

ん。それには電子計算機の利用が、極めて大きな役割をもつてでありましょう。

1960年人口センサスのアメリカ合衆国における集計には ERA-1105 が使用されていると先刻申し上げましたが、センサス局が作製した FOSDIC を使用して、調査票、マイクロフィルム、磁気テープを経て電子計算機が調査票の内容を読みとって集計いたします

から、日本で3年かかる作業を1年でやってしまいます。日本においてもこのような mechanism が近い将来に用いられることを私どもは心から念願いたしておりますが、その実現は本日ここにお集りの皆様の御努力に期待申し上げるほかはないと存じます。どうぞよろしくお願い申し上げます。

御清聴いただき有難うございました。

電子計算機の総合演算の高速化*

〔計数形電子計算機 MELCOM の高速演算方式〕

豊田 準三**

1. あらまし

世界最初のプログラム記憶式電子計算機 EDSAC は遅延記憶方式のものであった。この方式は幾多の優れた特長を有しているため、種々の方式が開発された現在でも、中形以下のものには色々な形式で使用されている。MELCOM-LD 1† は磁気円筒遅延方式による直列2進式計算機で目下三菱電機研究所で調整進行中の研究試作機である。

MELCOM-LD 1 は本体と演算高速化装置 (FLORA) とから成り、そのいずれにも各種の独創的な演算高速化の手段が講ぜられている全トランジスタ式計算機である。本体は遅延線形磁気円筒装置を主記憶装置として、4046語の記憶容量と4個のインデックスレジスタを有する変形2アドレスの2進式計算機として単独の機能を有する。FLORA は数個のシフトレジスタと全加算器数個を主構成部とした特定演算加速装置で、各種の興味ある高速演算を行うものであるが制御はすべて本体により行われる。

主入出力装置としては電動タイプライタを用い、アルファベットモード、数字モード、両者複合モードの

3種の入出力モードがある。複合モードでは自動的にアルファベットは8ビット、数字は4ビットで記憶装

第1表 MELCOM-LD 1 の主要性能*

演算制御方式	直列内部2進式、遅延形磁気円筒方式 (付属高速演算装置 FLORA は直並列式)			
プログラム方式	内部逐次 (次命令指定) 式			
命令方式 (インデックス種類)	変形2アドレス式、インデックスレジスタ4個、約130種 (FLORA を含めて約150種)			
語の構成	命令語33ビット (符号を含む)、数値語33ビット又は66ビット (符号を含む) 浮動小数点 仮数25ビット又は58ビット (符号を含む)、指数8ビット			
演算速度 FLORAなしの場合	固定小数点	浮動小数点		
	加減	0.165 ms		
	乗	10.6 "		
	除	10.7 "		
	FLORAつきの場合	加減	0.165 "	0.66 ms
	乗	0.66 "	0.66 "	
除	2.15 "	1.65 "		
主記憶装置 (方式) (容量、アクセス時)	遅延式磁気円筒			
	一般記憶	4000語	8.3 ms (平均)	
	高速記憶	32語	0.33 " "	
		12語	0.165 " "	
		2語	ランダム	
入出力装置	自動タイプライタ (4種類のいずれか)			
	光電テープリーダー (4台まで)			
	紙テープパンチャ (4台まで)			
	磁気テープ装置 (4台まで)			
	カードリーダー (4台まで)			
	カードパンチャ (4台まで)			

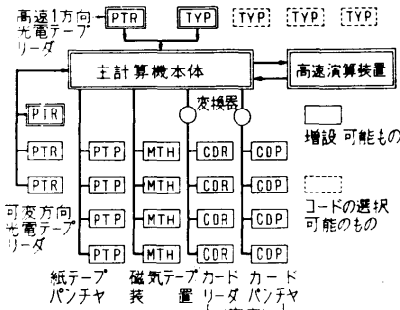
* High-speed Arithmetic Operations in the Digital Computer [Arithmetic Operation in MELCOM-LD 1], by Junzo Toyoda (Research Laboratory, Mitsubishi Electric Mfg., Co.)

** 三菱電機研究所 10月14日情報処理学会 関西講演会講演

† 本機を基本としての商品化 (MELCOM-1101) が無線機製作所で進捗中である。

* 本機を基本として製品化進捗中の MELCOM-1101 ではインデックスレジスタはなく、命令種類も整理されている。また入出力は同時に行われるようになっており、デジタル微分解析機として使用するための付加装置も計画されている。

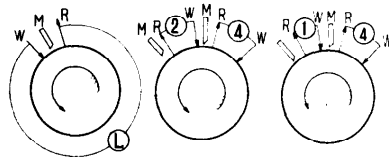
置に入れられるように工夫されている。出力に際しては上記各種のモードがフォーマットコントロールされるようになっている。付属装置としては光電テープリーダー、紙テープパンチャ、磁気テープ装置、カードリーダー、カードパンチャが各々4台まで取り付けられるようになっている。本計算機の主要性能は、第1表、構成概要は第1図を参照されたい。機械用語と翻訳用語による命令の主なもの、付表1、2に示してある。



第1図 総合機構

2. 遅延式磁気円筒装置

主記憶と演算の双方に使用され機械の心臓部となるもので、300φ×380のアルミ円筒に磁性皮膜を塗付させたものである。0.4kWの誘導電動機で駆動され約3600rpmである。長記憶トラック用と、短記憶トラック用の消磁用永久磁石がそれぞれ1個ずつ装備されている。クロックトラック、タイミングトラックは恒常的なタイミングパルス発生用として消磁は受けないようになっている。長記憶トラックは一般記憶用に用いられ、短トラックは高速アクセス記憶用のものと演算用レジスタとして使われるものがある。磁気円筒を輪切りにして構成の概要を第2図に示した。



第2図 ドラムの構造概要

- ①: 100 語長トラック
- ②: 2 語長トラック
- W: 書き込みヘッド
- ④: 4 語長トラック
- ①: 1 語長トラック
- R: 読み出しヘッド
- M: 消磁用永久磁石

