入出力制御方式*

石井善昭**

1. 緒言

初期の電子計算機は，おもに科学技術計算に使用され，入出力データ量が少なく，接続される入出力装置も限られたものであったから，入出力制御方式も能率の低い簡単な方式がとられていた．その後，電子計算機のおもな応用分野が事務処理になるにしたがって，入出力されるデータの量は飛躍的に増大し，一方新しい多種多様な入出力装置が開発，実用化され，そのデータ移送速度も著しく向上したので，電子計算機にしめる入出力制御の重要性は非常に大きくなってきた．これにともなって入出力制御技術も進歩し，電子計算機の論理構成の面からみると，おそらく最も著しい進歩をしたものは入出力制御方式ではないかといわれている．現在では科学技術計算のための電子計算機においても，Multi Accessed Computer のような応用分野が開拓され，入出力能力の重要性はなにも事務処理に限られなくなってきているし，さらに多重プログラミング技術が実用化されるにおよんで，入出力能力は単にその電子計算機システムの融通性，拡張性を規定するだけでなく，絶対的な処理能力ひいては Cost/Performance を決定づける重要な要素となっている．

入出力制御方式で考慮すべき点は次のようなところにあると考えられる．

（1） 入出力装置と中央処理装置の速度が桁違いに異なるので，中央処理装置の動作を拘束することがなく，しかも逆にこれを利用したより経済的な方式を達成しなければならない．

（2） 入出力装置は種類が多く，しかも，それらが1台の中央処理装置に任意の組み合わせで接続できなければならず Interface に一般性をもたせる必要がある．しかも，それは将来予想される入出力装置にもできるだけ適合性をもつ必要がある．

（3） 入出力装置の種類の増大につれ，他社の入出力装置の接続が要求されることがある．このときは障害時にその原因がいずれにあるかが明確になり，保守が容易に行なわれるようにしなければならない．磁気テープ装置，通信制御装置などが現実に，あるいは近い将来にこの可能性をもっている．

2. 主記憶と入出力装置間のデータ移送の方式

入出力制御は，基本的には電子計算機の主記憶と入出力装置との間のデータ移送をいかに行なうかの制御である．ここで最も大きな問題は主記憶部の速度と入出力装置のデータ移送速度があまりに異なることであり，これを解決すべく大きく分け3段階の進歩があった．

2.1 入出力動作中演算を止める方式

入出力動作を行なっている間は，中央処理装置の演算を止めてしまう方法は初期の計算機でとられた方式で，(a) 直接プログラムで入出力のデータ送受を制御する方式と (b) 一つの命令により入出力データ送受は自動的に行なわれる方式がある．(a) の代表的な例として IBM 701[1] の入出力制御方式がある．この電子計算機では Write あるいは Read 命令によって入出力装置を選択し動作準備をさせると，あとのデータ送受は1語ごとに Copy という命令で行なわなければならなかった．したがって Copy という命令を伝送す

Write (or Read)
Copy
Copy
⋮

第1図

べき語数に等しい回数実行しなければならないが，入出力装置とのタイミングを考慮すれば Copy と Copy の間に他の演算命令を実行することができ，原理的に内部演算との同時動作は不可ではないが実用にはならなかった．(b) の例としては，IBM 705[2] があげられよう．IBM 705 では，たとえば Write という命令が出されると主記憶の中に特定の記号が検出されるまで自動的にデータの送出が行なわれる．これらの初期の電子計算機では入出力データのための特別な Path はなく，演算に使われるレジスタを使用する[1,2]（たとえば Word と Character の変換に乗算回路のシフ

* Input/Output Control Technique, by Yoshiteru Ishii (Nippon Electric Co. Ltd., Tokyo)
** 日本電気株式会社

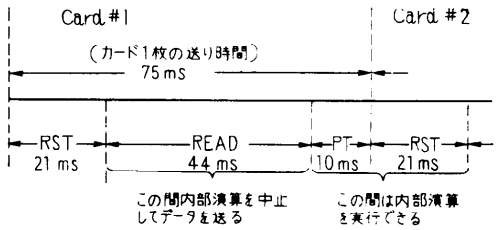
ト機能を利用するなど）ため内部演算と入出力の同時動作が不可能になっている．しかしながら，入出力装置の速度に比して内部処理は十分速いので，二つの入出力動作を同時に制御することは可能で，IBM 705 では磁気テープの読み取りと書き込みを同時に行なうことができる．

