6W-7

フローグラフジェネレータ（FLOLA）

山崎　雅生　　石黒　尚夫　　岩下　正雄
日本電気技術情報システム開発（株）　　日本電気（株）

1. はじめに

筆者等は先にデータフロー型画像処理プロセッサImPP（μPD7281）を開発したが，そのプログラム開発にはアセンブリ言語が用いられる．一方，データ依存並列性は，データフローグラフ表現の方が見やすい．

これらの両者は共に1対1に対応しており，相互に変換することが可能である．最初にプログラムを入力する場合，テキスト表現であるアセンブラ記述の方が速く，デバグ時にはフローグラフが有用である．

今回言語表現からフローグラフを自動生成するフローグラフジェネレータ（FLOLA: Flow Graph Generator & Layout System）を開発したので報告する．

2. マクロ表現の開発方針

マクロ表現の開発に当たっては以下のことを考慮した．

(1)できる限り簡略な記述にする．
(2)デフォルト値を許す．
(3)省略形を許す．
(4)キーワードは覚え易いものにする．
(5)ファンクション名は自動生成する．
(6)オプションパラメータの指定は，キーワードを用いる．
(7)下位のマクロはフローグラフ上で1つのファンクションに対応させる．複雑なマクロは階層マクロ記述とする．

3. マクロ表現

マクロ表現は図1に示すようにMCALLの後に汎用的な関数名を記述し，括弧の中に出力変数名，入力変数名，関数演算に必要なパラメータをキーワードを用いて記述する．これをマクロアセンブラにより図2に示すような出力変数の数，入力変数の数，パラメータの数等を先頭に付加した中間表現に変換する．

```
mcall add(dst30,dst31,dst21,rdcycs,1-h);
```

関数名　出力変数名　入力変数名　パラメータ

図1　マクロ表現の記述形式

4. フローグラフ生成

処理プログラムは，図3に示すようにマクロ表現で記述される．これはマクロアセンブラによりアセンブリ言語表現に変換され，アセンブラにより実行形式にアセンブルされた後，ローダによりImPP内のプログラムメモリによりロードされ実行される．

一方，フローグラフは同一のマクロ

```
2       :2      :1      :src10  :dst10  :lsa0   :lda0   :n000   :nop
:       :xy
2       :1      :2      :src20  :src30  :src10  :c001   :copym  :
:2      :1
3       :2      :3      :src40  :src41  :src42  :src21  :src20  :c002
:copybk :cntge  :1      :2*1    :m
```

図2　中間表現

Flow Graph Generator (FLOLA)

Masao Yamazaki,　Hisao Ishiguro and Masao Iwashita*
NEC Scientific Information System Development,ltd
*NEC Corporation

アセンブラ記述からフロー生成用のライブラリを参照して中間表現に変換され，フローグラフジェネレータにより2次元のフローグラフが出力される．フローグラフは，演算機能を矩形で，データの接続情報を折れ線で表す．

```
module ipp=8;
equate l=64;
equate m=256;
equate h=15;
equate host=0;
equate read=4;
equate write=5;
equate starts=0;
equate startd=32;
input lsa0,lda0,imor0 at 0,imor1 at 1;
output rdat0,rdat1,wdat,wadr,pend;
mcall nop(src10,dst10,lsa0,lda0,xy);
mcall copym(src20,src30,src10,2,1);
```

図3　プログラム記述（縮小処理の例）

5. 自動レイアウトと自動ルータ

中間表現を入力とし，まず入力変数と出力変数との接続を調べる．どの出力変数にも接続されていない入力変数は外部入力変数であると判断する．この際ループは仮にカットし，レベルをカウントする．

次にデータ依存性を調べ外部入力変数からパイプラインステージ毎に順位付けを行う．関数セルのレイアウトはここで求められた順位に従って上から下へ行う．レイアウト終了後，矩形間の接続線を自動的に発生するルーティング処理を行う．

自動ルータは予め等間隔のチャネルを用意し，この中に接続線を折れ曲がり点でライン単位に分割した始点終点ベクトル群を配置する．チャネルへの割当は，ベクトル間での衝突を判定しながら，空きスロットを見つけるスロットベクトル方式を用いる．チャネル幅はユーザが指定できるようにし，接続線数に応じて予め十分な大きさをもたせる．

接続処理終了後，矩形の縦横並びを維持したまま空きスペースをつめるパック処理を施す．生成されたフローグラフは図4に示すように一太郎等のワープロソフトを用いてプリンタに出力される．このときA4サイズに収容できる領域毎にカットし複数ページへの自動分割が行われる．

6. おわりに

ImPPのフローグラフジェネレータにつき述べた．これにより従来手書きで行っていたフローグラフ描画を，自動化することができ，大巾にソフトウェアプログラム開発効率が改善された．今後本手法をベースにしたフローグラフエディタ，ビジュアルシミュレータの開発を検討する予定である．

日頃貴重なご意見を頂く当研究部諸氏に深謝致します．

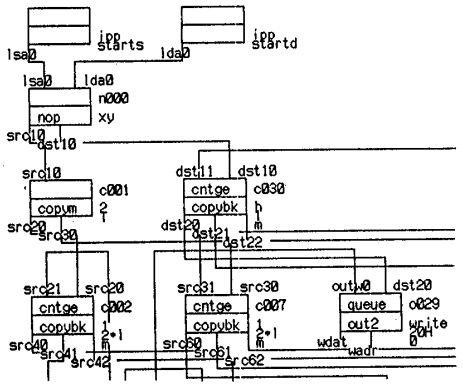

図4　フローグラフ出力結果

参考文献

(1)岩下，天満：A Data Flow Image Processor 2'nd ICS87 pp.408-409 1987

(2)太田，佐治：ImPP用高水準ﾃﾞｰﾀﾌﾛｰ言語 Streamの設計，情処学会第35回（昭和62年後期）全大 p.2435 1987