2.1 4 則演算部

加減算用として1語長遅延トラック(33ビット)と

全加算器 FA 1個から成る累算器 AC1, AC2 および2語長遅延トラックと全加算器から成る AC3 がある。乗除算用としては2語長トラックと符号用フリップフロップから成る演算回路が3個(MQ, ID, PNと名付けられる。)あり、その内のPNには全加算器がついている。乗除算は絶対値と符号から成る数値形式(記憶形)で行われ、2進数の筆算をそのまま論理回路化したものである。MQは乗数または商を収容し、IDは被乗数または分母を収容し、PNは積または分子を収容するレジスタの役割を演ずる。これら相互間の信号のやり取りは、乗除算桁移動回路で制御されるようになっている。2進数演算である上にレジスタとしては磁気円筒の2語長トラックを起用しているので使用部品は非常に少くなっている。

AC1, AC2, AC3にはそれぞれトラック番号として56, 57, 58を割り当て、MQ, ID, PNには52, 53, 54が割り当てられている(第3図参照)。

2.2 記憶部

短アクセス記憶用トラックとして2語長のもの2トラック(L49, L48), 4語長のもの8トラック(L47~L40)がある。残部はすべて100語長トラックで40トラック(L39~L00)あり、これが通常の記憶用にあてられている。

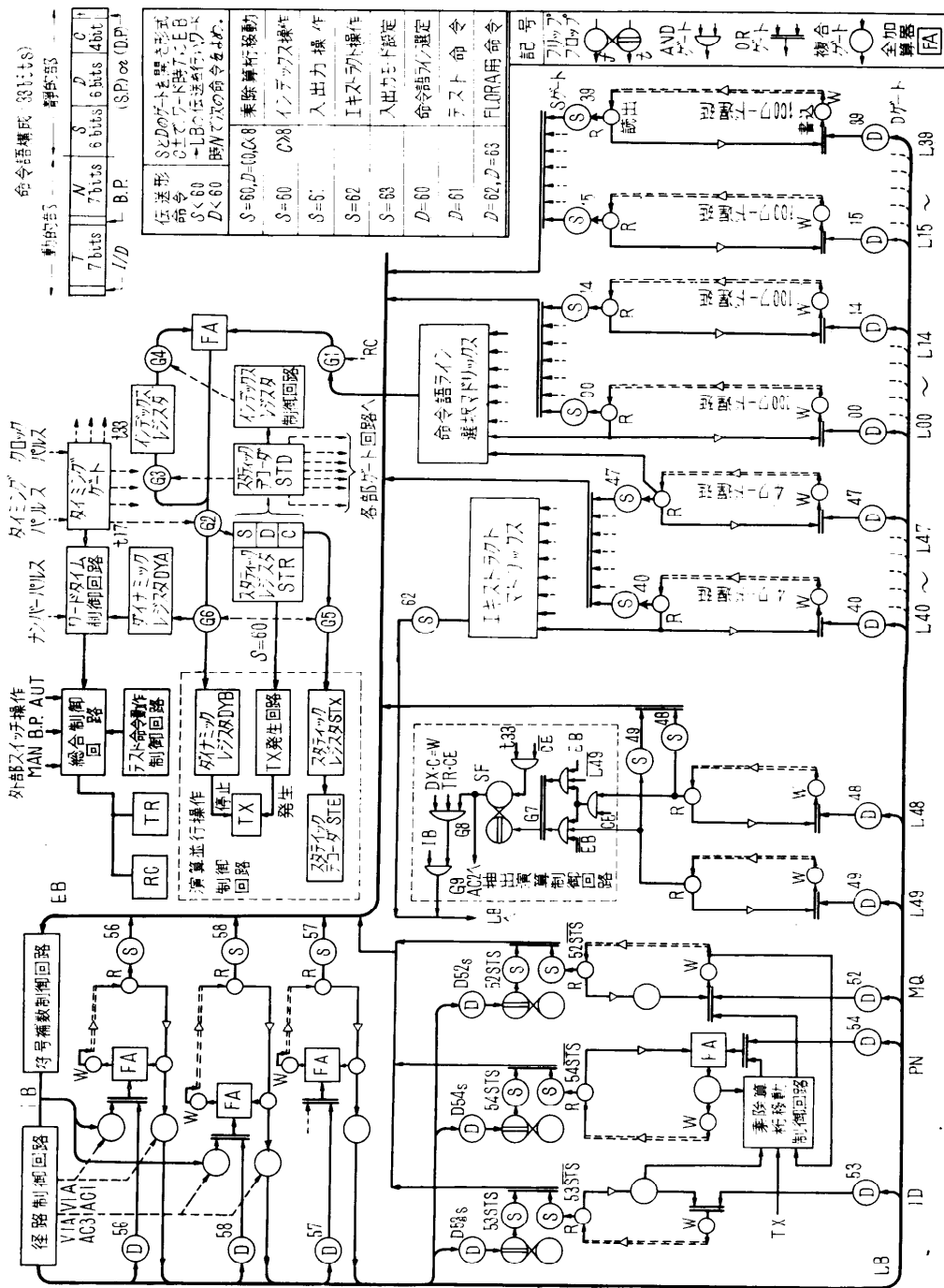
2語長のものは、165μs, 4語長のものは330μsの平均アクセスタイムを有し、100語長のものは最大16.5msで平均アクセス8.25msである。前項の演算部もランダムアクセスまたは165μsアクセスの記憶に使用できるので、合計して4046語の記憶容量となっている。

遅延線番号00~14の長語線、47番の4語線からは命令語が制御用レジスタへ読み出されるようになっているので、命令語としての活動ができる記憶場所は1504語あることになる。