このような入出力動作中は演算を止めてしまう方式は，

（1） 入出力制御のために特別な回路を必要としないので，安価に電子計算機を構成することができる．

（2） 処理が完全に sequential に行なわれる．すなわち，入出力動作実行中にプログラムが先行するということが起らないので，プログラムの作成に特殊なテクニックを必要としない．

（3）電子計算機システムの使用効率が非常に悪い．演算実行中は入出力装置が止り，入出力装置動作中は中央処理装置が止るので，電子計算機システム全体をみると常に一部しか動作していないことになり高価な装置が活用できない．この点を少しでも改善するために前述した磁気テープの同時読み書きや，IBM 1401[3] でとられているような実際にデータの伝送が開始するまでは，次の処理を進めるという方式が考えられた．たとえば，IBM 1401 ではカードの読取命令を出してから，カードが送られて実際のデータの伝送がはじまるまでの時間を内部演算に割当てている．同様なテクニックが高速製表印字装置の紙送り時間などにも適用されている．



第 2 図

入出力動作中内部演算をとめてしまう方式は (1) および (2) の利点により安価で低速な電子計算機で採用されることもあるが，高速，高価な電子計算機では(3)の欠点が致命的である．最近の電子計算機の高速化は著しく，小形で安価な電子計算機においてもこのような方法はほとんど採用されなくなってきている．

2.2 Buffer（Fixed Block Buffer）方式

電子計算機の主記憶が磁気ドラムから磁気コアにかわると，入出力装置との速度の差は格段とひらき，入出力動作中演算をとめる方式から Buffer Memory をもちいて，この速度の差を調整する方式が用いられるようになった．この方式ではデータの入出力は各入出力装置のもっている高速な Buffer Memory と主記憶の間で行なわれ，演算が中止されるのはその間だけであるので，入出力動作のための無駄な遊びはなくなる．通常各入出力装置には単位データ量（カード1枚分あるいは印字 1 行分）に等しい容量の Buffer Memory をもたせ次のように動作する．

〔入力の場合〕

（1） 入力バッファが空のときは自動的にここにデータを読み込んでおく．これは中央処理装置と独立に行なわれる．

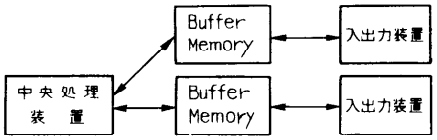
（2） 入力命令が出されると，バッファから主記憶へデータが送られる．

（3） 空になった Buffer に，再び次のデータを読み込んでおく．

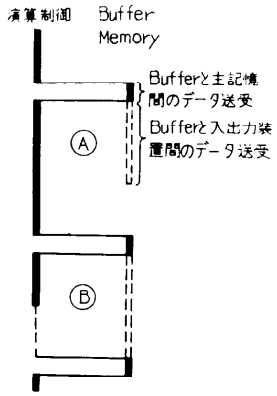
〔出力の場合〕

（1） 出力命令によって，単位量のデータが出力バッファへ主記憶から送られる．

（2） 中央処理装置は次の処理にすすみ，出力バッファから出力装置へのデータ移送は独立にすすめられる．



第 3 図



第 4 図

したがって，単位量のデータを処理するのに必要な演算時間がそのデータの入出力動作に要する時間より長い場合（**第4図**Ⓐ）には演算制御部は遊ばないが，同図Ⓑのように演算時間の方が短くなると遊びが生ずる．この間，他のプログラムの処理を行なえればこの無駄もなくなるわけで，ここではじめて多重プログラム処理（Multi-Programming）が可能となってくるが，NEAC-2230 ではⒷのような状態が生ずると自動的に他のプログラムに制御が移り，この待合わせ状態が解消されると再び制御が戻るようにハードウェアで制御している．バッファ方式の特色は，

