

## 文 献 紹 介

### 70-45 ILLIAC IV の Processing Element

Robert L. Davis: The ILLIAC IV Processing Element [IEEE Transactions on Computers, Vol. C-18, No. 9, Sept., 1969]

ILLIAC IV は 256 個の Processing Unit (PU) から構成される並列処理計算機である。PU はさらに Processing Element (PE) と 2048 語、1 語 64 bit、アクセスタイム 250 ns の Processing Element Memory (PEM) から構成されている。

すべての PU は、物理的にも、論理的にもみな同じ構成で、命令解釈、命令取り出しの look-ahead 等すべての PE に共通な制御部は有しないが、この点を除いて、大形は汎用計算機に匹敵する規模である。すなわち、PE には、5 個の 64-bit data register, memory address register, index register, 64 bit の carry propagation adder, 4 個の carry save adder, 64 bit の左右シフトが、1 クロックで任意にできる shifter などがある。

256 個の PU は、64 個ごとのグループにわけられ、各グループに 1 個の Control Unit (CU) と呼ばれる装置がある。CU は命令取り出し、解読などを行ない、そのグループ内の各 PE にマイクロオーダで共通の制御を送り PE を制御する。すべての PE は CU から送られるクロックにより完全に同期して動作するようになっている (これを lock-step synchronous operation という)。

このような構成である ILLIAC IV での問題は PE の演算方式と物理的な implementation にある。

PE の演算速度の設計目標は、浮動小数点加減算、浮動小数点乗算が、各々 250 ns (5×50-ns クロックタイム)、450 ns (9 クロックタイム) であり、高速演算方式が要求されたが、乗算や除算を高速化する有力な一つの方式である可変長桁移動方式 (Variable-shift) は採用できなかった。可変長桁移動方式は、演算データにより命令実行時間変るが、PE が多数あると lock-step synchronous operation を実現するためには、一番遅い PE に合わせなければならず、高速化は望めない。

乗算は、multiplier を 2 bit 同時に decoding し、4 個の Carry Save Adder を用いて、結局、8 bit 分

同時に乗算する一定長桁移動 (Uniform shift) 方式が用いられている。

除算は、1 ビットずつ商を求める non-performing 方式である。non-performing 方式は、restoring 方式と呼ばれる方式と同じと考えてもよいが、真の restoring 方式では、引算の結果が負になるとその後で restoring して補正するのであるが、non-performing では結果を accumulator に入れないように制御するので、この点 restoring と異なる。真の restoring 方式では、やはり多数の PE を lock-step synchronous operation で動作させるとすると、結局、毎回 restoring の step が必要となるからである。

ILLIAC IV には、256 個の大形計算に匹敵する PE があるため、Cost 上、logic の最小化は従来見られない新たな、非常に重要な問題である。しかし、logic の最小化といっても従来と規準が異なり、単に論理素子の数を減らすというのではなく、回路の共通化、すなわち種類 (Type) の最小化を計ることである。これには MSI を考えた場合、三つのレベルがあり、package の種類、diffusion set, metal mask 各々において種類を減らすことが重要である。

現在は、16-pin dual-in-line package (IC) を用い、これを 3.7×4.4 インチ大のカード上に搭載しているが、将来 MSI 化を計るために MSI package 単位の大きさに回路を分割し (これを board と呼ぶ)、board type の共通化を計っている。

1 個の PE には 154 boards あるが、制御部の 31 boards を除いて、board の種類は 18 種である。制御部の board はすべて種類は異なり、また gate 数の搭載密度も低い。ゲート数は全体の 7~8% であるが、board の占める面積は約 30% もある。近い将来、価格的にひき合った制御部の MSI 化は望めないであろう。  
(乾 範男)

### 70-46 プロセス制御用言語の試み

Peter I. P. Boulton and Pierce A. Reid: A Process-Control Language [IEEE Trans. on Computers, November 1969, Vol. C-18, No. 2, pp. 1049~1053] key: language syntax, PL/1 data acquisition language, process-control language, real time, software  
この論文では PL/1 を拡張して、アナログ、ディジ

