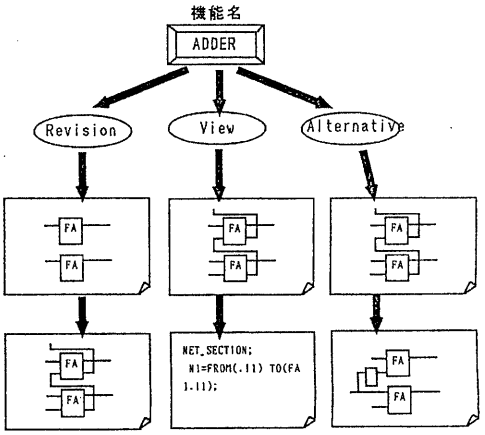

図 1: 各種の版数の概念図

*紙の上で扱うものと計算機入出力の対象となるものを区別して、後者を設計オブジェクトと呼ぶ。

†以下では、”版数”はこの拡張した概念のものを指す。従来と区別が必要な場合には、従来を狭義の版数、拡張を広義の版数と呼び分ける。

2.2 版数管理上の課題

2.2.1 版数管理の構成技術

版数管理では上記の版数を個別の概念として扱う必要がある。それを構成する技術は、文献 [1, 2, 4] を元に、以下のように整理できる。

1. モジュール構成：ブロック図では、ある部品‡の内部仕様は、より具体化された部品（サブ部品）の接続で表される。さらに、サブ部品も更に詳細な部品の組合せで記述される。部品とそれを構成するサブ部品との組合せをモジュール構成と呼ぶ。モジュール構成を管理する機構ないし方法を、構成管理と言う。このモジュール構成の指定の仕方に、次の3種類がある。

   静的構成 部品内でそれを構成するサブ部品の版数を指定する。

   動的構成 設計オブジェクトの版数が、それらの持つ属性に依存しない外部条件で動的に決まる。

   オープン構成 動的構成の一種であるが、設計オブジェクトのもつ属性を外部条件として指定できる。

‡機能の外部仕様を表すもの。

2. 管理階層：設計オブジェクトには、設計データの実体をもつものと、それらの集合を管理する管理オブジェクトがある。管理オブジェクトは、ファイル管理システムのディレクトリに相当するものである。上記のモジュール構成は、設計データの実体を持つ設計オブジェクトの間の構成関係である。それに対して、管理階層は、管理オブジェクト相互の階層関係を表す。管理階層については、どのような階層構成をとるか、管理階層自身に版数を持たせるか、また、設計オブジェクトの版数をどの管理オブジェクトに持たせるかなどによって種々の構成方式のバリエーションがあり得る。

3. 版数履歴：各版数内での版数の親子関係を指す。この版数履歴を制御することを版数制御といい、以下の形態がある。

   線形シーケンス 親となる版数と子が1対1に対応する。

   木構造 1つの親に対して複数の子が存在する。ある子に対する親は唯一である。

   グラフ構造 1つの親に対して複数の子が存在するが、ある子に対する親も複数あり得る。

4. 版数階層：前記の3種類の版数相互の階層関係を指す。たとえば、ある部品に対して複数の Alternative を見る場合には、複数の Alternative が View の下位階層である階層構成となる。一方、ある部品に対して、その詳細化過程を見る場合には、View が Alternative の下位階層である階層構成となる。

2.2.2 機能図での課題

機能図として記述される設計オブジェクトは、システムレベルの仕様から詳細なタイミングを考慮した仕様までの広範な抽象度を扱う。そのため、版数管理を実現するには、以下のような特性を考慮する必要がある。

1. トップダウン設計指向

   大規模な装置ないし LSI チップでは、抽象度の異なる機能図を各種組み合わせて仕様を記述する。その際、ある抽象度の部品に対する内部仕様として複数の Alternative[§] を設計することがある。