入力情報は各種の入力装置より同期回路を通じて、L47に読みこまれ、続いてL39へ次々と順序正しく100語まで1ブロックとして記憶させられる。


出力情報はL39から100語まで、L47からは4語が同期回路を通じて各種の出力装置により外部に取り出せる。また出力装置が電動タイプライタの時はAC1, AC2, AC3からもタイプアウトできるようになっている。

L47~L40のトラックはANDゲートで数種の相互結合がなされ、色々のエキストラクト操作(16種)ができるようになっている(第3図参照)。



第3図 MELCOM LD-1の自動演算機能

3. 計算機の基本的状態

計算機が何ら演算活動をしていない状態では、各遅延線を信号パルスが、その語長に相当した個数の単位語（4語長線では4語分）分が直列に連らなって循環記憶されているだけである。第3図ではその様子を  で示してある。

計算機の活動状態には命令語を読み出しつつある時期 RC と命令語で指定された演算動作を遂行しつつある時期 TR がある。そのいずれでもない時期には RC を待つ場合 WRC と、TR を待つ場合 WTR とがある。これらの状態の制御は総合制御回路から発生する電圧 RC, TR で行われる。

総合制御回路には、外部のスイッチ回路の結線があり、外部の人手操作による特定の計算機状態の設定や入出力装置の操作、自動計算開始の操作及びブレークポイント操作を行うことができるようになっている。

（外部スイッチをブレークポイント BP に設定しておく、記憶プログラムによる自動計算中に命令語中の特定のビット（BP ビット）を検知して自動計算が停止する。BP はプログラム作成時にタイプライタ操作により命令語に付加することができるようになっている（第3図右端の表参照）。

4. 数値語と命令語

4.1 数値語

数値は内部2進数で記憶形としては33ビット（内符号1ビット）で示される短数値語（SP）と、66ビット（内符号1ビット）の複数値語（DP）の双方が命令で簡単に使い分けできるようになっている。正の命令語は前者を、負の命令語は後者を取扱うのである（後者には例外あり）。（SP）は入出力の関係で10進数8桁、（DP）は10進数16桁に相当している。

加減算には「2の補数」が使用され、加減算遂行後は累算器内では補数形で記憶されているが、これを一般記憶へ転送すると自動的に記憶形になる。また累算器内でも記憶形にすることがプログラム操作で可能である。

4.2 命令語

命令語の構造とその意味は第3図の右表に示してある。現在は標準約150種あり、その主なるものは付表に示してある。

第3図右表に見る如く、T, N, S, D, C 符号の6部から成立している。T部はTR電圧の発生時を、N部はRC電圧の発生時を計時するために使用され、共

に時間的に動的な動作を行うものである。S, D, C 符号の部は、どのようなゲートの設定（電子回路構成）を行うかに使用され、時間的に静的である。命令より取扱われる語の情報源としてのアドレスはSとTで示され、送り先のアドレスはDとTで示される。次の命令語を読み出すべきワード時もNで指定されているので、変形2アドレスと言われる命令語形式である。TとNの間の1ビットがBPビットであり、Tの左の最高位ビットは即時演算か、時間待ち演算かを指定するビットでI/Dビットと称せられる。

S<60, D<60の命令が基本形であり、これは表に示したような意味を有しているので、トラック間の情報転送形の命令と言える。

遅延記憶トラック番号には60以上のものがないがS, Dは共に63まで使用されている。S, Dが60以上の命令語は特殊命令であり、乗除算、エキストラクト、入出力に関係するもの、インデックスレジスタ命令等がこの部類である。この場合はC部は情報伝送の性質を示さないで、各々の命令の小分類を示すようになっている。

5. インデックスレジスタ

数値的機械用語による命令にはインデックスを示すビットがない。インデックス操作を行う命令としてはS=60, C>8があてられている（付表参照）。付表により理解できるが、4個のインデックスレジスタを指定し、その内容により、S, D, T, Nのいずれか或いはその組合せのモディフィケーションが可能であり、インデックスレジスタ内容数を置数する命令も用意されている。

シンボリック機械用語でプログラムする場合には、プログラム編成ルーチン（SIMPR）により、インデックス操作命令とインデックスビットを有しない命令とが自動的に対をなして編成されるようになっている。したがってシンボリック機械用語の命令にはインデックス指定の部分がある。SIMPR そのものは内部数値的的命令で作られていることは勿論である。

インデックス操作命令は直後に続いた命令のモディフィケーションを行うのみで、RCだけで終了する。したがって命令語一語として挿入されても演算速度への影響は極めて小となっている。

6. 自動計算における内部動作（第3図参照）

自動計算といっても最初の命令（したがって最初の

RC の発生) は人手により外部スイッチ操作で総合制御部を活動させることにより行われる。以後は自動的に、命令を RC で読み、TR で遂行し、RC, TR, RC, TR …… と計算機状態の推移が行われて自動計算が行われる。RC と TR の間に、WTR, WRC が無いようにすることはプログラミング技術であり、ミニマムアクセスプログラミングといわれるものである。

6・1 機械内部の情報信号伝送

各遅延回路を循環して記憶されている命令語を示す信号は、RC の発生により命令語ラインの一つから G1 なるゲートを通じて制御部の 2 種のレジスタへ導かれる。循環記憶されている数値語が、 $S < 60$, $D < 60$ なる命令で取扱われる場合には、T で示されたワード時に数値語を示す信号が S ゲートより EB 母線へ取り出されて、符号補数制御回路、IB, 経路制御回路を経て LB 母線より D ゲートより送り先のトラックへ送り込まれる。S または D が 60 より大なる命令の場合は必ずしも EB→LB の形式の信号伝送は生じない。

6・2 制御部への命令語の読み出し

総合制御部から RC が発生すると、これが G1 を開いて、命令語ライン選択マトリックスで選択されたラインから命令語情報の信号が導き出される(どのワード時で読まれるかは、プログラム上で先行した命令の N で定まる)。インデックス操作命令がプログラム上で先行している場合には、インデックスレジスタ、G4 および FA により先行しているインデックス操作命令で指定された部分のモディフィケーションが行われる。