タル実時間信号を PL/1 変数とまったく同様に扱えるようにし、汎用のプロセス制御用言語として使用することを提案している。

現存するプロセス制御用言語の大多数は FORTRAN の変形であるが、統語構造と表現力を十分に検討した結果 PL/1 を基盤とすることに決定したと述べられている。

実時間用言語として要求されることは、つぎの三つにまとめることができる。

- (1) 実時間アナログ／デジタル信号を取扱う方法
- (2) 割込みを取扱う方法
- (3) タスキングの方法

上記3項目を実現するために PL/1 をいかに拡張するかを、この論文の大部分をさいて詳述しているが、その概要は以下のようなものである。

まず、(1) についてであるが、外部アナログ、デジタル信号は PL/1 variable identifier で表わす。外部信号を表わす variable identifier は PL/1 expression 中で使用でき、dimension の定義も行なうことができる。normal variable と異なる外部信号を表わす identifier は ANALOG という attribute を用いることにより指定する。信号の読み、書きの仕方は ACCESS attribute により指定するようにしてあり、基本的方法が四つ与えられている。

次に、(2) についてであるが、PL/1 は CONDITION という内部割込み構造表示法をもっているが、新しいデータ・タイプ INTERRUPT を導入することにより、割込み機能を使いやすくするようにしている。

最後に、(3) のタスキングであるが、一般にプロセス制御形の問題においては、外部信号のアクセスはそのプログラムによって定義されている他のタスクと並行動作しているもう一つのタスクであると考えられる。多重プログラミング・バージョンの PL/1 には、そのような並列タスクを制御するための基本ステートメントが与えられている。これらの機能に必要な拡張を施して、タスク情報、その制御を行なうようにしている。VERSION オプションはその一つである。さらに、PRIORITY, STATUS, COMPLETION をそれら本来の機能として使用するだけでなく、疑似変数としても用いることができるようにし、タスキングを容易ならしめるようふうされている。

この言語をインプリメントする際の問題点もいくつか指摘されている。

(宇都宮公訓)

### 70-47 ISDS-計算機命令の集合を設計するプログラム

F. M. Haney: ISDS-A Program That Designs Computer Instruction Set [FJCC 1969 pp. 575~580]  
key: addressing scheme, automated design of computers, computer-assisted design, computer instruction set, generalized instruction set, heuristic design problem, idstruction set design system

ISDS (Instruction Set Design System) とは、計算機命令の集合を設計するプログラムであり、計算機による論理装置設計自動化の一環をなすソフトウェアである。この研究の主たる目的は人手を介さずに、設計問題を解決するプログラムを書くテクニックの開発にあるが、この論文では、そのうちの一課題である、計算機の命令コードの選択について述べている。

ISDS は GIS (Generalized Instruction Set) と呼ばれる計算機の命令語の一般化されたモデルをもっており、このモデルを用いて命令形式をつくり、それを要求された条件を満たすように解析して、最適な命令語の集合を選び出すようになっている。

GIS を Backus Normal Form で書けば、

```
<simple instruction> ::= <operation>
<left operand part> <right operand part>
<result part> <condition part> <if part>
<else part>
```

となる。ここに、<operation>とは以下のようなものを含めた 36 個の operations の list である。add, subtract, multiply, divide, compare, branch, shift, move logical instructions, その他。

また <left operand part>, <right operand part>, <result part>, <if part>, <else part> は <address> である。

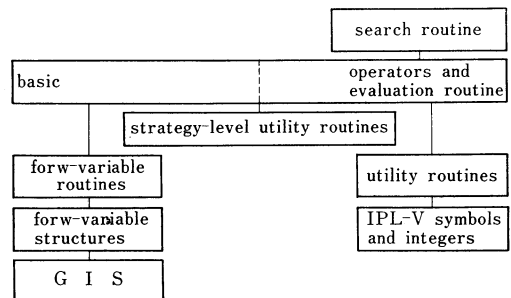


図 10 ISDS におけるルーティンとデータのハイアラキー

ISDS (IPL-V で書かれている) におけるルーティンやデータのハイアラーキーを図 10 に示す。

form-variable routines では form-variable に items を加えたり, items を削除したり, 探索したり, その他 attribute values に関する取扱いが行なわれる。

strategy-level utility routines は命令集合の構成をその主たる役目とする。

発見的設計プログラムは basic strategy routine と search routine とからできている。basic strategy routine は与えられたメモリー・サイズやワード・サイズを用いて各命令で指定すべきアドレスの数や各アドレスの一般形式を決定する。この結果を出発点として, search routine は命令の残りの部分の一つずつ埋めて行く。これは命令形式の余白がなくなるか, 制限コストに到達するまで繰り返される。