   それらを、処理系に引き渡すには、処理系が受け付けられる HDL に変換する。その HDL は、部品毎に唯一の Alternative が選択されたモジュール構成を持つ。変換された HDL は、論理が正しいことが確認されると設計の評価に使われる。そして、当初の目標性能ないしコストを見たさなければ、選択基準を変えて Alternative を選び直す。この過程が繰り返されて（設計の試行錯誤段階と呼ぶ）目標を満たすと、次のレイアウト・実装設計に引き渡される。

   その際、選択基準はモジュール階層全体に同一基準であることは少ない。たとえば、遅延時間がクリティカルな部品に対しては遅延を最小とする Alternative を、それ以外の部品に対してはコストを最小とする Alternative を選択するような基準が適用されるのが普通である。この Alternative を状況に応じて任意に選択可能とするモジュール構成（すなわちオープン構成）が、第一の課題である。

   さらに、版数履歴上の親子の対応つけの課題がある。すなわち、試行錯誤段階で、設計オブジェクトの更新が生じた場合、いつも直前の Revision から次の版が作られるとは限らない。木構造やグラフ構造の場合、親と子を対応つけるには、連番の前後のような簡単な仕組みでは対応できない。

2. 記法間の重層的関係

   機能図の記法は、機能ブロック図、状態遷移図など多様であり、それぞれ異なるシンボル、異なる論理的意味つけを持つ。そのため、これらを計算機処理の対象として編集・入力する操作も一様ではないことが多く、各記法専用のエディタが使われる。前述のように、用語の定義に厳密に従うと同一記法であっても異なる View として扱われるものが生じる。しかし、混合記述可能な HDL[11, 12] の処理系（論理合成ツールやシミュレータ）は、抽象度の異なる記述も見かけ上は等しく処理可能である。むしろ、記法の異なる記述を一律には処理できない場合が多く、抽象度の差よりも、記法の差の識別がより重要となる。

   さらに、機能を段階的に詳細化していく過程では、用途に応じて記法が使い分けられることが多く、異なる記法間でモジュール構成上の親子関係を持つことがある。 そのため、View 毎に閉じたモジュール構成では対応できず、版数階層の管理が必要となる。

2.3 関連する研究の概要

ここでは、特長的な版数管理法を採用している例を概説し、上記の観点で評価する。

GARDEN[1] の特徴は、特有の版数名（番号）付与によって版数履歴上の親子関係を表すところにある。その方法では、設計方針の変更とその更新とに設計データの修正を大分類する。そして、設計オブジェクト (VS_{xx}) に旧版数名と新版数名の2つを持たせ、それによって版数履歴上の親子関係がわかるようになっている。 このシステムは、*Design, View-Group, View*[¶] の3階層の管理階層を持つ。このうち、最下位の管理階層である *View* には、Alternative の特定の View に相当する実際の設計データが格納され、HDL、Layout, 混合階層記述の3種類の View が許される。 *View* 内部の部品の詳細仕様として、管理階層上の任意のノード（設計オブジェクト）を参照可能とすることにより、モジュール構成を実現している。

一方、Dittrich らの開発中のシステム [4] は、モジュールの動的構成のために包括的参照機構と版数の群化機構を備えている。すなわち、各設計オブジェクトは、Alternative と Revision の2段階で群化され、その実体とリンクで結んで関係付けられる。動的構成の定義は環境とよばれ、設計オブジェクト名と群名及び版の参照名が設定される。設計オブジェクトのモジュール構成において、特定の設計オブジェクト名で

[§] 厳密には、それによって識別される設計オブジェクト。以下では、混乱しない限り設計オブジェクトは省略する。

[¶] 版数の View と区別するためフォントを変えて示す

アクセスすると、その包括的参照機構によって、環境で定義された版が選ばれる。

しかし、これらの方法では、モジュール構成上の下位の設計オブジェクトについては、1機能名に対して唯一の実体を選択するため、状況に応じて異なる実体を選択することができない。

