講　　座

電子計算機とその応用（その1）

―― 電子計算機について ――

中　村　　充*

I. 緒　言

現在電子計算機の普及は目覚ましい。各大学でも次々と設置され，農業土木試験場でも農林省研究機関の共同利用ではあるが自由に使用できる。また今回農林省農地局設計課に FACOM—20 が設置され，行政部門への導入が行なわれたことは大きな前進である。このような状勢で電算機の利用は研究者のみならず，すべての技術者によって行なわれなければならないときが来ていると考えられる。これが学会で誌本講座をとり上げた最大の理由であろう。電子計算の優れた特色と各種の利用は，ここで改めて概説することもあるまい。本講座の中で数値解析，統計的手法，オペレーションズ・リサーチ (OR)，自動設計などについて具体的に解説する。

農地局設置の電子計算機は農地事業にたずさわる多くの人に開放されているのであるから身近かなものとして関心をもって欲しい。

農地局では電子計算機の運営協議会を作り，この中にはプログラム開発委員会が設けられた。本講座はこの委員会の委員で分担執筆する予定である。

II. 電子計算機について

1. アナグロ計算機とデジタル計算機

計算機には相似型計算機 (Analog computer) と計数型計算機 (Digital computer) がある。前者は計算尺のように被演算量を長さに相似させたり，浸透流電気モデルのように地下水流におけるダルシー法則をオームの法則に置きかえて地下水圧を電圧に，流れを電流に相似させたりするものである。

後者はソロバンや手廻し計算機に代表されるように，数字による演算を行なうもので本講座で扱うのは，デジタル型計算機に属する万能型電子計算機についてである。この発達の過程を略述するのも理解の一助となるであろう。

2. 単能の計算機

デジタル計算機の最古のものはソロバンであるが，機械といえる程度のものはパスカル (Pascal) に始まり，ライプニッツ (Leibniz) によって改良され，本質的には手廻し計算機や電動計算機と同じ機能のものが作られ以後製作技術も進んで現在の卓上計算機となった。これは演算のワンステップを行なわしめるもので被演算量をそのつど人間が与え，演算結果はまた，そのつどノートに書きとめ，被演算量として再び与えたりするもので一連の演算のワンステップを断続的に実行するものである。

3. 自動型計算機

これに対し最初に必要な数値を入れるだけで，その後は自動的に一連の演算を行なわせ，最後に必要な結果だけを印刷させる計算機がチャールズ・バッベージ (Charles babbage) によって考えられた。これは実現しなかったが，今日の電子計算機につながる最初の考案である。これは後にハーバード大学の Mark I (1944 年)，Mark II によって実現された。これは途中演算結果を計数輪に記憶させたり，継電器（スイッチ）を用いるなど，主として機械的な構成をもつ電気機械式計算機である。

4. 結線プログラム型電子計算機

IBM 社とペンシルバニア大学の協同で ENIAC (1945 年) が作られ，これは前述の Mark I，II の計数輪やスイッチを全部真空管に置きかえたもので，掛け算 300 回/秒，加減算 5,000回/秒という驚異的速さであり，最初の電子計算機である。しかしこれは演算内容の指定（プログラム）は結線で行なわれ，新しいプログラムを作るための結線が大変であった。

5. 万能型電子計算機

ENIAC に改良を加えたものが，英国の EDSAC をはじめとする各種の万能型電子計算機で，ENIAC との本質的な相異はプログラムそのものも記憶装置の中に記憶させ，演算の制御がこの記憶によって行なわれる。いわゆる人工頭脳としての現代の電子計算機となったことである。その間，構造的には真空管がトランジスター，ダ

* 農林省農地局プログラム開発委員会

イオード回路に，更に集積回路へ，記憶方法も高速記憶にはブラウン管記憶や磁性体，誘電体記憶装置へ，低速記録は磁気ドラムや磁気テープなどへと高速化，大量記憶化へと改良が行なわれている。