（1） 入出力動作によって専有される時間が Buffer Memory と Main Memory 間のデータ送受の時間に限定されること．

（2） 多数の入出力装置を内部演算と同時に動作させられること．これによって多重プログラム処理が可能になる．

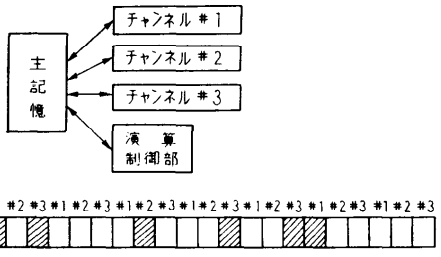
（3） 高速記憶装置が各入出力装置ごとに必要となること．これが最大の欠点である．

（4） 固定長の単位データ量を扱う入出力装置では問題ないが，磁気テープのように可変長のデータを記録する装置では，バッファの容量に注意を要するとともに，必要に応じてバッファを分割して Dual Buffer として取扱えるようにする必要がある．

（5） 他社の入出力装置を接続する場合のように責任の分担を明確にするためには，制御部にバッファをもたせる方が取扱いに便利になることがある．さらに後で述べる Multiplexor Bus Capability を使う制御装置(たとえば多重通信制御装置)では，各計算機の入出力チャンネルの性質が極めて密接に制御装置の設計に関連するが，かかる制御装置を統一する場合には各計算機の Multiplexor Bus Capability の性質の差を補償するかなり複雑なアダプタが必要となる．しかし制御装置に必要な大きさのバッファをおくことにより，Mulfiplexor Bus Capability を使用しないようにすることが可能で，このときには責任の分担が明確になる上に，バッファの価格増加はむしろアダプタの価格の減少に比し小さくなることも多い．

2.3 Direct Buffer 方式

以上のべた方式は一般的には独立した入出力データ送受の制御部を持たなかったのに対し，Direct Buffer 方式は入出力制御部を演算制御部から独立させ，この二つの制御部が主記憶を時分割使用することによって，内部演算と入出力装置の同時並行動作を可能にし



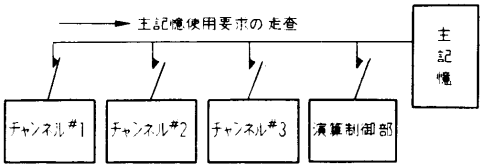
斜線部：主記憶とチャンネルとのデータ送受に使われるメモリサイクル，その他のメモリ・サイクルが内部演算に使われる．

第5図　チャンネルへのメモリ・サイクル割当て例

たものである．この方式では入出力装置はチャンネル（channel）を通して直接主記憶に結合されていて，そのチャンネルに与えられた主記憶サイクルを使用してデータの送受を行なう．チャンネルと演算制御部へのメモリ・サイクルの割りあてにはいくつかの方法があるが，いずれにしてもチャンネルには1字あるいは1語の Single Buffer または Dual Buffer しか必要ない．特に最近では主記憶が高速になってきており，多数の入出力装置と内部演算の同時並行処理が可能になっている．筆者は Direct Buffering の可能性を追求し，その理論的[4]結果に基づいて，はじめてこれを実用化した電子計算機 NEAC-2202 および NEAC-2204[5] を作ったが，これらの電子計算機は1桁分の Buffer しかもちいずに7台の入出力装置と内部演算の同時並行処理を実現している．

この Direct Buffer 方式は数々の有用性をもっているので，最近の電子計算機の多くに採用されている．NEAC シリーズ2200のモデル200を例にとるならば 2 μs という主記憶速度を生かして3本のチャンネルを 6 μs 間隔で走査し（その中の1本はさらに 12 μs の2本のチャンネルに分けて使用することもできる），データ送受の要求があれば割当てられたメモリ・サイクルでこれを行なう（**第5図**参照）．演算制御部は入出力装置より優先度が低く，割当てられたメモリ・サイクルをそのチャンネルが使用しなかったときのみこれを使用することができる．この方式では各チャンネルへのメモリ・サイクルの割当て回数を変えることによって，チャンネルの速度を変えることができ，少数の高速チャンネルとしたり多数の低速チャンネルとしたり，あるいはそれらの組合わせをつくることができるが，このメモリ・サイクルの割当てに同期式と非同期式があり，第5図の例は同期式である．非同期式の例として H 800 のトラフィック制御があげられ，**第6図**のように常に優先度の高い方から request