Katz らの開発中の版数サーバ [2, 7] は、オブジェクト間の関係を陽に指定する機構と、レイヤ概念によりモジュール構成を動的構成可能とするのが特長である。それによれば、版数つきの設計オブジェクトをレイヤで区切り、版数の探索順をレイヤ番号で指定する。その指定順に応じて、予め各レイヤに割り付けられていた版が目的の設計オブジェクトのアクセス可能なものとみなされる。モジュール構成は、有向非循環グラフで表される。View[‖]にまたがる混合構成はサポートしていない。 版数履歴は、専用の構造オブジェクトで管理される。同じ起源を持つオブジェクトどうしが Alternative として扱われる。

拡張の一貫として、fast, low power などの属性を設計オブジェクトに付与してオーブン構成を実現する方法にも言及されているが、具体的な実現法は提案されていない。

DDM[3] では、モジュールの動的構成を管理する専用の設計オブジェクト（Workspace と呼ばれる）を設け、モジュール構成のアクセスに階層的要素を持ち込んだ。すなわち、Workspace 同士に親子の階層関係を持たせることができる。各 Workspace は、独立にカレントの Revision を管理する。特定の Revision への参照を持たない Workspace が使われる時には、その親に遡ってカレントの Revision を見つける。版数履歴には、有向非循環グラフ構造をとることができるが、Revision 名には、単調増加する数値が使われる。

しかし、View,Alternative も考慮したモジュール構成は扱われていない。

ICD[5],DMS[8] では、線形シーケンスの版数履歴をとっている。設計オブジェクトに Revision の他に更新状態を管理する Revision Status を持たせている。この Revision Status は、更新中の設計オブジェクトで actual 状態になり、モジュール構成に現れる。モジュールを構成する設計オブジェクトを直接に版数名で指定せず、actual 状態という属性で指定しているので動的構成の一種と見なせる。

Cadence Framework は、市販の代表的なものであるが、その次世代では、モジュール構成に関して静的構成と動的構成をサポートする [6]。モジュール構成は階層的に定義可能であり、設計オブジェクト毎に機能名と View 名を指定する。

Octane[9] は、初期の CAD フレームワークである Oct[10] をベースに版数管理機能を拡張したものである。部品とモジュールが基本構成要素であり、同じプロテクション状態にある部品の集合において、版数履歴上の唯一の Revision からモジュール構成が作られる。

以上はいづれも単純な版数管理しかサポートしていない。

‖原論文では Type

3 版数管理法

3.1 モジュール構成

3.1.1 構成管理

機能名の他に前述の3種類のタイプの版数名によって識別する場合、3次元空間上の1点を指定する必要がある。すなわち、＜機能名＞＋＜Revision 名＞＋＜View 名＞＋＜Alternative 名＞の情報が必須である。オーブン構成を実現するには、各版数名を直接指定する替わりに、設計オブジェクトに依存する情報（版の属性）によって特定の設計オブジェクトの版を指定できるようにする必要がある。

ここでは、3種類の版の属性を統一的に扱えるように、次のように一般化した。

すなわち、各タイプの版毎に版数名と属性値を対で持つ。属性値は、複数の値からなり、版の種別毎に固有の値と属性値の使われ方を制御する値とに細分化される。前者を固有属性値、後者を属性制御値と呼ぶ。この版数名と属性値は部品の内部仕様を表す設計オブジェクト（図面）毎に宣言される。

一方、モジュール構成上の特定の版は、構成定義によって選択する。構成定義は、活性表示フラグと選択条件とからなり、特定の部品に対して定義される。活性表示フラグは、それが ON ならば構成定義が有効となり、OFF ならば無効と扱われる。選択条件は、複数の対象から特定の版数を選択する条件を規定する。定義された内容は、モジュール構成上の、該当の部品を構成するサブ部品の設計オブジェクトの選択に有効となる。サブ部品も構成定義を持つことができる。該当のサブ部品の構成定義の活性表示フラグが ON ならば、サブ部品配下の選択には、サブ部品の構成定義が優先する。