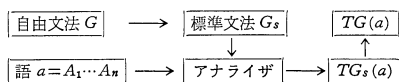
65536 語, 1 語 36 bit の計算機の命令形式を ISDS を用いてつくり出した例が最後に掲げられている。

(宇都宮公訓)

#### 70-48 無曖昧な自由文法のアナライザ

T. Kasami & K. Torii: A Syntax-Analysis Procedure for Unambiguous Context-Free Grammars [Journal of ACM, July, 1969, Vol. 16, No. 3, pp. 423~431] Key: syntax-analysis, parsing, context-free grammars, context-free languages, unambiguous grammars, recognizer, computing-time, normal grammars, phrase-marker, random access memory

本文献は, 無曖昧な自由文法に関するシンタックス・アナリシスの一方法を示すものである。中心となるアナライザは無曖昧な標準既約文法  $G_s = (V_N, V_T, P, S)$  に関するもので (標準文法;  $P = \{X \rightarrow w \mid w \in V_T \cup V_N V_N\}$ ), 入力語 (プログラム)  $a = A_1 A_2 \dots A_n \in V_T^+$  に対し, これが  $a \in L(G_s)$  であることを判断した後, その生成木表現  $TG_s(a)$  を生成し出力するものである。したがって, 全体的な手順としては, 与えられた無曖昧な自由文法  $G$  を  $G_s$  に変換する部分 (Standardization) および  $TG_s(a)$  を  $G$  の生成木表現  $TG(a)$  に変換する部分 (Structurally Discription Transformation) が, このアナライザの前後に付加されている。



アナライザの理論は, 語  $a \in L(G_s)$  の任意の部分

を生成する可能性をもつ変数を順次調査し結びつけるといった方法に基づく。  $a = A_1 \dots A_n$  の部分  $A_h \dots A_j$ ,  $1 \leq h \leq j \leq n$  が  $X \rightarrow A_h \dots A_j$  であれば, これを  $a$  の  $X$ -phrase といい,  $H(x, j) = \{h \mid X \rightarrow A_h \dots A_j, 1 \leq h \leq j\}$  で可能な範囲を把握する。ここで  $1 \in H(S, n)$  であれば,  $a \in L(G_s)$  といえる。また標準文法においては,  $X \rightarrow A_h \dots A_j$  であれば,  $Y \rightarrow A_h \dots A_k$ ,  $Z \rightarrow A_{k+1} \dots A_j$  かつ  $X \rightarrow YZ \in P$  (または  $X \rightarrow A_j$ ) であるから,  $X$ -phrase に関する可能な部分句  $T(X, j) = \{(Y, Z, h, k) \mid h \in H(Y, k)\}$  が定まる。ただし条件にあう  $X$ -phrase,  $X \rightarrow YZ \in P$  などが無い場合は, 各々  $\phi$  とする。この  $(Y, Z, h, k)$  が可能な parsing の方向をどう与えるものであるから,  $T(X, j)$  を  $1 \leq j \leq n$  に関して順次帰納的に生成し, 可能な  $T(X, j)$  を結びつけていけば  $a \in L(G_s)$  の生成木を得るわけである。したがって  $a$  の各  $X$ -phrase ごとに phrase-marker  $[x, ]x$  を付加して出力し,  $G_s$  による生成木表現  $TG(a)$  が求められる。

なお, 無曖昧な標準既約文法のアナライザに関する補題として,  $n^2 \log n$ -認識可能であることが証明されている。したがって, この意味においても, 本アナライザは, かなり斬新なものと考え得る。

(山下 元)