を調べてメモリ・サイクルを割当てている．両者の違いは同期式では各チャンネルの最大データ移送速度は一定であるが，非同期式では他のチャンネルの動作状態によってそれが変わってくる点にある．



第6図　トラフィック制御

Direct Buffer 方式は従来は多く大形機に用いられていたが，最近では中形機以下でもこの方式を採用しているものが多い．Direct Buffer 方式の特徴は次にのべるようなものである．

（1） Buffer Memory としての高価な高速記憶が不要になる．

（2） バッファの大きさによって入出力データ量が規制をうけることがなく，可変長記録を行なう磁気テープ装置などに対してはバッファ方式より融通性に富んでいる．

（3） 主記憶の速度に比例して多数のチャンネルをとることができるので，主記憶の高速化を生かすのに最適である．

（4） メモリ・サイクルの割当てを制御したり，演算を中断するなどの機能が中央処理装置に必要である．

（5）主記憶をバッファとして使用することにより，入出力装置と中央処理装置の結合が密になるので，入出力装置と入出力制御との機能，責任をできるだけ分けようとするときむずかしく，特に他社の装置と結合する場合に機能および責任の分担，保守などでむずかしい点が多い．

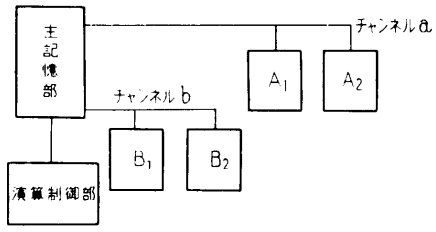
Direct Buffer 方式を用いたときには入出力命令はデータの送受を開始させるだけの意味しかなく，この命令に対するデータ送受が行なわれている間に中央処理装置は先の命令へ進んでしまうので，入出力動作の終了を何らかの方法でプログラムに知らせてやらなければならない．この機能は特に多重プログラムを実行するときに不可欠の機能で，命令でチェックする方法とハードウェア的にプログラムへの割込みを起す方法があるが，最近の電子計算機ではほとんど後者の方法がとられているようである．

3. チャンネルの方式

入出力装置と主記憶を結ぶデータの Path をチャンネルと呼んでいるが，これにもいくつかの方式があり，(1)主記憶と入出力装置の接続方式と(2)チャンネルの共用方式の二つの面を中心としてこれを述べてみよう．

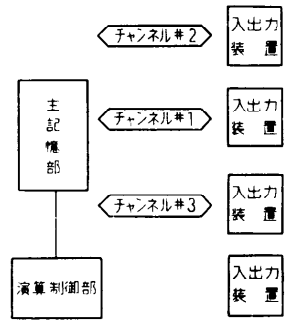
3.1 主記憶と入出力装置の接続方式

主記憶と入出力装置の接続がハードウェアで固定されているかプログラムで変えることができるかによって，固定チャンネル（Fixed Channel）と浮動チャンネル（Floating Channel）に分けることができる．固定チャンネルの場合は入出力装置 A_1 はチャンネル a を使用するということが定まっており，プログラムによって制御することができないのでハードウェアは簡単になるが，次のような不便を生ずる．**第7図**の例ではチャンネル a に接続されている入出力装置 A_1 が動作中はチャンネル b が空いていても，これを使って入出力装置 A_2 を動作させることができない．したがって N 本のチャンネルがあっても N 本のチャンネルが同時に動作することは一般には期待できない．IBM SYSTEM/360 の Selector Channel などはこの例である．



第7図　固定チャンネル

この非能率さを改善するために考えられた方式が浮動チャンネルで，主記憶とチャンネルと入出力装置を別個のものとして独立させ，その接続をプログラムで制御するものである（**第8図**）．これによってプログ



第8図　浮動チャンネル

ラムの要求さえあればかならずチャンネル数だけの入出力装置を同時動作させることが可能である．浮動チャンネルを採用している代表的な電子計算機としてNEAC-シリーズ2200とBurroughs B5000/5500[6]があるが前者では入出力命令で使用する入出力装置とチャンネルを指定するようになっており，後者ではチャンネル指定はプログラムで行なわずにハードウェアが自動的に空いているチャンネルをさがすようになっている．