モジュール構成上で指定された部品がそれ自身の構成定義を持たない場合には、それより上位に遡って構成定義を探す。最上位には暗黙指定の構成定義を用意し、いづれにもない場合にはこれが使われる。

次に、これらを具体的な例で説明する。

3.1.2 属性

Revision には、版数名としては更新番号が付与される。例えば、v1.0。その固有属性値のうち、設計状態を表す Dstatus は、次の3値のいづれかをとる。すなわち、設計状態値 current は、該当の設計オブジェクトが現在編集中で、今後も更新があり得ることを表す。設計状態値 release は、該当の設計オブジェクトは凍結されたもので、変更不可であることを表す。該当の設計オブジェクトが上記のいづれでもないことは、設計状態値 backup で表す。

更新された日付けは、固有属性値 Date で表される。例えば、911202 は、１９９１年１２月２日の更新を意味する。属性制御値 Switch は、任意の数値をとる。

View には、版数名としてはエディタに依存した記法名が付与される。例えば、機能ブロック図エディタでは、block。その固有属性値 Level は、設計オブジェクトの抽象度を表す任意のテキストをとる。例えば、動作記述レベルなら、behavior, 構造記述レベルならば structure など。

Alternative には、版数名としては実現名が付与される。例えば、with_cla。その固有属性値には、たとえば次の3タイプがある。いづれも任意の数値をとる。

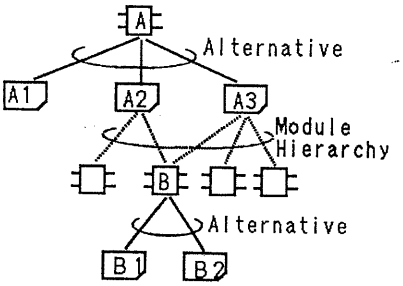

図 2: 属性値による選択の説明図

Speed 値 遅延時間の相対値を表す属性値。値が小さいほど遅延時間が短い。

Area 値 面積（ゲート量）の相対値を表す属性値。値が小さいほど面積が狭い。

Power 値 消費電力の相対値を表す属性値。値が小さいほど消費電力が少ない。

以下の説明に次の記号を使う。すなわち、図面 D の属性値 D は、以下のように表される。

$$D = \begin{pmatrix} \alpha_s & \alpha_d & -1 & \alpha_w \\ \beta_l & -1 & -1 & \beta_w \\ \gamma_s & \gamma_a & \gamma_p & \gamma_w \end{pmatrix}$$

ここで、

α_s :Dstatus 値 α_d :Date 値 α_w :Switch 値

β_l :Level 値 β_w :Switch 値

γ_s :Speed 値 γ_a :Area 値 γ_p :Power 値 γ_w :Switch 値

-1 は未使用の値を意味する。

図 2 の例で説明する。この例では、設計オブジェクト A,B が部品で、A_x,B_x が各々の内部仕様であり、部品と内部仕様の繰り返しで有向非循環グラフ構造を形作っている。各図面の属性値として以下のように宣言する。

$$A_1 = \begin{pmatrix} current & 911202 & -1 & 10 \\ structure & -1 & -1 & 20 \\ * & 20 & \_ & 1 \end{pmatrix}$$

なお、以下では説明を簡略化するため、Revision と View の属性値は省略する。各タイプの版の属性値は独立に評価されるため、一種類のタイプの版の属性値の扱いの繰り返しで複数の属性値が処理される。

$$A_2 = \begin{pmatrix} 5 & 50 & \_ & 1 \end{pmatrix}$$
$$A_3 = \begin{pmatrix} 10 & 40 & \_ & 1 \end{pmatrix}$$
$$B_1 = \begin{pmatrix} 10 & 40 & \_ & 0 \end{pmatrix}$$
$$B_2 = \begin{pmatrix} 5 & 50 & \_ & 0 \end{pmatrix}$$

A_x 系の図面の Alternative の属性値 Switch は 1 であるのに対して、B_x 系のものは 0 が宣言されている。さらに、$*$ は don't care を、"_" は未宣言を表す。