6. ソフトウエアの開発

（1） **自動プログラミング**　EDSAC 以後の機能上の開発ではソフトウエアの工夫があげられる（計算機械そのものに関することをハードウエア，計算機を使うためのプログラミングに関連したことをソフトウエアと呼んでいる）。その第一は自動プログラミングの開発であろう。これは使用者が機械語で命令するのでは使いにくい点を考慮して，計算機自身の中に通訳ルーチン（インタープレイター），翻訳ルーチン（コンパイラー）（ルーチンは，明確な機能をもった命令群）などを設えて，使用者は，READ, WRITE, GO TO n, A+B, A/B など日常の記法でプログラムを作ればよいようになっている。こうして機械そのものを知らない人でも容易に使用できるようになった。

（2） **組み込み関数，ライブラリー関数の開発**　更に電子計算人口の増加にともなって使用度の高い関数，たとえば e^x, $\sin x$, $\log x$, $\sqrt{x}$, $|x|$, $x^{a/b}$ などは計算機の中に組みこまれていて，これを計算するためのプログラムは作る必要はなくなっている。これを組み込み関数と呼んでいる。また計算機の中に組み込まれていなくとも，それ自身明確な機能をもった一連の命令群（erf x, $\Gamma(x)$, etc）で，全体のプログラムの部分として用いられるものをサブルーチンと呼び，これは既成品（ライブラリー関数）として保存すれば必要に応じて多くの人の使用に提供することができる。

（3） **プログラム・ライブラリーの開発**　さらに一般の利用が普及してきたのは，プログラム・ライブラリーの開発である。これは前述のサブルーチンのように全体のプログラムの部分として用いるもの（プログラマー用）の他に，全体としてまとまった一連のプログラム，たとえば不定流の計算とかスベリ円の解析，ラーメンの計算なども一般的プログラムとして作っておけば，実際の具体的数値を与えれば，プログラミングを知らない人でも必要な計算結果を得ることができるわけで，われわれの分野でも，設計基準やハンドブックのようにプログラム・ライブラリーを完備することが必要であり，前述のプログラム開発委員会でも農業土木プログラム・ライブラリーを作る作業を進めつつある。

III. 電子計算機の構成の概要

Ⅱ．で説明したように自動プログラミングの完備した現在，電子計算機の機構は知らなくともその使用はできるのであるが，全く知らずにプログラムの方法を学ぶのではなじみが薄く取付きにくい点もあると思うし，外見を見ても何のことか全くわからないので，ここで簡単に計算機の仕組みについてのべておく。

1. 考えるということ

考えるということは，体験や教育によって得られた記憶と，与えられた条件と記憶から論理操作が行なわれ判断されるということである。

電子計算機では，記憶や条件の内容，すなわち論理変数には0，1の2記号のみで組立てられる。0，1で組立てられた論理変数 x_i について，一つの法則に従って論理操作を行ない，

$$y = f(x_1,\ x_2, \cdots\cdots x_n) \quad \cdots\cdots(1)$$

なる y を求める。y も又0，1で示される。y は数値計算の結果であったり，期待値であったり，判断であったりする。このことは人間の頭脳の考える作用と同質であり，電子計算機を人工頭脳とよぶゆえんである。

0，1の二つの信号は，現象が起こっているいないという明確な識別ができ，途中で適当に増幅すればながく情報を正確に維持できて便利であり，デジタル型電子計算機のあらゆる情報は0，1で組み立てられる。

2. 2進演算

0，1の記号で数を組み立てると次のようになる。

10進法	0	1	2	3	4	5	6	7	8
2進法	0	1	10	11	100	101	110	111	1,000

例

$$\begin{array}{rcl} 5 & \rightarrow & 101 \\ +\ 3 & \rightarrow & +\ 11 \\ \hline 8 & \rightarrow & 1,000 \end{array}$$