以上述べたように，チャンネルの使用効率から考えると浮動チャンネルが優れているが，固定チャンネルにした場合にはチャンネルの種類をいくつかもうけることによって低速な入出力装置のために高速なチャンネルが占有されることが防げるという利点がある．紙テープに対しても，磁気テープに対しても，同一のデータ移送速度をもつチャンネルを使用することは非能率的であるが，浮動チャンネルの場合はどの入出力装置に使用されるかわからないので，チャンネルを高速にしておかねばならない．NEAC シリーズ2200では，こうした不利をなくすために1本のチャンネルをその半分のデータ移送速度をもつ2本のチャンネルとして使用したり，複数個のチャンネルをまとめて1本の高速チャンネルとして使用することもプログラムで制御できるようにしている．

3.2 チャンネルの共用方式

従来のチャンネルは，一時には1台の入出力装置に対してのみサービスを行なうものであったが，紙テープのように高々 1kc 程度のデータ移送速度しか持たぬ入出力装置のために数 100 kc のデータ移送速度をもつ入出力チャンネルを使用するのは非常に能率が悪く，しかもこれを救うために低速のチャンネルを多数もうけるのはハードウェアが増えて好ましくない．そこで1本の高速チャンネルのハードウェアを多数の低速な入出力装置で時分割的に共用しようという方式が考えられた．これが Univac 490[7] の ESI (Externally Specified Index)モード，IBM SYSTEM/360 の Multiplexor Channel，あるいは NEAC シリーズ2200のチャンネルの Multiplexor Bus モードである．これらの方式では，データを格納あるいは読出すべき主記憶のアドレスあるいはその index を入出力装置から指定することによって1本のチャンネルのハードウェアを多数の入出力装置が共有するようになっている．通常のチャンネルではデータを格納するアドレスは中央処理装置の内部のカウンタあるいは Control Word など，そのチャンネルに固有の機構によって指定されるので，複数個の入出力装置とデータの送受を行なうことはプログラムによらざるを得なかったが，外部から指定することによってこれがハードウェア的に可能になっている．Univac 490 の ESI モードの動作は入力のときを例にとると次のようになる．

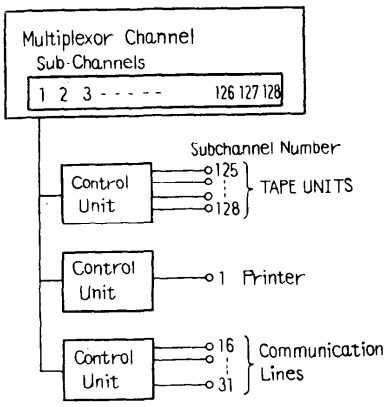
（1） 入出力装置から Index が送られてくる．

（2） Index で指定された主記憶を読み出す．ここには次に送られてくるデータの格納番地が入っている．

（3） このアドレスをアドレス・セレクタにセットすると共に Increment（あるいは Decrement）したアドレスを元の主記憶に格納する．

（4） 次に送られてきたデータをアドレス・セレクタの内容に従って格納する．

したがって ESI モードでは各入出力装置に異なった Index を与えておくことによって，それぞれ異なる主記憶番地へデータを送ることができる．Univac 490 には Externally Specified Address Mode もあるが，これは入出力装置が Index を送るのではなく，直接アドレスを送る方式である．SYSTEM 360 の Multiplexor Channel[8),9)] は Model 40 を例にとれば，128本の Sub-Channel に分かれており，その各 Sub-Channel に1台の入出力装置が接続できる（**第9図**参照）．