3.1.3 構成定義

モジュール構成上の特定の設計オブジェクトのアクセスには 2 種類のモードを用意する。すなわち、デフォルトモードと明示モードである。

明示モードでは、特定の設計オブジェクトは機能名と版数名を陽に指定する。すなわち、静的構成である。例えば、adder:v1.0:block:with_cla。ここで、adder は機能名である。以降の ":" は版数名の区切り記号であり、版数名は Revision, View, Alternative の順で指定されている。

一方、デフォルトモードはオープン構成を扱うモードである。このモードでは、版数名の替わりに属性値が特定の設計オブジェクトの選択に使われる。

構成定義には、次のような属性値の判定条件を指定する。

```
BEGIN Alternative
  IFDEF Switch == 1
    Area >= 50
    ELSE
    Speed <= 10 & Area < 50
    ENDELSE
  ENDIF
END Alternative
```

この判定条件により、Alternative の属性値では、Switch が 1 の場合、$Area >= 50$ を評価し、さもなければ、$Speed <= 10$ & $Area < 50$ を評価する。その結果、評価に適合した、即ち、Switch == 1 で $Area >= 50$ の条件を満たす属性値をもつ図面 A_2 と $Switch! = 1$ で $Speed <= 10$ & $Area < 50$ の条件を満たす属性値をもつ図面 B_1 が選択される。構成定義を利用した操作の一例として、部品 A から B に向かって階層を下る方向にブラウジングをとりあげる。その場合、機能名 A,B の指定だけで、順次、図面 A_2, 図面 B_1 が表示される。

3.1.4 版数の異なる設計オブジェクトの選択

上記の構成定義は、指定された設計オブジェクトを頂点とするモジュール構成上の下位の設計オブジェクトの版の選択に作用する。そのため、これだけでは下位に同じ機能名の設計オブジェクトが複数存在した場合、全て同一の版が選択されることになる。

しかし、現実の設計現場では、同じ機能名の設計オブジェクトでも、それが使われる場所によって異なる Alternative が選ばれるのが普通である。

これに対応する仕組みが、モジュール構成上の任意の設計オブジェクトへの定義と、先の 3.1.1項に示したモジュール構成の選択手順である。これによって、1 つのモジュール構成上に同一機能名で版数の異なる設計オブジェクトを指定できる。

この例を図 3 に示す。この例は、設計オブジェクト B をトップとしてその下位の階層の C、D がサブ部品として機能モジュール A を持っている。設計オブジェクト B、C にそれぞれ構成定義として $a > 1, a = 1$ が指定されている。一方、A の版として A_1, A_2, A_3 があり、それぞれ属性として $a > 1, a = 1, a < 1$ が宣言されている。

このような状況で、モジュール構成上の A としては、各々

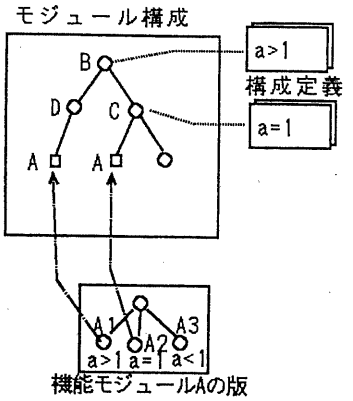

図 3: 同一機能名で版数の異なる設計オブジェクトの指定

次のものが選択される。すなわち、設計オブジェクトDのサブ部品のAの選択に有効な構成定義は、設計オブジェクトBの$a > 1$である。したがって、機能モジューAの版の内、判定条件を満たすA_1が選択される。設計オブジェクトCのサブ部品のAの場合には、有効な構成定義は設計オブジェクトCの$a = 1$である。したがって、機能モジュールAの版の内、判定条件を満たすA_2が選択される。