一般的に x_i を0，1の2進法の記号とし（$x_n\ x_{n-1}\cdots\cdots x_1 x_0$）となる2進数と10進法の数 X との対応は，

$$X = 2^n x_n + 2^{n-1}x_{n-1} + \cdots\cdots 2x_1 + x_0 \quad \cdots\cdots(2)$$

とあらわせる。他方，

$$Y = 2^n y_n + 2^{n-1}y_{n-1} + \cdots\cdots 2y_1 + y_0$$

として，これと X との和を2進法で行なうには，

$$X + Y = Z$$

$$Z = 2^n z_n + 2^{n-1}z_{n-1} + \cdots\cdots 2z_1 + z_0$$

における末尾から $i+1$ 番目のケタ $2^i x_i$, $2^i y_i$, $2^i z_i$ の間に，C_{i-1} を i 番目のケタからの繰上りとして，

$$\left.\begin{array}{l} x_i + y_i + C_{i-1} = z_i + (\text{繰上がり}\ C_i) \\ \text{しかし}\quad x_i + y_i + C_{i-1} > 1 \text{ のとき } C_i = 1 \\ \qquad\quad 〃 \quad \leqq \quad 〃 \quad\ C_i = 0 \end{array}\right\} \cdots\cdots(3)$$

である。この組合わせが2進加算の基本形で，これを表にしておく（これを2進加算真理値表という）。

表-1　2進加算真理値表

x_i	y_i	C_{i-1}	z_i	C_i
0	0	0	0	0
0	1	0	1	0
1	0	0	1	0
1	1	0	0	1
0	0	1	1	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	1

3. 基本論理操作

電子計算機の論理操作はすべて **表-2** の素子の操作に分解される。

A, B の論理積 (∩) and とは A, B が共に起こる部分を示し，A∩B で示す (共通部分)。現象が起こっている状態を 1，起こっていない状態を 0 として，A が 0, 1, B が 0, 1 をとるとすれば，A∩B は A=1, B=1 である。

A, B の論理和 (∪) or とは A か B の少なくともいずれかがおこる部分を示し，A∪B で示す (和集合)。A が 0, 1, B が 0, 1 である時，A∪B は (A, B) が (0, 1) (1, 0) (1, 1) となることである。A でない (not) ということを A^c と書き A の補集合という。A=1 の時 A^c=0, A=0 の時 A^c=1 である。これらはすべて電気回路として与えられ，これらを **表-2** に示す。実際にはトランジスター，ダイオード回路，集積回路として，**表-2** の素子が約 10cm 角の一枚の板 (集積回路では約 2cm 角の板) に配線されている。このような素子の組合せで，記憶を除く電子計算機のすべての機能が作られる。これを以下順を追ってのべる。

表-2　基本論理操作

基本操作	論理類 and (論理記号 ∩)	論理和 or (論理記号 ∪)	否定 not (論理記号 c)
真理値表	A \ B: 0 → 0 0 ; 1 → 0 1	A \ B: 0 → 0 1 ; 1 → 1 1	A′: 0 → 1 ; A^c: 1 → 0
回路模型	スイッチ回路 A B A∩B ダイオード回路 +E A B A∩B A.B共に電位が上がらなければ. A∩B は電位は上がらない。	スイッチ回路 A B A∪B ダイオード回路 A B AB共に低電位のときのみA∪Bは下がる。	スイッチ回路 A A^c
記号	A B A∩B	A B A∪B	A A^c

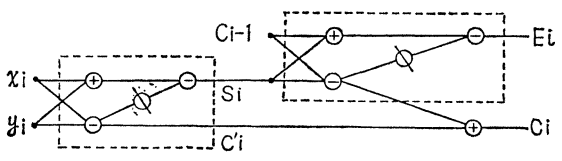


図-1　2 進 加 算 回 路

4. 2進加算回路

図-1 に示すように and, or, not 回路を組合わせると，2進法演算における加算が行なわれる。図中の点線ワクは半加算器といい，入力 x_i と y_i を加算して S_i・繰上り C'_i が得られ，これと下位のケタからの繰上り C_{i-1} と S_i を加算して z_i を，その繰上りと C'_i のどれから繰上っても，これは上位への繰上り C_i となる。