第9図　Multiplexor Channel の構成

NEAC-シリーズ2200の Multiplexor Bus モードは入出力装置からアドレスを送ってやり，そのアドレスに従ってデータを格納する方式である．この方式では前述した ESI モードと異なり，アドレスが直接データを格納あるいは読み出すアドレスとなっている．NEAC-シリーズ2200では，特に Multiplexor Chan-

nel というものは設けず，どのチャンネルも必要に応じて Multiplexor Bus モードで動作できるようになっている．

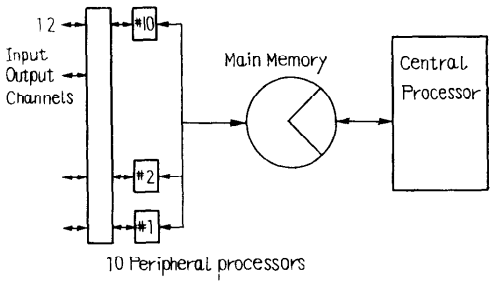
このようにチャンネルを共用する方式は低速の入出力装置のために高速チャンネルを専有されない点に特長があり，通信制御装置などで有用な方式である．すなわち通信回線からのデータ移送速度は非常に低いものであるから数10から数100の通信回線が1本のチャンネルを共有して主記憶とのデータ送受を行なうことが可能であり，しかも各通信回線ごとに異なった Buffer Area を主記憶内にとることができるので融通性があります．

しかしながら外部から回線番号あるいはアドレスを指定するというように，中央処理装置の論理構成と非常に密接に結びついた方式では，前にも述べたように他社の入出力装置を接続する際にその差異を補正するアダプタが複雑になることがある．例えば通信制御装置を統一しようとするような場合，そのアダプタが高価になって必ずしも有利といえない．むしろ大部分の伝送上のブロックより大きいバッファを通信制御装置において Multiplexor Bus モードを使用しない方式をとることによりバッファが大きくなることはあってもアダプタが単純になりシステムとして経済的になると考えられ，機能や責任の分担も明確となり，保守も容易になる．

3.3 チャンネル・プログラム

このように入出力制御部が本体と独立してその機能を果たすようになると，これを一つの計算機と考え，これにもプログラムを与えようという考え方ができる．このプログラムはチャンネルと主記憶のデータ送受を制御し，従来 Control Word ともよばれていたものであって gather write や scatter read を行なうことによってチャンネルのもつ機能をたかめている．System 360[10] は数種の Command によってプログラムを与える方式をとっているし，CDC-3600[11] では Data Channel が Control Word をもっていて House-Keeping Module が必要な House Keeping を行なっている．D 825[12] ではこうした機能を I/O Control Module がもっている．この考え方をさらにすすめて I/O Control のみを Peripheral Processor としたのが CDC-6600[13] で 4,096 語のメモリを持つ10台の Peripheral Processor が主記憶と入出力装置の間のデータ送受を制御している．

チャンネル・プログラムを与える方式はチャンネル動作の融通性を大きくするが，(1)プログラムが主記憶に入る場合にはこれを読み出すメモリ・サイクルが必要で，チャンネルのデータ移送速度が可能な最高速度より減ずる，(2)ハードウェアコストが高くつくなどの点から Cost/Performance がかならずしも大きくなるとは限らず，小形～中形計算機ではほとんど用いられない．



第10図

3.4 Interface について

初期の計算機には入出力装置の制御部をも含んだものがあったが，主記憶とのデータ送受が独立した入出力制御部によって行なわれるようになると，各入出力装置に特有な制御機能は各制御装置にもたせ，チャンネルとの接続条件（これを Interface とよんでいる）をすべての入出力装置について同一になるように制御部がつくられるようになった．これには次のような利点がある．

(1) チャンネルに接続する装置を指定しなくてもよくなり，システム構成の融通性があります．

(2) 最近各社から発表されているシリーズ・コンピュータでは各モデルの中央処理装置のチャンネルが同一 Interface をもつようにすることにより，任意の入出力装置を任意の中央処理装置に接続することができ，切換装置を通して1台の入出力装置を異なったモデルの中央処理装置が共有することが可能になる．また一つの入出力装置には一つの制御部をつくれば済み，多種類の入出力装置の整備が容易である．

3.5 Interruption について

入出力命令を出しても，実際のデータ送受はその後プログラムとは独立に進行させ，プログラムは先の命令を実行していく方式では，入出力動作の終了をなんらかの方法で知らなければ次の入出力命令を出すことができない．特に多重プログラム処理を行なう際には，入出力動作の終了を契機としてプログラムの切換えを行なうのでその重要性が大きい．初期の電子計算