3.1.5 構成定義の版数

任意のモジュール構成を指定する目的には、"X月X日のモジュール構成"、"ある機能のモジュール構成"、"誰それの組み立てたモジュール構成"を見たいという要求に答えることがある [9]。このうち、特定の日付けに関連するモジュール構成は、この定義自信が持つ Revision と属性値を利用することによって可能となる。残りの2者の要求については、現状は未サポートであるが、前者の方法の延長線上の技術で実現可能である。

ところで、構成定義の Revision は次のような性質を持つ。すなわち、属性値が current のものが編集可能であり、release でかつ活性表示フラグが ON のものが有効と見なされる。

release, backup のものを編集コマンドで呼び出すことで current になり、それを現用として格納することにより release となる。この時に、直前までに release であったものは、backup に変更される。release は、同一設計オブジェクトの版数履歴上の唯一のものに付与するので、適用される構成定義のユニーク性が保たれる。

特定の日付けのモジュール構成を見るには、属性値の内の Date 値を使う。これを図 4 の例で説明する。同図では、設計オブジェクトBに指定されている構成定義が、3つの Revision を持つ。現用の構成定義は、属性値が release の$V_{1.2}$である。これのかわりに、9月20日当時のモジュール構成をみるには、専用のコマンドによって選択条件を、$Date =< 910920$と指定する。その結果、条件を満たす$V_{1.1}$が現用となり、目的のモジュール構成が選択できる。

しかし、このような古い版を再現する上で次のような問題が生じる。すなわち、データベースの有限な空間を有効的に使うためには、限られた数の版だけを保存し、他を消去することが行なわれる。そのため、設計の過程で古い版の設計

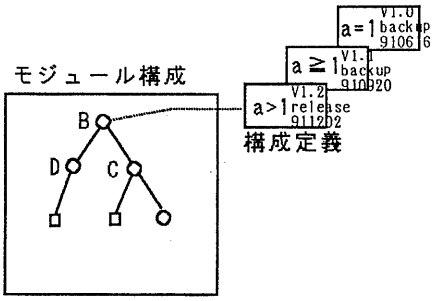

図 4: 特定の日付けのモジュール構成の指定

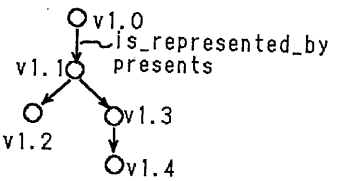

図 5: 版数履歴

オブジェクトがデータベースから消去されていて再現不可能な状況が生じ得る。

特に、ここで述べてきたような多元版数管理を行なうと、1つの機能当たりの版の数が多くなるので、この問題は重要である。

これに対処するため、構成定義によって参照される設計オブジェクトに、凍結カウンタを設ける。このカウンタは、参照元の構成定義の定義と消去に同期して値が変化する。すなわち、1つの設計オブジェクトは、複数の構成定義によって参照されるので、構成定義が定義される毎に加算され、消去されると減算される。凍結カウンタを持つ設計オブジェクトを消去する時に、そのカウンタ値を見て、0なら無条件に消去できる。0以外ならば、その設計オブジェクトを選択する構成定義が残っているので消去には注意を要する。

3.2 版数履歴

View と Alternative については、同じタイプの版同士では親子関係を持たず、Revision のみ管理する。この親子間の関係は、文献 [2] と同様に設計オブジェクトの関係を表すリンクで陽に指定する。

図 5 の例で説明する。図は、1つの設計オブジェクトの5つの Revision の版数履歴を表している。各設計オブジェクトには、$V_{1.0}$から$V_{1.4}$の Revision 名が付与されている。ここでは、各設計オブジェクトが持つ関係名"is_represented_by"と"presents"とが親子関係を表す。前者が子から見た親を、後者が親から見た子を指定する。このように、設計オブジェクトの関係を陽に持つことにより、版数名に依存せずに木構造ないし有向非循環グラフで親子関係が管理できる。

