6J-4　入出力に図形表示を用いた機能レベル論理設計支援システム

藤田昌宏　田中英彦　元岡達
（東京大学　工学部）

1．　はじめに

デジタルシステムは、ＶＬＳＩ等の出現とともに近年、益々大規模化・複雑化している。このため、機能レベルをサポートする論理設計用ＣＡＤシステムの必要性が高まっている。そこで本研究では、つぎの点に重点をおいてＣＡＤシステムの検討・試作を行なった。

①　階層・構造化設計を行なう。

②　システムの記述は、構造（structure）＋動作（behavior）で行なう。

③　動作の記述には、ハードウェアだけでなくソフトウェアの記述もできるようにする。

④　入出力に用いるグラフィックエディタは、操作数の減少とプログラムの作りやすさをともにみたすようにする。

⑤　実装段階ＤＡとのインターフェイスを考えておく。

構造情報はＨＳＬを中心にデータベースとしてもつようにし、シミュレータ、ベリフィケータを加えてＣＡＤシステム全体の構成は、Ｆig．１のようになる。今回は、グラフィックエディタ、シミュレータを中心に実験システムを製作した。　以下に、各点について順に述べていく。

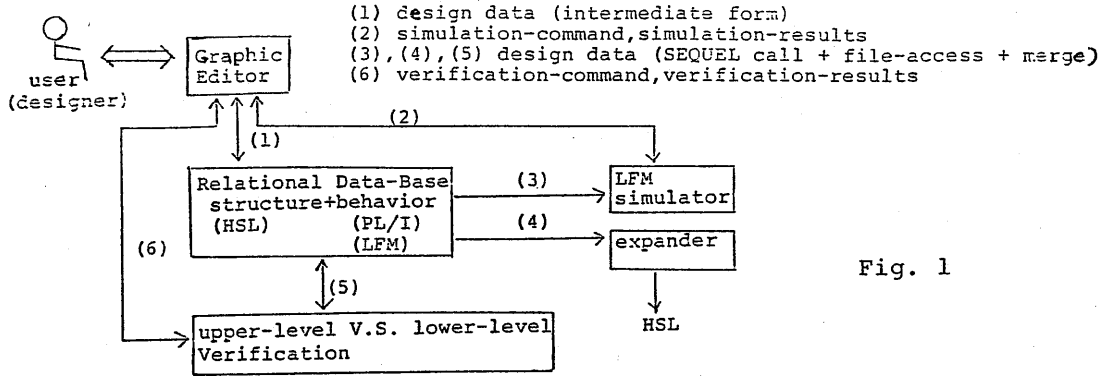

Fig. 1

2．　設計手法＜１＞

複雑・大規模なシステムを設計するためには、どうしても全体をいくつかに分割して、考える範囲を限定しなければ扱いきれない。このため、階層・構造化設計が向いている。また、階層の各レベルにおいて設計の確認を行なってからつぎに進むのがよい。そこで本システムでは、各レベルにおいて構造の記述と動作の記述をともに行ない、展開することなしにシミュレーションが実行できるようにしている。また、動作の記述はソフトウェア的なalgorithmic な記述もハードウェアを充分意識した記述も行なえるようにしている。（Ｆig．２）

一方、ハードウェアの記述だけでなく、要求仕様定義サポートツールともつながっていることが望ましい。本グラフィックエディタには、ＳＡＤＴ技法サポートも可能なように汎用性をもたせている。

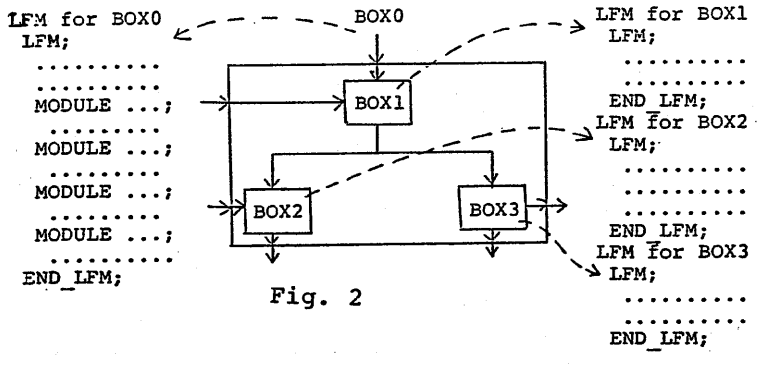

Fig. 2

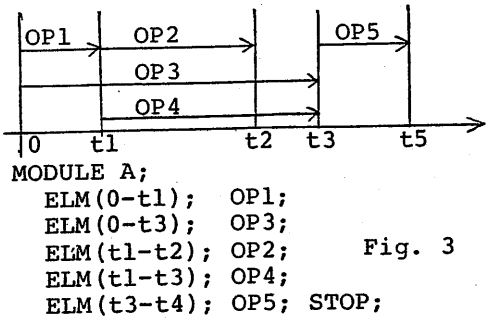

Fig. 3

３．　機能レベルハードウェア記述言語：ＬＦＭの拡張的使用

ＬＦＭ＜２＞は、Ｆｉｇ．３のように動作に要する時間を任意の細かさで記述できるハードウェア記述言語であり、物理実体の宣言部と各モジュールの動作の記述部に分かれる。また、function宣言することにより、その部分を通常のＰＬ／Ｉで記述することができ、ソフトウェアの記述にも用いることができる。このためあるモジュールを設計する際、まずソフトウェア的に使用してアルゴリズムが正しいことを確認した後、ハードウェアとして記述し設計を進めていくことが可能である。