モディフィケーションを受けた命令は G2 の動作により前半の静的部はスタティックレジスタ STR に入り、後半の動的部はダイナミックレジスタ DYA に導入される。読まれた命令がインデックスレジスタ内容の置数命令の時は、G3 を通じてインデックスレジスタへの置数動作が行われる。

6・3 読み出された命令により生ずる機能

STR に入った静的部は、スタティックデコーダ S TD によりデコードされて各種のゲートを動作させるべき電圧を発生し、命令語で指定された演算内容に対応した電子回路の設定が行われる。

DYA はワード時制御回路と共に計時装置を構成し、別に発生しているタイミング信号との時間的相互関係により、いつ TR を発生すべきかの制御動作を行う。この計時装置からの信号で計算機の状態制御が行われ、総合制御回路より TR が発生する。

TR が発生すると、既に静的に設定されている電子回路を通じて指定された演算操作が行われる。 $S < 60$, $D < 60$ の時は前述のように EB→IB→LB の経路で D で指定されたトラックに情報信号転送が行われるが、送り先が全加算器 FA を有するトラックの場合には、そこに循環記憶されている内容と FA を通じての加算が行われ、その結果が新しい循環記憶情報となり、これが加算操作である。加減算いずれの場合も、EB→LB の途中で符号処理を受けるので、累算器内では結果数は補数形となっている。途中で符号処理を受けないで、記憶形のまま他のトラックに転送を行う場合がコピー操作である。このように加算か減算かコピーかを指定するのが C 部の役目であり、転送時にはそれに適合した電子回路の設定が完了している。

乗除算、桁移動等では EB, LB は経由しないで、MQ, ID, PN の間で信号のやりとりが行われ、C 部はこの場合に乗、除、右桁移動、左桁移動等の小分類を示すように用いられている。

C 部はこのほか転送経路の指定にも用いられる ($S < 60$, $D < 60$ の場合)。すなわち一旦 1 語長トラック、あるいは 2 語長トラックを経由した後、LB よりの送り込みを行うという経路指定である。この場合 1 語 (VIA AC1) または 2 語 (VIA AC3) 遅れたワード時に信号の伝送が行われる。これ以上ずれたワード時に宛先へ送り込む場合は一旦アクセストラックに記憶させ、次の命令で所定のワードタイムに転送を行えばよい。この場合プログラム上で命令が 2 個必要となる訳である (1 語長トラックほどのワード時にも情報の受渡しできるランダムアクセスメモリである)。

6・4 即時遂行演算と時間待ち演算

TR を発生させる時期に 2 種類ある。RC が済んだ直後のワード時で発生する場合と、あるワード時待ち合せた後発生する場合である。乗除算、入出力の命令は前者に属し、加減算等は後者に属する。後者の場合でもプログラムがミニマムアクセスされている場合には自動計算途中では待ち合せのない能率的な進行が行われる。即時演算か時間待ち演算かは、命令語が読み出される最後のビット (I/D ビット) が 0 か 1 かで決定されるようになっている。

6・5 グループ演算

加算は通常遂行時間が 1 ワード時の時間待ち形の命令で行われるが、この命令の I/D ビットを 0 にしておけば RC の直後から TR が発生し、T 部に示されたワード時の直前まで継続されるような即時演算とな

る。すなわちこの命令が読み出された直後のワード時から、T-1のワード時までの数個の数値が累算器にグループとして加算されることになる。減算、絶対値加減算、グループコピイ等についても同様であり、この種のグループ演算が容易だということは、数個の加算をプログラム上で操作することの繁雑さを考えると、他には見られない遅延記憶方式の魅力的特徴であり、この方式はグループ演算につき本質的に高速といえる。

6・6 次の命令の読み出し動作

TRが終了すると、DYAとワード時制御回路で計時されて、Nで示されるワード時で次の命令を読み出すべきRCが発生して次の演算に移行する。ミニマムアクセスされたプログラムの遂行に際しては、TRの終了直後にRCが発生し、ここにも待ち合せ時間のないことはTRの発生と同様である。

テスト命令は、計算機回路のある個所で、あるワード時に、ある信号パルスが検出されるか否かをみるという演算動作である。何等の信号も検出されない場合はNワード時にRCが発生することは通常の場合と同様であるが、符号パルス以外のパルスが検出された時はN+2ワード時で、符号パルス以外のパルスが検出された時はN+1ワード時にRCが発生するようにテスト命令動作制御回路が働くようになっている。

以上で本機における自動計算の一駒の動作大要を示した。乗除算の詳細、入出力動作の詳細は紙面の都合上割愛する。これらの自動計算の進行はすべて一定のタイミングで行われる。そのタイミングの最小単位がクロックで、その外命令語内の各部の区切りに相当するタイミングパルス、長トラックのワードタイムを示す数を各ワード時ごとに発生させるナンバーパルス等が刻時的電気信号として発生するようになっている。

7. 演算の並行操作による高速化

一般に乗除算、桁移動等は加減算の多数から成立したものとみなされるので、後者に比して演算に長時間を必要とする。したがって両者を同時に並行的に進行させた場合、総合演算速度の向上が期待できるわけである。

第3図中のSTX、DYB等はこのような操作を行うために設けられた論理回路である。命令語の静的部がSTRに入った場合に、もしS=60に対応した信号があれば、この特定の静的部のデコード電圧「S=60」により、ゲートG5が動作してSTRの内容がSTX

に導入され、これがさらにSTEでデコードされて、乗除算回路の電子回路設定が行われる。引続く動的部に対してはG6が動作してDYAへ入らずにDYBに導入されるようになっている。S=60なる電圧はTX発生回路を動作させ、RC終了直後からTXが発生する(即時遂行演算)。このTXはMQ、ID、PN相互間の信号伝送を制御するものであり、既に設定完了している回路を通じて乗除算等が遂行されることになる。