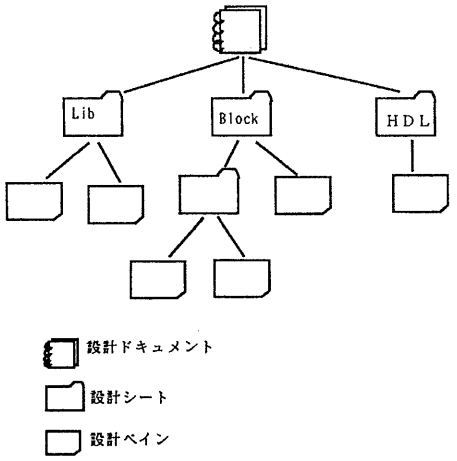

図 6: 階層管理にかかわる関係

4 設計オブジェクトの構成

4.1 管理階層

設計データの実体を構成する設計オブジェクトは、部品とその内部仕様である。機能図では、その内の内部仕様（図面）は、ページ単位に区切られる。そのため、ページを設計データの実体の管理単位として、図面ページと名付ける。

この図面ページを管理するには、次のような機能を持った管理オブジェクトが必要になる。すなわち、第一に部品とその内部仕様を対応つけるうえで、部品対応に図面ページの集合を管理するものが必要である。さらに、記法の差を区別して扱うには、特定の View 毎に図面ページの集合を管理する必要がある。また、特定の設計プロジェクトで扱う機能図を一括して管理する主体もいる。そのため、次の 3 種類の管理オブジェクトを用意した。

設計ドキュメント 機能図全体を管理するもので、管理階層のトップに位置する。

設計シート 特定の View を管理する。設計ドキュメントは 1 種類以上の設計シートの集合である。

設計ペイン 設計シートを構成する図面に対応し、図面ページの集合を管理する。

管理オブジェクトは、図 6 に示すように、1 つの設計プロジェクトに対応して設計ドキュメントをトップとする階層を形作る。設計ドキュメントの配下には、View に対応する設計シートが位置する。図中、Block, HDL と名付けられたものが、それぞれ機能ブロック図、HDL 記述で記述された設計オブジェクトの管理元となる。ライブラリを管理する Lib および ViewGroup（後述）にも設計シートを用いる。設計シートは、各記法の設計オブジェクトを設計ペインを使って管理する。

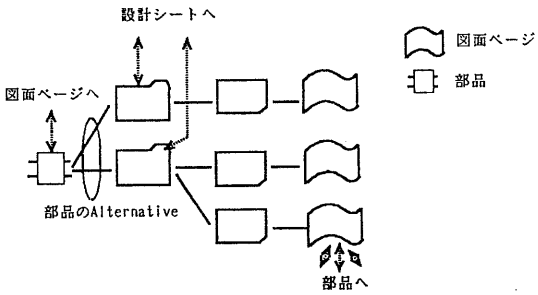

図 7: 版数階層にかかわる関係

4.2 版数階層

2.2.2項で述べたように、機能図では、1 つの部品の内部仕様が異なる記法で記述されることがある。また、異なる記法間でモジュール構成上の親子関係を持つことがある。そのため、部品と図面ページの間に、同一の View 毎に図面ページをまとめるものが必要になる。View の集合を管理する ViewGroup をもうけて、この配下の View が、Alternative を管理できるようにする。

これにより、部品の配下で全ての Alternative が管理でき、部品とその内部仕様の Alternative の関係が自然に表せる。

図 7 に版数階層にかかわる関係を図示する。部品と、図面ページとは、間に設計シートと設計ペインを挟んで関係付けられる。1 つの設計ペインは、部品の 1 つの Alternative を代表する。それらを記法毎にまとめて管理するのが、設計シートで表される View である。前記の管理階層上の View を代表する設計シート ViewGroup とは、この View を介して関係付けられる。

4.3 モジュール構成

1 つの機能名で呼ばれる設計オブジェクトには、上記のように部品、設計シート、設計ペインと図面ページとがある。これらが 1 セットとなってモジュール構成上の 1 ノードとなる。図 8 に示すモジュール構成図の中で、1 点鎖線で囲んだものがノードを表す。モジュール構成上のノード間のリンクは、ノード内の部品と図面ページが持つ。また、構成管理に必要な情報は、それぞれ以下の設計オブジェクトが持つ。すなわち、構成定義は、部品が持つのに対して、属性と凍結カウンタは、設計ペインが持つ。