これは(3)式，および **表-1** の各対応関係を回路として与えたもので，これを素子として加算のすべてが行なわれる。たとえば **表-1** の 6 行目，x_i=0, y_i=1, C_{i-1}=1 の場合を **図-2** に示す。

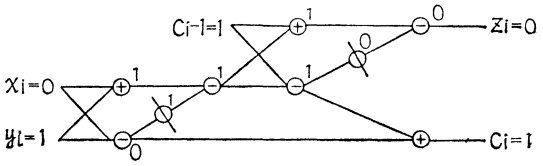


図-2　2 進 加 算 例

図-2 は2進数の i 番目のケタの計算を示し，i 番目の数字が0と1で $i-1$ 番目のケタからの繰上り1がある場合，答えが0で繰上りが1でわることを示す。この回路は **表-1** のすべての組合わせを満足するものである。

5. 補 数 器

電子計算機で減算をする時は，引く数を補数に直して加算にする。これは対数計算では仮数部を常に正の和で示すのと同様で，

$$\log(2/3)=\log 2-\log 3=0.301-0.477$$
$$=0.301+\bar{1}.523$$
$$=\bar{1}.824$$

のように，-0.477 を $\bar{1}.523$ として加えることと類似している。2進数で補数を求めるには次のようにする。

例　8−5=8+(−5) として計算する。

$-5\rightarrow -101\rightarrow(-1000+0011)\rightarrow\bar{1}011$

$(8-5)\rightarrow(1000+\bar{1}011)=11\rightarrow 3$

X の補数を X' として

$$-X=X'=-2^n x_n-2^{n-1}x_{n-1}\cdots\cdots-2x_1-x_0$$
$$=-2^{n+1}+2^n(1-x_n)+2^{n-1}(1-x_{n-1})$$
$$+2(1-x_1)+(1-x_0)+1 \quad\cdots\cdots(4)$$

この補数 X' を用いて

$$Y-X=Y+X'$$

として減算は加算になる。

(4)式中の ( ) の中について

$1-x_i$ は x_i=1 なら 0
　　x_i=0 なら 1

すなわち $1-x_i$ は x_i を not の回路で否定すればよい。よって補数を求めるには(4)式の内容から各ケタの数を否定（0→1, 1→0とする）して1を加えればよい。

例　1011011 の補数 $\bar{1}0100100+1=\bar{1}0100101$

よってこれは，**図-3** の回路で示される。

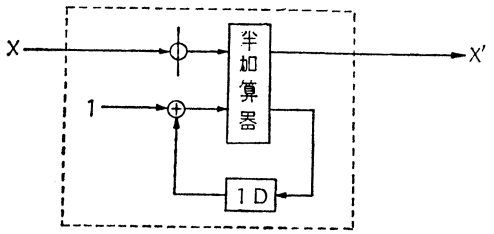


図-3 補数器

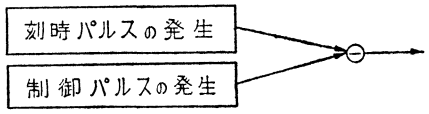
ここに 1D は1パルスだけ情報の伝達をおくらせる記号。

図-3 は2進数の末位の数を否定（1を0，0を1とする。）して1を加え，その繰上りを次の数の否定値に加え，順次補数ができ上る。X' を加算機の x_i に入れれば，他の和 y_i からの減算ができる。

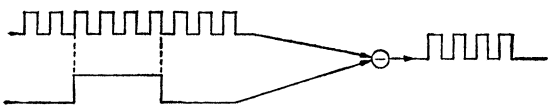
6. 演算の進行

ここで演算がどのように進行するかをのべておく。

演算は基準パルスの発生に合せて（同期方式）進行する。従って電子計算機の中の必要な箇所に必要なパルスをとり出すには，後述のデコーダー，コーダーによってゲートの操作をすることによってもできるが，次の回路でもできる。