DYBはDYAと同様に計時動作を行うものであり、TXの継続時間を規定する。すなわち何桁乗算するか、何桁移動するかがこれにより制御されるのである。

乗除算、桁移動等が進行中は、STRもDYAもアイドルである。またMQ、ID、PNに関するSゲート、Dゲートは閉であり、これらの回路はEB、LBとは切放されているので、一方においてEB→LBの形式の命令が遂行可能である。すなわち乗除算、桁移動の命令と、加減算、判断等の命令の遂行が並行的に行われうることになる。この手法は数学的演算の総合速度の向上に有効なものであるが、formulae translationあるいはsymbolic interprete(これらには桁移動操作が多く含まれる)に適用して最も効果が大きいと思われる。

8. 対応情報の抽出演算(辞書ひき操作)

特定のコードに対応した数値的情報を抽出する(辞書をひく)操作は、数表の探索と共にプログラムで処理するには厄介な演算の一つである。この演算をエキストラクト演算の特殊な場合として取扱うための論理回路が第3図抽出演算制御回路である。

この演算を示す命令はS<60, D<60, C=W(Wは2進数1100のローマ字表示)で示される即時遂行形の演算命令である。例えばS=30, D=31の場合にはそれぞれのゲートが開となり、TR時にはL30から絶えずEBへ情報信号の送付がある。これがLBに出現しさえすればD=31のゲートを通じてL31へ送り込まれるようになっている。LBに信号が出現するのを抽出的に制御するのが抽出演算論理回路で第3図の記号によればゲートG7の出力を示す論理式は次の如くなる。

$$G7 = EB \cdot \bar{L}49 \cdot F + EB \cdot L49 \cdot \bar{F} \quad F = CE \cdot L48$$

$$CE = \text{偶数ワード時項} \quad CE = \text{奇数ワード時項}$$

今L48にall 1(11……11)が記憶されているとすると上式は「偶数ワード時のEB上の信号とL49か

らの信号が完全に一致しない限りは、偶数ワード時中のいずれかのビットタイムに出力がある」ことを示している。G7の出力でSFがセットされるから次の奇数ワードではSFの出力はなく、したがってG8の出力もない。だからL30よりの信号はLBには出現しない。逆に双方が完全に一致しておれば、先行した奇数ワード時の最終パルス $CE \cdot t_{33}$ によりSFはリセットされたままであるから後続の奇数ワード時には、 $TR \cdot CE \cdot DX \cdot C = W$ の項の存在により、G8の出力が生じIB→LBの情報信号転送がG9を通じて行われ、奇数ワード時に抽出さるべき情報がL31へ送り込まれることになる。L49は2語長であるので、その偶数ワードに抽出用のコードを記憶させておけば、この演算により100語長のトラックからは奇数ワード時に最大50個までの対応情報が一つの命令で抽出できることになる。

上記の演算でSFの信号をAC2へも導くことにより抽出された個数を計数することも可能である。

抽出用のコードが1ワードの小部分である場合には、L48の対応した領域にのみall 1を記憶させて

第2表 情報抽出加算プログラム

LINE 00							
L00	T	N	S	D	C	NOTE	
00		02	03	47	49	0	Extractor Partial field
03		04	05	47	48	0	
05	u	06	06	00	20	z	CLR working line
06	u	07	07	00	21	z	
07		08	09	00	57	z	
09		10	11	00	56	z	
11	u	12	13	10	20	w-	SELECTIVE COPY WITH COUNT
13	u	14	14	11	21	w-	
14		15	16	57	61	z	
16		17	00	01	60	z	Jump to 01.00
17	u	18	18	20	56	5	Sum
18	u	19	16	21	56	5	

先行して記憶されている内容
 (47.00)=フィールドディング情報
 (47.02)=品名コード(エキストラクター)
 (L10): 偶数ワードは品名コード, 奇数ワードは金額
 (L11): " " "
 演算結果: AC1に金額総和, AC2に個数
 [註] 01.00は2進数→10進数変換及び10進数のタイプアウトルーチンの最初の命令が記憶されているアドレスである。