#### 70-49 Look-Aside メモリの研究

Francis F. Lee: Study of "Look-Aside" Memory [IEEE Trans. on Computers, VOL. C-18, No.11, NOVEMBER 1969, pp. 1062~1064] Key: associative memory, high-speed memory, memory organization, simulation, system design.

look-aside とは, 中央処理装置と主記憶装置の間に小容量, 高速の連想記憶装置を置き処理時間の短縮を計る方式で look-ahead にたいして名付けられた。

連想記憶装置は 4~256 個のセルを含み, 各セルはデータ部分, アドレス部分およびコントロール部分をもつ。アドレス部分が示す主記憶装置の内容がデータ部分に格納され, コントロール部分は purging algorithm に必要な情報をもつ。

この方式において, システム・パフォーマンスに影響するものは, 直感的に次の四つが考えられる。

- (1) 連想記憶装置と主記憶装置の速度比
- (2) 連想記憶装置の容量
- (3) 連想記憶装置より不要な情報を除く purging algorithm

(4) このシステムで実行されるオブジェクト・プログラム

purging algorithm は paging algorithm と異なり不必要となった語単位に移動される。二つの簡単なアルゴリズム 1) The single-activity-bit algorithm, 2) Round robin algorithm が述べてあり、後者のほうがより簡単である。

以上の観点より、シミュレーションが3種のプログラムにたいし行なわれたが、速度の点において、いずれのプログラムにおいても二つのアルゴリズム間に大差は見られず、また、1  $\mu$ s の主記憶装置に 128 個のセルで 100 ns の連想記憶装置をもつとすると実効サイクルタイムは 350~400 ns となる結果が得られた。

(吉川忠克)

### 70-50 スイッチング関数の Disjunctive Decomposition を求める一方法

V. Yun-Shen Shen & Archie C. Mckellar: An Algorithm for the Disjunctive Decomposition of Switching Function [IEEE Trans. on Computers, March 1970, Vol. C-19 No. 3 pp. 239~248] key: characterization constant, decomposition algorithm, decomposition graph, mod 2 canonical form, mod 2 map, simple disjunctive decompositions, switching functions

スイッチング関数のすべての simple disjunctive decompositions を得るアルゴリズムについて論じている。Ashenhurst による従来の方法、すなわち decomposition chart による decomposition test は可能なすべての bound sets について行なわれるため多大な計算時間を要していた。本論文のアルゴリズムではまず 2 変数の bound set をもつ decomposition test を行なう。これをもとにすれば decomposition 不可能な bound sets がいくつか直ちに判明するので、それらの bound sets を decomposition test の対象から除外する。以上の手法による計算時間の短縮が本論文の目的である。

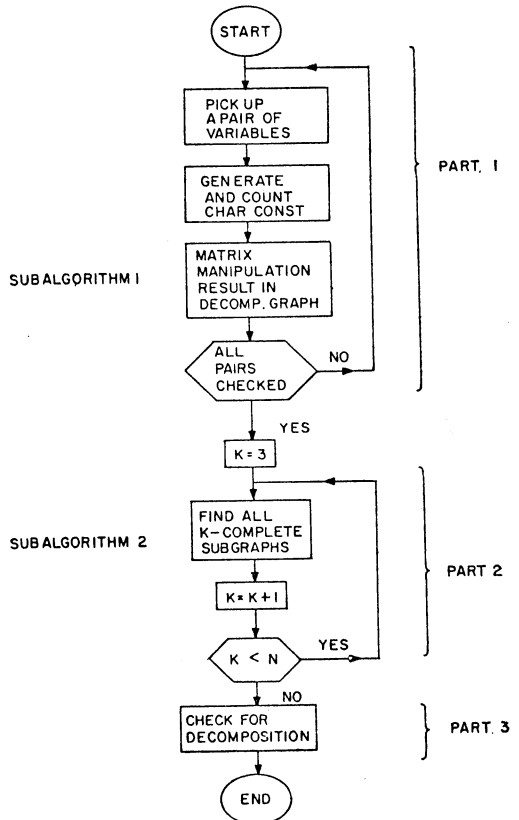
理論的にはこのアルゴリズムは以下の定理に基づいている。

定理 1 : partition  $[A, B]$  に対応する mod 2 map の第 1 列を除いて、零でない行がすべて同じであるとき、およびそのときにかぎりスイッチング関数は bound set  $A$ , free set  $B$  に関して decomposable である。