たとえば4パルスをとり出す時は，



情報は and, or, not 回路（5項参照）から判るように逆方向には移動しない。

and 回路は A, B なる入力のうち A を指令，B を情報の通路とすれば，A が 1（高電位）であれば B に来る信号パルスは通過するが A が 0 であれば B に来るパルスはしゃ断される。すなわちゲートとしての機能となる。このように同期方式では演算の進行は刻時パルスによって進行するから，演算速度は刻時パルス発生の速度によって定まる。これは今後ますます高速化し，現在の集積回路ではナノ・セコンド（$^{-9}10$ sec）の速さである。

7. ケタ送り (Shift register)

ここでフリップ・フロップ回路について述べる。**図-4**

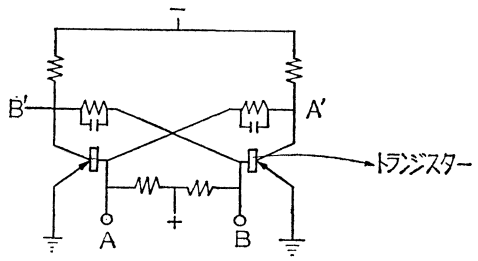


図-4 フリップ・フロップ回路

のような回路で A, B は入力 A', B' は出力でこの回路の特徴はもし A に入力1があれば出力 A' は1となり B' は0となって B に入力がない限り，常にこの状態を保持する。このようにフリップ・フロップは情報も記憶することができる。B に入力があれば直ちに A' は0となり B' が1となる。

次にフリップ・フロップ (F, F) によるケタ送りを示す回路は **図-5** のようである。P なる F, F 2 0, 1 が

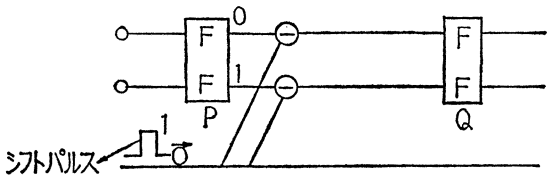


図-5 フリップ・フロップの情報の移動

記憶されているときパルスが来れば0につながる⊖はパルスを通さない。1につながる⊖はパルスを通して，Q なる F, F には P と全く同じ 0, 1 の状態が作られる。すなわちケタが送られたことになる。

図-6 について説明すると，A, B はそれぞれ10個の F, F で構成された情報記録部で，0 または1を記録できる一マスを1bit といい，普通20～60bits で一つの記憶部または演算部を構成する。（FACOM 270～20 では16 bits で1語を構成し1文字は8 bits，整数は1語，小数点付数（実数という）は2語で示されている。倍精度の計算する時は，それぞれ2語および4語，すなわち 32 bits，および 62bits を必要とする）。さて **図-6** で B にある記憶を A にとり出すためにシフト・パルスを一つ送れば **図-5** の原理に従って **図-6** 下の図のように，B の末尾の情報1は，A, B の先頭に送られ，B では一つだけ循環する。従って10個のパルスで B にはもとの情報を記憶したまま同じ情報が A に送り出されたことになる。もし A にはじめに別な情報が入っていた時は，