おけばよく、これは一種のフィールド操作である。この命令の具体的応用としては、多数の品名と金額

第3表 シンボリック命令語の変換プログラム

LINE 05							
LOC	T	N	S	D	C	NOTE	
59		01	28	60	00	0	CLR MQ, ID, PN.
28		30	05	61	00	v	Gate type in
05		05	05	00	61	x	Wate ready
06		08	09	47	56	4	(47.00)--AC1 Lower typed inform.
09	u	14	14	56	44	0	(AC1)→L44
14		15	16	04	48	0	000000 zz (04.15)→48.01
16		17	18	62	57	6	(48·AC1)→AC2 Least signif. 2 ch.
18		19	20	34	57	9	$\frac{1}{2} \times 2^{-32}$ (34.19)→AC2
20		21	23	57	61	z	-(AC2)≠0?
23	u	28	28	00	47	z	→CLR L47
24		25	26	47	61	z	→(47.01)≠0?
26		29	30	62	57	4	Uppertyped inform. =0(43.44), -AC2
27		28	31	47	56	0	None addressed. →≠0(47.00)→AC1
31		32	33	34	48	0	With address. 0z000300 (34.32)→48.00
33		34	35	62	53	6	*s (48·AC1) ₀ →ID ₀
35		14	36	60	00	5	Shift 14 bits (ID) ₀ =*s
36		37	38	62	49	6	*i*s (48·AC1) ₁₁ →49.01
38		39	41	62	56	7	*s*i*s* (48·AC1) ₁₁ →AC1
41		42	43	62	56	7	*i*s* (48·AC1) ₁₀ →AC1
43		44	45	43	48	0	00zz0000 (43.00)→48.00
45		46	47	62	58	7	α (48·AC1)→AC3
47		48	49	62	56	6	*i*s* (48·AC1) ₀ →AC1
49		50	51	04	48	0	000z0000 (04.50)→48.00
51		52	53	62	57	7	*t (48·AC1)→AC2
53		54	55	62	48	6	*u (48·AC1)→48.00
55		56	58	48	56	5	*u (48.00)→AC1+
58	u	61	61	56	56	5	$4 \cdot \frac{1}{2} + \frac{1}{2} \times 8$ (AC1)→AC1+
61		62	63	56	57	5	$5 \cdot \frac{1}{2} + \frac{1}{2} \times 8$ (AC1)→AC2+
63	u	69	69	57	57	5	(AC2)→AC2+
69		70	71	34	49	0	5 bits left shift zzzzzzzz (field) (34.70)→49.00
71		73	74	47	61	z	-(47.01)≠0? (addressed?)
74		75	78	62	57	4	--=0 (43.44) _s →AC1
75		76	77	53	57	5	→≠0 $\frac{1}{2} \times 2^{-32}$ (ID) ₀ →AC2
77		78	79	58	56	4	α (AC3)→AC1
79		80	81	34	56	9	"t" $\times 2^{-34}$ ad (34.80)→AC1
81	u	86	86	56	56	5	(AC1)·AC1+
86		87	88	56	57	5	4 bits left shift *bad*s
88		89	91	47	48	u-	(AC1)→AC2 (47.01) _s →48.00
91		93	94	49	57	5	cva *i*s *bad*s*i*s (49.01)→AC2+
94		84	97	00	56	z	CLR AC1
97	u	98	99	37	56	w	SELECTIVE ADD *OP from L37
99		00	32	57	56	5	(AC2)→AC1
32		36	60	43	48	0	(43.00)→48.00 00zz0000
60		61	80	05	60	2	Jump to 05.80

To none addressed command conversion

To none addr. cmd. conversion for #OP=110~106

中から特定の品名に対応した金額のみ抽出し総和（平均値をも）を求めるような場合に適用できる。第2表はこのようなプログラムの一例である。このような論理を有しない場合に、これだけのことをプログラム操作する手間と演算時間に比較すれば極めて能率的であることは理解できるであろう。

この命令はシンボリックな外部命令を、数値的な内部機械用語に翻訳する Interpreter にも能率的に使用できる。第3表は MELCOM-LD1 を付表2に示したようなシンボリック単アドレス自動中央小数点式計算機として使用するような命令に変換するための翻訳ルーチン ASIA の一部で数値的外部機械用語によるプログラムで示した。これよりシンボリックにタイプインされた命令が、付表2-(4)に示されたような内部2進数的構造を有する命令に変換される。変換され

たものは T. N. S. D. C と同一構造となっているので命令の遂行時に必要となる翻訳操作が単純化される。外部のシンボリックオペレーションコードと内部数値コードの対照表は、L35, L36, L37 の3長トラックに辞書として記憶され、これを対応情報抽出演算により辞書ひき操作が LOC 97 の命令で行われるようになってい。この命令に到達するまでに、オペレーションコード以外の部分の10進数→2進数変換、桁移動等が行われて、最後に上記辞書ひき操作が行われて変換を終り、LOC 35 の命令で別のルーチンへとボタンタッチされるのである。

なお LOC 35 の桁移動命令の遂行と並行して、2桁の10進数→2進数の変換が行われていることを2重の縦線で示してある。並行演算操作の例として示した。

第4表 高速演算装置による演算の種類と所要時間

分類	命 令				動 作 内 容	所 要 時 間 WT*
	操 作	P	D	C		
基 本 演 算	FL. Add		63	5	x を (ER) に加えて結果を ER に残す。	4 (8)
	FL. Subtract		63	9	x を (ER) から引いて結果を ER に残す。	4 (8)
	FL. Multiply		63	x	x を (ER) に掛けて結果を ER に残す。	4 (8)
	FL. Divide		63	z	x で (ER) を割って結果を ER に残す。	10 (42)
	FX. Multiply		62	x	x を (AR) に掛けて結果を AR に残す。	4 (8)
	FX. Divide		62	z	x で (ER) を割って結果を ER に残す。	13 (48)
数 変 換	Dec. to Bin		62	4	AR にある 10 進数が 2 進数になって ER に入る。	1 (2) **
	Bin. to Dec.		62	5	AR にある 2 進数が 10 進数になって ER に入る。	2 (4) **
群 演 算	FL. Block. Add	u	63	3	$A_i + B_i$ を C_i に入れる。	4 (8) **
	FL. Block. Subt	u	63	6	$A_i - B_i$ を C_i に入れる。	4 (8) **
	FL. Block. Mult.	u	63	y	$A_i \times B_i$ を C_i に入れる。	4 (8) **
	FL. Block. Scale	u	63	w	$(AR) \times B_i$ を C_i に入れる。	4 (8) **
	FX. Block. Add	u	62	3	$A_i + B_i$ を C_i に入れる。	1 (2) **
	FX. Block. Subt	u	62	6	$A_i - B_i$ を C_i に入れる。	1 (2) **
	FX. Block. Mult	u	62	y	$A_i \times B_i$ を C_i に入れる。	4 (8) **
	FX. Block. Scale	u	62	w	$(AR) \times B_i$ を C_i に入れる。	4 (8) **
探 表 演 算	FL. Max. Select		63	1	Max(X_i) を ER に、対応する i 番号を AR に入れる	1 (2) **
	FX. Max. Select		62	1		1 (2) **
	FL. Table Look		63	7	X_i のうち $X_i > (ER)$ である最初のものを選んで ER に入れ、i 番号を AR に入れる。	1 (2) **
	FX. Table Look #1		62	7		1 (2) **
算	FX. Table Look #2		62	8	$X_i = (ER)$ をみつけて i 番号を AR に入たる。	1 (2) **
	FX. table Look #3		62	9	$X_i = (ER)$ をみつけて A_i を AR に入れる。	1 (2) **