定理 2 :  $\{x_i, x_j\}$  をスイッチング関数  $F=f(x_1, x_2, \dots, x_n)$  の変数の 1 ペアであるとし、 $U_{ij}$  は  $\{x_i, x_j\}$  に関する characterization constants の集合であるとする。 $U_{ij}$  の要素を零でない行がすべて同じであるような  $2^{n-k} \times 2^{k-n}$  マトリックスに並べることが可能であるとき、およびそのときにかぎり関数  $f$  は  $k$  個の要素からなる bound set  $A$  に関して decomposable である。ただし、 $\{x_i, x_j\} \subset A$

また、そのフローチャートを第 1 図に示す。

(宇都宮公訓)



### 70-51 TSS におけるパフォーマンスのモニタリング

Tad B. Pinkerton: Performance Monitoring in a Time-Sharing System [Comm. ACM, Vol. 12, No. 11, 1969, pp. 608~610] Key: performance monitoring, performance measurement, multiprogramming performance, software measurement, system evaluation

本論文は、汎用 TSS のソフトウェア測定ファシリ

ティ (software measurement facility) に関するものである。

このファシリティは System/360 Model 67 を使用したシステム (Michigan Terminal System; MTS) において実現され、DCF (Data Collection Facility) と呼ばれている。DCF により、オペレーティングシステムやユーザプログラムの behavior およびパフォーマンスをモニタリングできる。

モニタリングすべき事象が発生すると、スーパーバイザにその旨が連絡され、専用のスーパーバイザがデータ

をバッファに貯える。データは two-word (8-byte) の prefix と、0~6 の additional words で構成され、事象の発生時刻、事象のタイプなどを記録する。バッファが一杯になると、buffer-managing job が起動され MT にダンプする。この MT はオンラインで処理される。1つのバッファには、平均 310 個のデータが収集できるが、バッファに格納できないデータ (たとえば、tape errors の発生) は、その個数をカウントしておく。

---

 ニ ュ ー ス
 

---

**FACOM 230-15**

富士通(株)は、このほど小型電子計算機 FACOM 230-15 を発表した。

FACOM 230-15 は、FACOM 230 シリーズのモデル 10 と 25 の中間に位置する高性能で、拡張性に富んだ小型汎用の電子計算組織である。

したがって、小型で DPS の一般の充実とともに情報処理ネットワークの完備の一環として有機的に利用できるよう、ハードウェア、ソフトウェア両面で各種の考慮がはらわれている。入出力装置についても 230 シリーズの他のモデルと同様に多種多様の装置が利用できるほかに、新たに高性能の内蔵磁気ドラム、小型 CRI ディスプレイ、大容量 2 連ディスクパック装置などが開発され、コストパフォーマンスが一段と向上するとともに、操作性が飛躍的に向上している。

このような点から広い分野での応用が可能であるがそのうち一部を示すと

## (1) 単独システムとして

- ・事務計算 (バッチ処理, インクワイアリなど)
- ・科学技術計算

## (2) 情報処理ネットワークの一環として (通信回線利用)

- ・リモートバッチ処理

## (3) 大型計算機の補助として

- ・PRE-PROCESSOR
- ・データ入出力, 変換用

などがある。

## 仕 様

方式: ストアード・プログラム方式, 10 進演算方式併用, 固定語長, 可変長フィールド方式併用  
主記憶装置

素子: 30 ミル磁気コア

速度: 1.5  $\mu$ s/16 ビット (サイクルタイム)

語長: 16 ビット+1 パリティ・ビット

計 17 ビット

容量: 4 K, 8 K, 12 K, 32 K 語

(8 K, 16 K, 24 K, 16 K バイト)

4 K 語 (8 K バイト) で増設可能

補助記憶装置

記憶媒体: 磁気ドラム

容量: 131 K, 262 K バイト

アクセス・タイム: 10/83 ms (50/60 Hz)