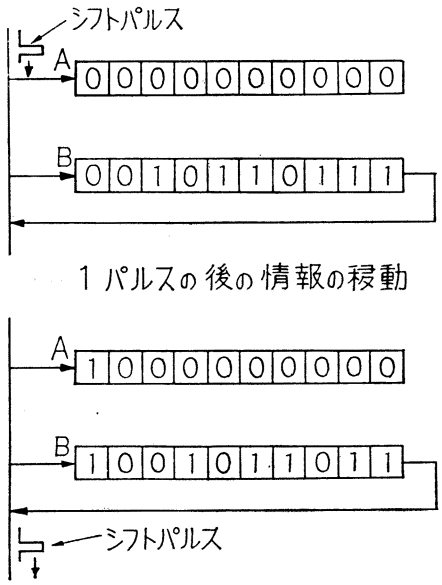


図-6 ケタ送り

この情報は消されて B の情報が入ることになる。これでケタ送りおよび情報の転送が理解できる。ケタ送りと加減算を組合わせれば掛け算，割り算が出来ることになる。

8. 比較回路（符号）

掛け算，割り算における符号の演算などは **図-7** の比較回路で行なわれる。この回路は二つの入力信号が等しければ1，等しくなれば0となる。図中に $x_1=1$, $x_2=0$ のとき（異符号のとき）答えは $y=0$（負）となることを示す。

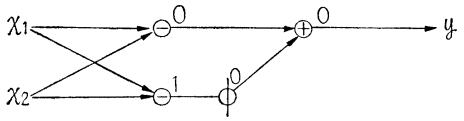


図-7 比較回路

9. 演算装置

以上で4則演算や情報の移動についてのべたが，ここでは演算装置で最も重要な2進累算器（Binary accumulator）についてのべる。**図-8** は2進累算器で加算器は **図-1**，2 に示したものである。F，F とあるのは1パルスだけ情報を送らせかつ記憶させるもので，**図-4** のフリップ・フロップ回路である。**図-8** の加算器は上段から順次2進数の末位からのケタの数を担当する。例で説明すると，A_1，A_2，A_3，A_4 なる4個の2進累算器（B，A と略記する）に最初は情報が0，すなわちどの F，F も0であるとする。ここに $X=101$ が送られてく

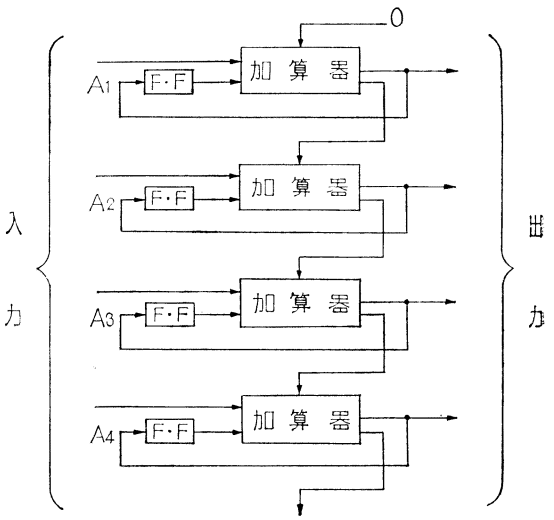


図-8 2進累算器（並列型）

ると A_1，A_2，A_3 の F，F は，最初の情報0と加算した結果を末位から 1，0，1 と記憶する。次に $Y=11$ が送られると A_1 の（B，A）は F，F=1 と Y の末位の1が加算され，出力は0となって F，F は新たに0を記憶し，繰上りの1は A_2 の加算器に入る。A_2 の（B，A）では F，F=0 に Y の2ケタ目の1と A_1 からの繰上り1で，出力は0，繰上り1となる。A_2 の F，F は0を記憶する。A_3 の（B，A）では F，F=1 と繰上り1が加算されて，出力0，繰上り1となる。よって A_3 の F，F は0を記憶し，A_4 の（B，A）では入力 A_3 からの繰上りだけであるから A_4 の F，F は1を記憶する。この結果 $X+Y=101+11=1,000$ を計算し記憶したことになる。

10. デコーダー（Decoder）；コーダー（Coder）

さて，記憶部にある情報を演算部に呼び出したり，演算部の中でも前項の例に見るようにケタごとに指定した2進累算器に入れ，減算のときは補数器を通してから累算器に入れ，割り算のときはケタ送りをしながら補数器を通して2進累算器に入れる必要がある。また演算部で計算された結果を記憶部にしまったり，印刷させたり，情報を制御する必要がある。さらに制御装置の中にあって命令を解読（Decode）し，それを実行するための制御（基本回路のゲート操作）の信号（Code）をただすための回路が必要である。**図-9** はデコーダーで m 個の入力変数から加法標準型に表わされる $2m$ 個の項を回路 and で作る。図では x，y，z の3入力変数に対するもので $2^3=8$ 個の組合わせができる。これらの組合わせを記憶部の番地（1語あて）に用いたり，2進，10進