機では入出力動作の終了によって Indicator がセットされ，命令でこれをテストしてやる方式が用いられたが，そのテスト命令をプログラムの各所にバラまかなければならなかった．

最近の電子計算機では入出力動作の終了によって自動的に一定番地に制御が移るプログラム割込方式 (Program Interruption) が用いられ，各プログラムはいつ入出力動作が終了したかに常に気を配っている必要がなくなった．さらにこの割込に優先度をもたせたり，ある入出力装置からの割込みを禁止したりする機能もあって，多数の入出力装置を能率よくプログラム制御できるようになっている電子計算機が大部分である．入出力装置からの割込原因も単にデータ送受の終了のみならず誤りの発生などもあり，より完全に入出力装置の状態をプログラムに知らせる傾向にある．

4. 電子計算機システムにおける入出力処理方式

4.1 入出力制御のソフトウェア

以上のべてきたように，入出力制御には種々の方式があり，しかもハードウェア技術のたゆまない進歩によって，それがますます高性能になると共に複雑さも増している．従来もプログラマは入出力処理ルーチンのために多大の努力をはらっていたが，新しい電子計算機の高性能で複雑な入出力機能を100%活用したプログラムを作ることは次第に困難になりつつある．とくに入出力動作のタイミングのとり方については，一つのプログラムを走らせるときはまだしも，多重プログラム処理を行なう場合には，他のプログラムのことまで考慮することは非常に困難であるからシステムの高能率な利用ができない．こうしたプログラマへの負担を減らし，システムの高能率な運用をはかるには入出力処理を1個所で集中して制御することが望ましく[16]，この目的のためのソフトウェアが開発された．これが IOCS (Input Output Control System)とよばれるものである．

（a） IOCS の機能

IOCS の持っている機能は，たとえば IBM 1410/7010[14] では次のようである．

（1） データ・レコードを読む．

（2） データ・レコードを書く．

（3） エラー状態を検出し是正する．

（4） データ・レコードの block および un-block を行なう．

（5） 読取/書込と演算の Overlap を行なう．

（6） テープ・ラベルの読取，チェック，書込みを行なう．

（7） ユーザの書いたプログラムへの出口を与える．

（8） 通信回線よりの割込を検出してこれを処理する Supervisor へ制御を渡す．

（9） 多数のテープの rewind その他を自動的に行なう．

（10） 同時動作している入出力装置からの割込みを検出する．

他の電子計算機の IOCS においても上述したような機能が標準的であろうと考えられる．IOCS[15] は大別して二つに分けられ，一つは要求された IOCS ルーチンを generate する MACRO Generator を使用するもので他は Object Program をロードするときに必要な IOCS ルーチンをロードするものである．前者ではプログラマは IOCS ルーチン，データ・ファイル，入出力エリヤなどの定義により注意を払わねばならないし，後者では IOCS ルーチンへの連結を指定する必要が度々ある．前者の分類に入る IOCS の代表的なものは IBM 1401 あるいは 1410/7010 の IOCS であり，後者の代表的なものは IBM 7090/94 の IOCS である．IOCS はさらに入出力装置の同時処理能力によって (1)同時処理をしないもの，(2)同時並行処理はするが割込み処理能力はないもの，(3)同時並行処理も割込処理能力もあるもの，に分けることができる．IOCS を使用した場合にはプログラマから見たらいずれも大差ないが，IOCS としては割込機能があるときは入出力動作の終了を常に監視している必要がない．同時並行処理を行なうときにはチャンネルのスケジューリングも IOCS が行なって高い効率で使用できるよう工夫している．

（b） IOCS の利点

IOCS を使用する場合の利点は次のようにまとめることができよう．

（1） 内部処理と入出力動作の Overlap を考慮しなくても IOCS が処理してくれるのでプログラム作成が容易になる．

（2） 入出力処理が標準化される．これは特に大きなプログラムを多数のプログラマが分担して作成する場合に有利である．

（3） オペレータとの Communication が標準化されるのでオペレータが各プログラムの内容をくわしく知っていなくてもよい．

（4）新しい電子計算機にプログラムをかける場合，

あるいは同じ電子計算機で構成が変わった場合に元のプログラムの修正が容易，あるいはパラメータを変えるだけで無修正で使用できる．

（5） 融通性が高くなる．たとえばプログラムをかける時点で使用したい入出力装置の一つが故障という場合でも，入出力装置の指定をプログラム実行時に変えることによってこのプログラムを実行することができる．