(5/4.15 ms に短縮可能)

情報転送速度: 225/270 K バイト / S (50/60 Hz)

演算制御

演算方式: (1) 並列 2 進法, 2 の補数表示, 固定小数点方式 (2) 直並列 10 進法, 符号+絶対値表示, 固定小数点方式

命令数

- (1) 基本モジュール I ……29 種
  - (2) 基本モジュール II ……15 種
  - (3) 科学用モジュール ……12 種 計 56 種
- (入出力命令は指令形式で拡張可能)

演算速度: 2 進 16 ビット加算 3  $\mu$ s

10 進 6 桁 加算 28.5  $\mu$ s

アドレス方式: (1) 1 アドレスまたは 2 アドレス方式 (2) 修飾: 5 種 (直接, 間接, 相対, 相対間接, インデックス)

インデックス・レジスタ: 4 個

割込み: 1 レベル

入出力制御

制御モード: 2 種

データチャネル: 5 種

情報転送速度: 最高 800 K バイト/秒

制御可能入出力装置: 255 台

**東芝が TOSBAC-5600 を発表**

東芝(株)は 1970 年代の情報処理にふさわしい大型電子計算機システムとして TOSBAC-5600 を開発し発表した。このシリーズはモデル 10, 30, 50 の 3 機種からなり、各モデルは上位方向に互換性を有し次の特長を有している。

(1) スリー・ディメンショナル・システムを実現  
完全なマルチ・プロセッシングおよびマルチ・プログラミング制御のもとにローカル・バッチ処理, リモート・バッチ処理, TSS が同時かつ能率的に一つのシステムで処理される。

(2) モジュール構造を採用

システム作成の簡素化, 保守の容易性, システムの

拡張に対する融通性などを考慮してハードウェア、ソフトウェアの両面において完全なモジュール構造を採用している。

### (3) 豊富なマス・ストレージ

ユーザの業務量と所要経費にみあった適切なシステムを構成できるように多種類のマス・ストレージが用意されている。

以上の機能を実現するためのハードウェアは、1語36ビットで32K語を単位として最大256K語まで拡張可能な $0.5\mu\text{s}/2$ 語の高速メモリー・モジュールを中心としたシステム構成である。入出力装置からの割込みはプロセッサの負担となることはなく、コア・メモリー・インタリーブ機能を有している。またソフトウェアは保護機能と使用制限機能をもった共有データベースのもとでのスリーディメンショナル・システムの実現、ユーザ・プログラム実行中にも診断プログラムを同時に通すことにより、リソースの使用効率を高めること、オペレータと計算機との対話を容易にすることを目的として設計されている。

これらの目的はハードウェア、ソフトウェアを制御するGECOS III (general comprehensine operating system) がマルチ・プログラミングおよびマルチ・プロセッシングのもとにハードウェアの有している機能を有効に駆使することにより実現されている。

### 光半導体素子 (レーナ)

最近、沖電気はS字型負性抵抗特性をもった発光素子を開発した。GaAlAsの液相成長においてドナー不純物にTe, アクセプター不純物にZnを同時にドーピング(ダブルドーピング法)、結晶成長の過程で徐冷速度を制御することによって、1回の成長過程でPNPNの4層構造を作ったものである。

この素子の長所は外部光、および電気のどちらでもスイッチングでき、かつ可視光(ピーク波長 $\sim 6,800\text{ \AA}$ )を発することにある。すなわち、受光、発光、記憶作用をかねそなえた素子である。ここでいう受光特性とは、電流-電圧特性におけるターンオーバー電圧 $V_t$ の減少、あるいは消滅を意味する。

おもな特性として、ターンオーバー電圧 $V_t$ 、同電流 $I_t$ 、保持電圧 $V_s$ 、同電流 $I_s$ があるが $V_t$ としては、 $2\sim 50\text{ V}$ 、 $I_t$ は $0.001\sim 1\text{ mA}$ 、 $V_s$ は $1.5\sim 2\text{ V}$ 、 $I_s$ は $0.1\sim 10\text{ mA}$ のものが得られている。