の変換を行なったり，その他制御信号をあてておくわけである。

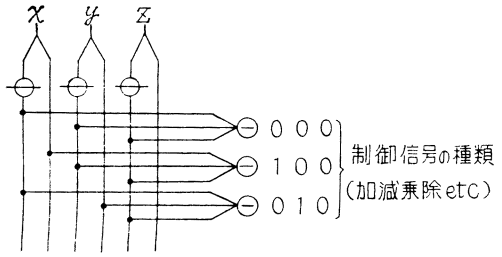


図-9　デコーダー

コーダーは，デコーダーで解読した指令に基づいて，これを実行するためのゲート操作を行なうためのパルスを発信する。これは m 個の入力のうち必要なものを or で集める回路で 図-10 に示す装置になっている。

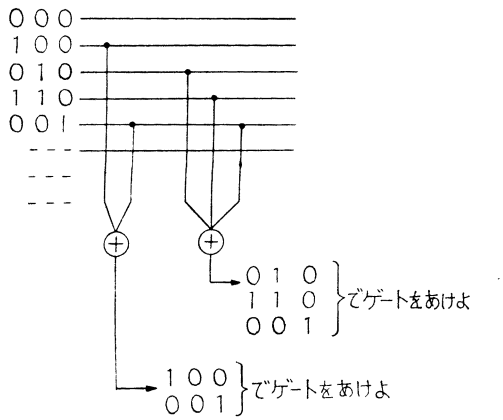


図-10　デコーダー

11. 記　憶

大きな数値計算や，大きなプログラムを組むときは読み込ませる情報はもちろん，途中演算結果を多数記憶させる必要がある。従って計算機の能力は記憶容量によって決まってしまうといってよくも，記憶には電気的に刻時パルスの進行に従って高速で読み込み読み出しのできる主記憶装置と，磁気ドラムや磁気デスクのように機械的回転を伴なう低速記憶装置，磁気テープのように取りはずして保管できる外部記憶装置がある。低速記憶や，外部記憶は一括して読み込み，読み出しして主記憶装置に出入し，演算の各ステップでそのつど呼び出したり，読みこんでいたのでは電子計算機の高速度性は失なわれるともいってよい。従って電子計算機の性能は主記憶装置 (Core memory) の語数で定まるといわれている。この語数は中型機で 10～30K. words，大型機で 50～100K. words である。前述のフリップ・フロップ回路も記憶装置の一種であるが，単なる記憶にはより簡単な磁心による記憶などが用いられる。これは20～60の網目（1辺の長さ約10～15cm 程度）状に磁心を配置し，これに縦横の読み込み電流を与えるようにし，この交点を磁化によって 0，1 を記憶させる。

12. 計算機の機構

以上で電子計算機の計算方法の大略は理解されたと思う。ここで全体の構成の一例を示すと，図-11 のとおりである。前述のように計算機の中にはきわめて多くの 0，1 が入っていて刻時パルスの発信によって解読，演算，転送，記憶などが行なわれているわけである。

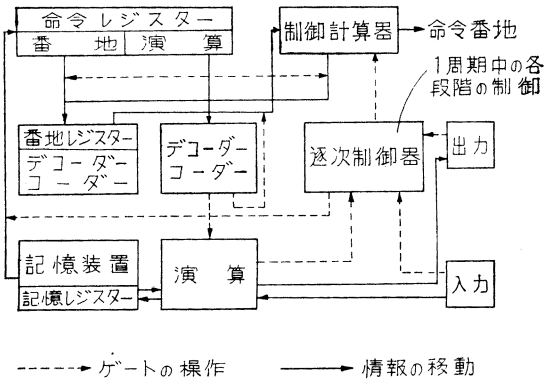


図-11　電子計算機の構成

13. 入出力

入力はカード，または紙テープにあけられた孔に光を通しこれを光電管で受けると孔を通して光を受けたものは電流が通じて1，他は0と読みこまれるわけである。これにより，計算機が作知する。