* () 内は2倍精度の場合、** 10進1ケタあたりの所要時間

1 WT (ワードタイム)=165 μ s

*** 群の中の1個の数の処理に要する時間

FL. 浮動小数点演算, FX. 固定小数点演算

付表 1. 機械用語による命令の外部形式 (数値形式とアルファベット形式)

命 令 の 内 容	Pre- fix	(Numeric)					Suffix	(Alphabetic)
		T	N	S	D	C		I*W*, (OP)/S*T*N#.
(S.T)をそのまま D.T コピイせよ	(u)	T#	N#	S#	D#	0	(-)	(I#)(W#), t D#/S*T*N#.
(S.T)の符号を逆にして D.T にコピイ	(u)	T#	N#	S#	D#	1	(-)	(I#)(W#), tchD#/S*T*N#.
(S.T)と (AC1)とを交換せよ		T#	N#	S#	D#	u		(I#), tvxS*T*N#.
(AC1)→D.T; (S.T)→AC1	(u)	T#	N#	S#	D#	u		(I#), tuxS#/S*T*N#.
(ライン S)をライン D へ選択コピイ	u	T#	N#	S#	D#	w		(I#) W#, socD#/S*N#.
エキストラクト (40・41)→D.T	(u)	T#	N#	62	D#	0	(-)	(I#)(W#), et 41/D*T*N#.
クリヤして (S.T)を加算せよ		T#	N#	S#	A#	4	(-)	(I#), acl&/S*T*N#.
アキュムレータの内容を D.T へ記憶せよ	(u)	T#	N#	A#	D#	4	(-)	(I#)(I#), aclD#& T*N#.
(ID)×(MQ)を PN へクリヤして加算せよ		B#	N#	60	00	1		(I#)(W#), mpy/N#.
(PN)+(ID)を MQ へ残せ (S.P)		B#	N#	60	00	2		(I#)(W#), div/N#.
(MQ)を右へ B# だけ桁移動せよ		B#	N#	60	00	4		(I#)(W#), sh1/N#.
(ID)を左へ B# だけ桁移動せよ		B#	N#	60	00	5		(I#)(W#), shr/N#.
(I#)で後続命令のD部をモディファイせよ		N#	60	I#	0	u		I#, imd/N#.
” T ”		N#	60	I#	3	u		I#, imt/N#.
(I#)でS部を (I+1)でT部をモディファイせよ		N#	60	I#	1	v		I#, imst/N#.
(I#)をPだけ増加せよ (P=1,2,4,8)		N#	60	I#	P	w	-	I#, inc (p#)/N#.
(I 番インデックス内容+P)<Q#? のテスト	Q#	N#	60	I#	P	w		I# Q#, inb(p#)/N#.
AC1 の T 部を I 番インデックスに移せ		N#	60	I#	0	x		I#, ist/N#.
I 番インデックスの内容を AC1 の T部に加算せよ		N#	60	I#	1	y		I#, iadd/N#.
I 番インデックスの内容として Q# を書き込め	Q#	N#	60	I#	0	z		I#, iset/N#.
T# にマークを付して次は C.N の命令を読め	T#	N#	C#	60	0			(I#), jmpT#/C*N#.
無条件に C.N へ飛べ	T#	N#	C#	60	2			(I#), junc/C*N#.
ST の内容を AC1に加算してAC1の命令を実行せよ	T#	N#	S#	60	5			(I#), jax/S*T*N#.
自動計算を一旦中止せよ	T#	N#	00	60	z			halt/N#.
(40・41)≠0? のテスト, =0 ならば N. ≠0 ならば N+1 へとべ	(u)	T#	N#	00	61	0	(-)	(I#)(W#), bt 41/T*N#.
オーバフロ? のテスト	T#	N#	00	61	v			bof/N#.
(S.T)<0? のテスト, >0 ならば N., <0 ならば N+1 へとべ	(u)	T#	N#	00	61	y	(-)	(I#)(W#), bng/N#.
(S.T)≠0? のテスト	(u)	T#	N#	00	61	z	(-)	(I#)(W#), bnz/N#.
M 番の磁気テープへ L 39 の内容を書き込め	L 2#	N#	61	M#	0	2		M# wmt/N#.
M 番のテープリダ上の停止コードを順方向に探せ	L 2#	N#	61	M#	K#	4		M#K#, dfsp/N#.
” ” 逆方向に探せ	L 2#	N#	61	M#	K#	4	-	M#K#, drsp/N#.
AC1 の内容を M 番のタイプライターでタイプアウトせよ	L 2#	N#	61	M#	1	8		M#0, typx/N#.
L 39 の内容をM番のエープバンチャでバンチャせよ	L 2#	N#	61	M#	0	u	(-)	M#0, ppt(s)/N#.
M 番の磁気テープを読めとれ	L 2#	N#	61	M#	0	x		M#0, rmt/N#.
Nモード入力に設定せよ	L 1#	N#	63	00	6			form/N#.
Cモード	L 1#	N#	63	00	8			forc/N#.
その他合計約 150 種								

(例 1) アドレス 26.12 の内容を CA1 に加算せよ 12 34 26 56 5

(例 2) インデックス No. 2 でモディファイされたアドレス 26.12 の内容を AC1 に加算せよ 200, addx/2612 34.

(註) プレフィックス u は群演算を示し, サフィックス-はダブルプレジジョンを示す.