GaAsを母材としたレーナも同じく開発されている。この場合の発光は赤外光(ピーク波長 $\sim 9,000\text{ \AA}$ )

である。

応用例として、各種光論理回路、光パルス発生器、光メモリー、光遅延回路、リモートコントロール、無接点切替表示回路、波長変換などが考えられる。

### 高速漢字電子編集植字システム

日本電子産業(JEM)は第3世代植字機としてのJEM-3850高速漢字電子編集植字システムを発表した。これは日本科学技術情報センターなどで実用されている軽印刷用のJEM-3800高速漢字プリンタの印字、品質および機能を向上させた一般商用印刷に適する高品質の漢字電子植字システムである。

本システムは制御装置内部にミニコンピュータを内蔵し、植字装置の制御用として、また入力情報の編集コンピュータとしての両方に用いられ、植字装置としては多種文字の漢字植字には最適なフライング・スポット方式の文字発生およびプリントCRT上への表示によるプリント方式を採用したCRTビーム制御技術を中心とした完全な電子光学式植字装置である点が大きな特徴で、機能の概要は次のとおりである。

プリント文字種は標準で20,000字種を有するほか、1,250字ごとの書体交換、50字ごとのカセット文字交換による外字処理が可能である。

プリント文字の大きさは6~24ポイントで、任意の変形率の長平体が可能であり、縦横組の混植などがビーム走査制御により高速におこなわれる。プリント速度はラインフィード時間を含めて平均8ポイントで150字/秒の高速である。

入力はコンピュータ・オンラインあるいは磁気テープ、紙テープ装置により、出力は校正モニター用のソフトプリンタおよび版下作成用のフィルムレコーダがある。

JEM-3850は編集コンピュータとしても用いられ、EDIC-1なる編集植字ソフトウェアが用意されているほか、従来新聞社などで一般に使用されている漢字キーボードを用いて10,000字種の漢字入力を簡単におこなうことができる新しい漢字パターン入力方式による漢字辞書ソフトウェアも完成している。

同機の第1号機は今年6月学習研究社に納入され実用化の予定である。

### 「情報処理大学院」事業計画を決定

上級情報処理技術者の養成機関である(財)情報処理研修センター(通称「情報処理大学院」,理事長 山内

二郎(青山学院大教授))は、5月29日第1回理事会を開催し、昭和45年度事業計画、収支予算などを決定し、10月開講に向けて具体的準備を開始した。

同センターは3月30日付で通産大臣の認可をうけ、当初基金5億300万円を以て発足し、45年度中にさらに5億円を追加して10億300万円の基金となる予定。同センターは、東京・芝浜松町の世界貿易センタービル7階に設置される。

情報処理研修センターは、本格的な情報化社会の到来をむかえて深刻な問題となりつつある上級情報処理技術者、あるいは教育担当者の養成を主眼とし、①システム・エンジニアやシニア・プログラマなどの上級情報処理技術者の養成 ②企業内情報処理部門や利用部門の管理者の教育 ③情報処理教育担当者や企業内インストラクタの教育訓練を目的としている。

すでに10月1日の開所を目標に、協力団体である(財)日本情報処理開発センター内に情報処理研修センター業務準備室と基本問題委員会(委員長森口繁一・東大教授)を設け、教育カリキュラム、研修センターのレイアウト、教育用電算機の選定、教育方法の検討、担当講師への交渉等を行ってきた。

同事業計画では、初年度(半年間)は4コース(延べ6クラス・160名)を予定しており、

システム・エンジニア・コース(1クラス, 30名, 6カ月), シニア・プログラマ・コース(1クラス, 30名, 6カ月), インストラクタ・コース(延べ2クラス, 各30名, 各3カ月), 管理者コース(延べ2クラス, 各20名, 各2カ月)が企画されている。平年度には6コース(延べ11クラス, 380名)を計画している。

同センターの教育の特色は ①マス教育をさけ少数のグループ教育方式をとり、②演習、事例研究を主とした実務教育を重視し、③講師陣には内外各方面の学識経験または実務経験ゆたかな最高水準の人々を委嘱する。④将来は研修センター自身がシステム開発のプロジェクトを採り上げ、シンク・タンクとして技術の向上と発展に資することが考えられている。