4.4 版数履歴

図 9 に図面ページの版数履歴にかかわる関係を図示する。親子関係を表す関係名（ 3.2節参照）は、図面ページ相互が持つ。 管理元の設計ペインとのリンクは、current の設計状態値を持つ図面ページとの間に張られる。その時の関係名は、設計ペイン側が "consists_of"、図面ページ側が "is_a_part_of" を持つ。

設計ペインも同様に版数履歴を持つ。

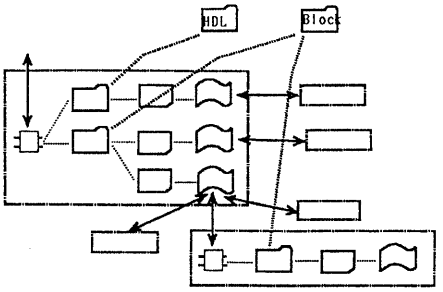

図 8: モジュール構成にかかわる関係

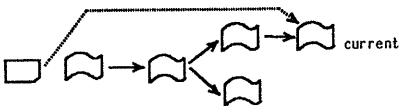

図 9: 版数履歴にかかわる関係

5 むすび

従来の版数の概念を拡張した多元版数管理法と呼ぶ手法について基本概念を報告した。版数管理を扱う主なツールには、設計オブジェクトを作り出すエディタや処理系と、それらを管理する管理機構とがある。現在、前者については、文献 [13] で報告したものに、ここで述べた概念にあわせた改良を進めており、後者についても文献 [14] のものを核としたシステム化を進めている。

謝辞

日頃、有益な助言を頂く LSI 研究所 設計システム研究部 安達徹主幹研究員に感謝します。

参考文献

[1] Flávio R. W., Arnaldo H. V. : Design Version Management in the GARDEN Framework ; 28th DA Conf. pp.704-710 1991.

[2] R. H. Katz, R. Bhateja, E. E. Chang et al. : Design Version Management ; IEEE Design & Test pp.13-22 Feb. 1987.

[3] S. Banks,C. Bunting, R. Edwards et al. : A Configuration Management System in a Data Management Framework; 28th DA Conf. pp.699-703 1991.

[4] K. R. Dittrich, R. A. Lorie : Version Support for Engineering Database System; IEEE Trans. on SE. Vol.14, No.4 pp.429-437 Apr. 1988.

[5] T. G. R. van Leuken, P. van der Wolf : The ICD Design Management System; ICCAD'85 pp.18-20 1985.

[6] L. C. Liu, P. C. Wu, C. H. Wu : Design Data Management in a CAD Framework Environment;27th DA Conf. pp.156-161 1990.

[7] D. Gedye, R. Katz : Browsing the Chip Design Database; 25th DA Conf. pp.269-274 1988.

[8] P. van der Wolf, T. G. R. van Leuken : Object Type Oriented Data Modeling for VLSI Data Management;25th DA Conf. pp.351-356 1988.

[9] M. Silva, D. Gedye, R. Katz et. al. : Protection and Versioning for OCT;26th DA Conf. pp.264-269 1989.

[10] D. S. Harrison, P. Moore, R. L. Spickelmier : Data Management and Graphics Editing in the Berkeley Design Environment;ICCAD'86. pp.24-27 1986.

[11] 電子協編 : LSI 設計用記述言語 第 1.0f 版 1991.

[12] D. R. Coelho(片桐徹訳) : The VHDL Hand Book 大倉商事 1990.

[13] 小林、若林、脇村 : トップダウン設計向き機能図記述言語; 信学会論文誌 Vol. J74-A No.2 pp.256-264 1991

[14] 高原厚 : 分散設計環境に置ける設計データ管理の一手法; 第 4 3 回情処全大予稿集 1R-5 1991