計算機に印刷させるには，ライン・プリンターを用いる。

IV. 電子計算機言語

Ⅱ.6 で述べたように初期の段階では，電子計算機を使うためには計算機自体の言語（機械語）を理解しなければならず，この機械語は計算機ごとに異っており，いろいろの機械語でプログラムを作ることは大変な仕事であった。そこでプログラムを作るプログラムが工夫され，プログラミングの仕事のうち比較的機械的な部分を機械自身にやらせるような自動プログラムが開発された。この最も基礎的なものがアッセンブラーであり，さらにもう一段水準の高いコンパイラーが開発された。これはわれわれが，普通に使っている数式や日常語に近い形でプログラムを書いて入れると，機械がそれを自分自身の言語に直すものである。この種のプログラム用語で

現在もっとも普及しているのが FŌRTRAN ALGŌL CŌBŌL などである。これらは1度演算装置に読み出されて機械語に飜訳され，その後命令レジスターに命令として入って初めて演算が実行されるのである。

ALGŌL は ALGO rithmic Language（算法言語）の略語で，主としてヨーロッパの大学で科学技術計算用に開発された言語で，1960年に発表された ALGŌL 60 がその基本となっている。他方 FORTRAN は FOR mlua TRANS lation（数式の翻訳）の略で，10 年前に IBM 社により，ALGŌL と同様科学技術計算用に開発された言語である。なお COBOL はデータ処理用に開発された言語である。

V. プログラムとプログラミング

電子計算機による計算は命令と呼ばれるオペレーションのつながりで実行されるが，この計算機用言語で書かれた命令のつながりをプログラムという。また計算機で問題を解くには，それを構成する細部の過程にまでたちいたって分析し，その個々の過程を命令のつながりとして実行できる形に書き直す必要がある。そしてこの問題を分析する作業をプログラミング，命令へ書き直す作業をコーディング，さらに作成したプログラムを検査する作業をデバッグイングと呼んでいる。

つぎに，詳しく説明すると，電子計算機のプログラムを書くには，下に示す五つの段階が必要である。

①問題の設定，②解法の選択，③問題の解析，④命令を書くこと，⑤検査。

すなわち，①プログラムを書く前に，その問題が明確にきちんと設定されなければならない。そして少なくとも，(1)与えられた情報，(2)必要な答，(3)答の精度，(4)データの与え方，(5)答の出し方などについての情報が必要である。②一つの問題に対して解法はいくつもあるのが普通であり，計算機の機構や動作，プログラミングやコーディングの方法を観察してプログラムが最適になる解法を採用する。③次に問題を解析し，所要の手順および手順間の関係を定める。解析結果を数学的な形式の手順の一覧表に書き，問題の処理手順の各部分とその順序を図示（流れ図；フローチャート）する。なおどの程度解析するかはプログラムと問題の両方から定まってくる。

以上でプログラミングが完了し，④のコーディングに入り，特定の計算機が実行できるオペレーションを使って詳細にプログラムを書き表わす。⑤プログラムを書いたら正しいかどうかを検査する。またプログラムを検査するためにチェック・データを使う。チェック・データは一般にチェックのための計算が人手でできるような簡単なものを用いるとよい。判断がいく通りもあるときは，すべての可能性について検査する必要がある。

次回以降，FORTRAN（川口，白石；農土試），ALGOL（内藤，中村和；農地局技術課，設計課），数値解析（白石，川口），統計的手法（内藤），オペレーションズ・リサーチ（中村充；農土試），自動設計（中村，坂上；設計課）などについてのべる予定でである。

〔1969. 2.18. 受稿〕