なお、研修生の受講料については、補助を受けることにより、所要資金の数分の一とし、その負担を軽減するために生徒一人につき半年間に100万円以上の資金を投入する予定で、このため初年度においても1億2,340万円余の事業費を計上している。

### 国際会議案内

1970年7月27—28日

**ACM Symposium on Compiler Optimization**,  
University of Illinois, Urbana. III. Spon: ACM  
SIGPLAN. Contact: Robert S. Northcote, ILLIAC  
IV Project, 170 E. R. L., University of Illinois,  
Urbana, IL 61801. (♣切3月30日)

1970年8月24—28日

**IFIP World Conference on Computer Education**,  
Amsterdam. Sponsor: IFIP Technical Committee  
for Education and Administrative Data Processing  
Group. Chm: A. A. M. Veenhuis, Sec. Gen, IFIP  
Conf. Cptr. Educ. 1970, 6 Stadhouderskade,  
Amsterdam 13, Netherlands. US Reps: W. F.  
Atchison, Mrs. S. Charp, D. Teichrow.

1970年9月1—3日

**ACM NATIONAL CONFERENCE**, New York  
Hilton, New York City. Conf. Chm: Sam Matsa,  
IBM Corp., 410 East 62 St., New York, NY  
10021. Prog. Chm: Robert E. Bemer, General  
Electric Co., 13430 North Black Canyon Highway,  
Phoenix, AR 85029.

1970年9月7—11日

**Sixth International Congress on Cybernetics**,  
Namur, Belgium. Contact: Secretariat, Association  
Internationale de Cybernetique, Palais des Ex-  
positions. Place Andre Rijckmans, Namur,  
Belgium. (♣切: 2月1日)

1970年9月9—18日

**International Federation for Documentation  
(FID) Annual Conference and International  
Congress on Scientific Information**, Buenos  
Aires. Contact: USNCFID Secretariat, Nat'l  
Academy of Sciences, 2101 Constitution Ave.,  
N. W., Washington, DC 20418.

1970年9月14—18日

**7th International Symposium on Mathematical  
Programming**, The Hague, Holland. Sponsors:  
ACM SIGMAP, in conjunction with many other  
societies. Contact: G. Zoutendijk, c/o Computing  
Center. University of Leiden, Station-plein 20,



Leiden, The Netherlands.

1970年10月4—9日

**American Society for Information Science, 33rd Annual Meeting**, Bellevue Stratford Hotel, Philadelphia. Sponsor: ASIS. Contact. Kenneth H. Zubriskje, Jr., Biosciences Information Services of Biological Abstracts. 2100 Arch St., Philadelphia, PA 19103.

1970年10月5—7日

**Symposium on Feature Extraction and Selection in Pattern Recognition** Argonne National Laboratory, Chm : K.S. Fu, School of E.E., Purdue Univ., Lafayette, IN 47907 (※切: 7月1日)

1970年10月28—30日

**Eleventh Annual Symposium on Switching and Automata Theory**, Santa Monica, California. Sponsors: IEEE Computer Group and Department of Systems Science of the University of California, Los Angeles. Contacts: papers, Peter

Weiner, Department of Computer Science, Yale University, New Haven, CT 06520; arrangements, J. W. Carlyle, Department of System Science, University of California, Los Angeles, CA 90024. (※切: 5月15日)

1970年11月17—19日

**1970 Fall Joint Computer Conference**, Astrohall, Houston, Texas. Sponsor: AFIPS. Gen. Chm: Robert Sibley, Jr., Dept. of Computer Science, University of Houston. Cullen Blvd., Houston, TX 77004. Program chm: Larry Axsom, IBM Scientific Center, 6900 Fannin St., Houston, TX 77025. (※切: 4月10日)

1971年8月3—5日

**ACM National Conference**.....Vol. 11, No. 3 を参照のこと

1971年8月23—28日

**IFIP Congress 71**.....Vol. 11, No. 3 を参照のこと (※切: 11月30日)