４．　グラフィックエディタ

ソフトウェアの負担をふやさず、操作数をへらすため、box とarrow を３次元的に表現した。＜３＞ユーザは、ハードウェアによる回転を用いてみやすい方向からみることができる。（Ｆｉｇ．４）また、Ｚ軸を利用してさまざまな情報を入れることができる。各box やarrow にその内容を記述してあるファイルをつけているため、他の用途、例えばストラクチャーチャート等のサポートツールとしても利用できる。

５．　使用例

本システムを用いた設計の様子をＦｉｇ．５に示す。図から分かるように、グラフィックエディタによる構造の入力と、テキストエディタによる動作の入力を行ない、それらを階層的に管理している。シミュレーションは、各box ごとでも、表示しているモジュール全体でも行なうことができる。

６．　今後の課題

今後は、関係データベースを中心とし、シミュレーションだけではなくベリフィケーションも加え、Ｆｉｇ．１のシステムに発展させる予定である。

＊　　参考文献

＜１＞『トップダウンによるＣＡＤシステムの設計』　藤田　他、　第22回情報処理学会全国大会
＜２＞論理装置の汎用機能シミュレーションシステムーＬＦＭ』　情報処理学会論文誌20－ 3
＜３＞入出力に図形表示を用いた要求仕様及び、…』　藤田　他、　第23回情報処理学会全国大会

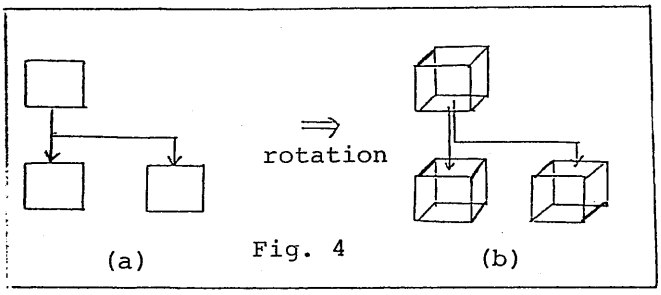

Fig. 4

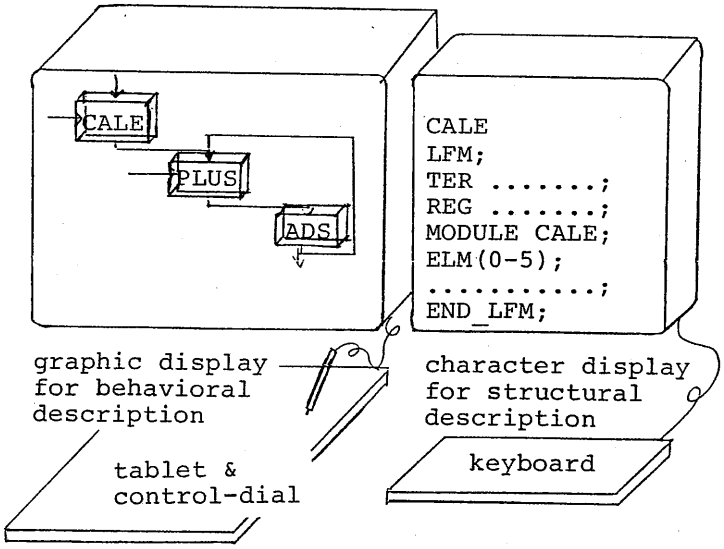

Fig. 5

```
behavioral description for CALE

LFM;
  TER XS(8),YS(8),XE(8),YE(8),
      V0(1),AX(8),AY(8),READY(1);
  REG DX(8),DY(8);
  FUN MINUS(F8 F8,F8);
 MODULE CALE:
     ELM(0-5);
       READY=0B;
       DX:=MINUS(XE,XS);
       DY:=MINUS(YE,YS);
     ELM(5-10);
       AX=DY;
       AY=DY;
       ACT PLUS;
       STOP;
END_LFM;

behavioral description for PLUS

LFM;
  TER AX(8),AY(8),OFX(1),OFY(1),
      SX(8),SY(8),BX(8),BY(8);
  REG DUMX(9),DUMY(9);
  FUN ADD(F8 F8,F9);
 MODULE PLUS;
     ELM(0-5);
       DUMX:=ADD(AX,BX);
       DUMY:=ADD(AY,BY);
     ELM(5-10);
       SX=DUMX(1-8);
       SY=DUMY(1-8);
       ACT ADS;
       OFX=DUMX(0);
       OFY=DUMY(0);
END_LFM;
```