I# = 0~3 (インデックスレジスタ番号), M# = 入出力機械番号

W# = 群演算におけるワード数, K# = テープのブロック数

& = x, y, z のいずれか (AC1, AC2 AC3 の記号)

9. FLORA による各種高速演算

この付加装置は本体の遅延方式による演算と協調する直並列演算回路により基本演算の高速化と高級な演算の回路化による計算機機能の拡充を目的としたもので、構成概要は第 4 図に示した。これにより処理できる演算の種類と内容、および所要演算時間は第 4 表に示してある。第 4 図の E および F レジスタはそれぞれ (33+33) ビットのシフトレジスタであり、後者には

15個の全加算器が分布配置されている。E レジスタと F レジスタの間に一群のゲートが設けられているが、これら 3 者で直並列の高速乗算が行われ、この装置の中心をなす部分である。R レジスタは 8 ビットのシフトレジスタで第 4 表に示した各種演算を制御するために必要となる情報が一時的に記憶される。

演算により内容が異っているが、主遅延記憶部→E (または F) レジスタ、あるいは E→F レジスタの情報伝送の組合せであり、これに R レジスタが参加し

付表 2. 翻訳命令系 ASIA で用いられる命令群

(1) 相対アドレス設定のタイプライタ操作 (合計 6 種)

u=#
x=#

(2) アドレスのない命令群 (合計 23 種)

(外部コード) (内部コード)

fx-1 122 AOレジスタの固定小数点数を浮動小数点数になおせ

tloc 120 自動計算停止直前に遂行された命令の LOCをタイプライタせよ

halt 119 自動計算を停止してマニュアルモードにせよ

bell 118 ベルを鳴らせ

t#c# 111 タブ数を # に、キャリッジ数を # に設定してタイプライタ操作せよ

#b→b# 108 # 番の B レジスタ内容を # にせよ

a→d# 104 AOレジスタ内容を # 番 D レジスタへコピーせよ

(3) アドレスのある命令群 (合計 50 種)

(外部コード) (内部コード)

OP アド OP
コード レス コード

ib<c 0##a# 91 (#B) に (#D) を加算した結果が (#B) < (#C) ならば ##a# へとべ

cad 0##a## 89 (##a#) を # 番 A レジスタへクリヤ加算せよ

mlt 0##a## 83 # 番 A レジスタ内容に (##a) を乗じ結果を # 番 A レジスタに残せ

div 0##a## 82 # 番 A レジスタ内容を (##a) で除し結果を # 番 A レジスタに残せ

*add 0##a# 79 000a0~##a0 の内容を # 番 A レジスタへ群加算せよ

mean 0##a# 75 000a0~##a0 の平均値を計算して # 番 A レジスタに残せ

sqrt 0##a## 74 (##a#) の平方根を # 番 A レジスタに残せ

nlog 0##a## 72 (##a#) の log e を作り # 番 A レジスタに残せ

jump 0##a# 67 無条件に ##a# の命令へとべ

ja=0 0##a# 65 # 番 A レジスタ内容が =0 ならば ##a# の命令へとべ

pntp 0##a# 58 000a0~##a# の内容を紙テープへパンチせよ

fxcr 0##a# 56 ##a# にある固定小数点10進数をタイプライタしてキャリジレタンせよ

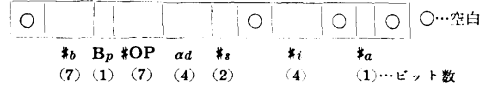
cmtb 0##a# 53 ##a# にある命令をタイプライタしてタブ動作を行え

rdtp 0##a# 45 テープリードして 000a0~##a# へ記憶せよ

cmin 0##a# 44 命令をタイプライタして ##a# へ記憶せよ

fxin 0##a# 42 固定小数点をタイプライタして ##a# へ記憶せよ

(4) 翻訳命令が内部で記憶されている形式

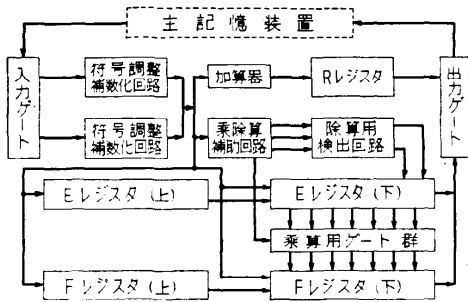


ad; u~z のデコードされた 2 進数
 #s; アドレスのサフィックス (0~3)
 #b; アドレス #i #u (00~99) に対応した 2 進数
 #i; インデックス番号 (0~9)
 #a; A レジスタ番号 (0 又は 1)
 Bp; ブレークポイントビット (0 又は 1)

(5) 数値のタイプライタ形式

固定小数点数	整数部 8 桁	小数部 8 桁まで
(例)	12345678.98765432	1960.12
浮動小数点数	仮数部 14 桁まで	指数部 2 桁
(例)	19601225987625/50	(19601225987625 × 10 ⁻²)
	-123/55	(-123 × 10 ³)

(註) 記号 #.....10進数 1 キャラクタ (0~9)
 ().....タイプライタのタブ操作
 a.....u, v, w, x, y, 又は z (u~z レジスタ)
 A0.....0 番の A レジスタ (等)
 #b.....# 番の B レジスタ
 (0025u0).....25u 番地の内容を示す
 アドレス表示形式 (例)
 0 #s #i #ua #i 0025u3
 3 番のインデックスレジスタ, および u レジスタの内容でモディファイされた 25 番地



第 4 図 高速演算回路の構成

て、いろいろな機能が生ずるようになっている。詳細の説明は別に発表が予定されているので、ここでは機能の紹介だけにとどめておくことにする。

あとがき

電子計算機の総合演算の高速化の手法を MELCO M-LD 1 を引用して具体的に説明した。本機は研究試作機であり、機器構成上の経済性の考慮は充分でない。また命令語の種類を欲張りすぎた感がないでもない。これらを整理したのが MELCOM-1101 であり目下製品化進捗中である。

参考文献

- (1) 高橋, 西野; マンチェスタ大学の高速桁上げ回路の追討, 情報処理, Vol. 1, No. 1. (1960)
- (2) 穂坂; 遅延線によるブロック演算, 情報処理, Vol. 1, No. 1. (1960)
- (3) 畔柳; 高速演算方式, 電子計算と制御, Vol. 2, No. 8~9. (1960)
- (4) 山下英男監修; “電子計算機”(昭和 36 年 6 月, オーム社)。