（6） File が統一的に管理されその Maintenance が容易になり，また Fileの Protection なども行なえる．

4.2 システム構成

入出力処理方式にはすべてを中央の計算機でやってしまう方式と衛星計算機を使用する方式がある．衛星計算機を使用する方式とは大形計算機で出力データあるいは入力データの編集を行なってこれを低速の入出力装置と送受するのは非能率的であるとし，大形計算機での処理結果は磁気テープなとに生のまま出力し，これを小形の電子計算機で編集して出力してやり，逆に入力のときは小形計算機で1度編集し，大形計算機がそのまま使える状態にして磁気テープなどに出力し，これを大形計算機が読みこむという方法である．これは磁気テープなどを介したオフライン衛星計算機の例であるが，大形計算機と小形計算機を直結するオンライン衛星計算機方式もある．IBM 7070 や 7090 に対する IBM 1401 は前者の例，H 800 に対する H 200. CDC 3600 に対する CDC 160 A などはオンライン衛星計算機の例である．NEAC シリーズ 2200 や IBM System 360 は各種モデルの中央処理装置同志をオンラインで接続することができるが，オンライン衛星計算機をもちいることによって中央大形計算機の負荷を分担できる反面，システム全体の Monitoring の複雑化などの問題もあり，Cost/Performance の評価は Case by Case に行なわれる必要があろう．

このような多重処理システム（Multi-Processing System）では，いわばひとつの中央処理装置が他の中央処理装置を周辺装置と見ることになり，同様な入出力制御が使用される．さらに GE 635 のように内部記憶が中心となっていはば演算制御部も他の入出力装置と同じく周辺装置の一つと見るような計算機もつくられるようになってきている．

5. 結 言

ここでは計算機の入出力制御方式につき概観してみたが，電子計算機では入出力装置が重要な要素であり，電子計算機システムにおける周辺装置台数の増大と種類の多様化（最近では大容量コアメモリまでが周辺装置として使用されはじめている），内部演算速度の向上による多重プログラム処理の必要度の増大，多重処理の実用化，リアルタイム処理の高度化などにともなって，入出力制御の重要性はますます大きくなってきており，今後とも計算機の方式設計，論理設計の主要なテーマの一つとして取上げられるものと考えられる．

参考文献

1) Werner Buchholz: The System Design of the IBM Type 701 Computer, Proc. of IRE Oct 1953.
2) IBM Reference Manual; 705 Data Porcessing System
3) IBM Reference Manual; 1401 Data Processing System.
4) 石井：時分割演算による計算機に関する研究，1961年2月
5) 金田，石井，斎藤，山田：NEAC-2204 電子計算機システムについて，電子計算機専門委員会資料，1961年3月
6) The Descriptor: A definition of the B 5000 information processing System. (Manual)
7) Univac 490 Real Time System SERIES Ⅲ: (Manual)
8) A. Padegs: The Structure of System 360 Part Ⅳ Channel Design Consideration, IBM System Journal Vol. 3 No. 2 1964.
9) IBM System/360 Engineering; Proc. of FJCC 1964.
10) IBM System Reference Library: IBM System/360 Principles of Operation.
11) Charles T. Casde: Planning the 3600, Proc. FJCC 1962.
12) James P. Anderson, Samuel A. Hoffman, Joseph Shifman, and Robert J. Williams: D 825 A Multiple Computer System for Command & Contorl, Proc. FJCC. 1962.
13) James E. Thornton: Parallel Operation in the Control Data 6600, Proc. of FJCC. 1964.
14) IBM Systems Reference Library: IBM 1410/7010 Operating System, Basic Input/Output Control System.
15) F. Peter Fisher and George F. Swindle: Computer Programming Systems, HOLT, RINHART AND WINSTON. INC, 1964.
16) William Orchard-Hays: The Evolution of Programming Systems, Proc. of IRE Jan. 1961.

（昭和40年9月13日受付）