《資　料》

# ミニコンのエディタの多ユーザ化*

岡　敬二**　木本晴夫***　細見輝政***

**Abstract**

We present a programming technique for making an interactive program which serves for many users at a time. The feature of this technique is to transform the already developed program into a re-entrant program to serve for many users. So in this technique we do not make use of the time sharing technique.

By using this technique, we developed an editor program which can be used by up to three users at a time. The developed program runs on a mini-computer PDP-11/20.

## 1. まえがき

ミニコンにおいても多くのユーザに同時にサービスできるようなシステム・プログラムが作成されているが[1),2)]，そのとき全く最初から作成するよりも，既に作成されている1ユーザを対象としたシステムプログラムのかなりの部分をそのまま利用して，少し手を加えるだけで作成が行なえるならば，プログラム作成上能率的である．

本稿では，エディタのようなインタアクティブ(interactive)なプログラムを，そのような方針で多くのユーザが同時に利用できるようにすることを目的とし，その作成を行なったことを報告する．

作成されたマルチ・ユーザ用のプログラムはエディタ・マルチ・プログラムと呼ばれ，PDP-11/20 ミニコンピュータの上で動く．具体的にはもとの1ユーザを対象としたプログラムのかなりの部分をユーザ数が3ユーザに制限されたものではあるが，再入可能ルーチン化し，あるユーザの処理途中 I/O 待ちなどが生じたときには他のユーザに制御を移し処理させるようにしている．I/O についてはすべて割込みで処理し，I/O のための制御の占有を避けて処理時間の短縮をはかっている．エディタのようなインタアクティブなプログラムではキーボートでの I/O 待ちが多くの時間を占めることから，このような方法のみで十分で，時分割によるプログラムの切換えの必要はなかった．

利用した1ユーザを対象としたエディタは，PDP-11/20 用として，DEC 社から供給されている EDIT-11 エディタで，PDP-11/20 の DOS (DISK OPERATING SYSTEM) のもとで動く．この DOS も1ユーザ専用にサービスするもので，多くのユーザに同時にサービスできないが，今回は同時に複数のユーザが使用できるようには改造せず，シリアリ・リユーザブル (serially reusable) ルーチンとしている．その理由は，DOS のサービスはファイルの取り扱いに関するもので，エディタを用いる作業の最初と最後にのみ使用され，本体のエディタの使用と比較して使用ひん度が僅少で，再入可能にしておく必要性があまりないからである．DOS が多くのユーザにサービスできるように，ユーザから DOS の使用要求があるごとに，そのユーザのユーザ・コードを DOS に知らせて，ユーザがサービスを受けられるようにしている．

## 2. PDP-11/20 計算機について

PDP-11/20 計算機は1語 16 ビットのミニコンピュータで，8個の汎用レジスタ (R0～R7) を持ち，そのうち R6 はスタック・ポインタ (SP) に当てられ，コア上の領域を示し，システム・スタックとして用いられる．また R7 はプログラム・カウンタ (PC) に割り当てられている．割込み処理は，割込みが生じると，PC の内容とそれまでの CPU 実行状況を示すプロセッサ・ステータス・ワード (PSW) の内容であるプロセッサ・ステータス (PS) が，自動的に SP を介してスタックされ，PC と PSW に新たに内容が，インタラプト・ベクトル (interrupt vector) と呼ばれ

* Converting an Editor for a Single User into for Multi Users, by Keiji OKA (Dentsu Advertizing Ltd.), Haruo KIMOTO and Terumasa HOSOMI (Department of Information And Computer Science, Faculty of Engineering Science, Osaka University).
** (株)電通
*** 大阪大学基礎工学部情報工学科

る2語からなるメモリ・ロケーションからロードされる．インタプラト・ベクトルの最初の語には，割込みサービス・ルーチンの実行開始番地，2番目の語には新しい PS が入っており，以後これらにより割込みサービス・ルーチンに移る．インタラプト・ベクトルの内容はプログラムで書換え可能で，書き換えておくことにより，割込みが起こったとき，あらかじめ作成しておいた割込みサービス・ルーチンへ制御を移すことができる．インタラプト・ベクトルの番地はインタラプトを起こすデバイスにより別々に定まっている．インタラプト・サービスが終了したときは，もとのプログラムの流れに制御を戻すために RTI (Return From Interrupt) 命令を実行させる．これによって SP の示す記憶領域上のスタック領域の先頭から2語が自動的に PC, PSW に戻される．

PDP-11/20 計算機には，DEC 社から DOS が提供されており，これは1ユーザを対象としてサービスを行なうものである．DOS はその DOS モニタ内のメモリ・ロケーションにユーザのユーザ・アイデンティフィケーション・コード (UIC) を格納する場所を持ち，この UIC によってサービスするユーザを識別する．今回はこの DOS を改造することなく，そのままの形を利用して多くのユーザにサービスを行なわせるようにしている．

## 3. 多ユーザ用のエディタの機構

以下では，まずエディタ・マルチ・プログラムがいくつかの主な機能的な部分から成ることを述べ，次にそれらの部分がどのように関係して，全体としてエディタ・マルチの作業を為すかを述べ，最後に，エディタ・マルチ・プログラムの動作の中で最も中心となる各ユーザ間の制御の受け渡しの詳細について述べる．

### 3.1 エディタ・マルチ・プログラムの構成

エディタ・マルチ・プログラムは，主に4つの機能的な部分から構成されている．最初は，ディストリビュータと名付けられた部分で，各ユーザに制御を配分する機能を持つ．

2番目は，エディタ本来の機能を遂行する部分で，制御を有するユーザはこの部分で，自己のための処理を行なう．ここでエディタ本体は，多くのユーザにサービスするために再入可能ルーチンとなっているが，使用できるユーザの数は3ユーザに制限されている．またその中の一部分は再入可能ではなく，シリアリ・リユーザブルの形としている．

3番目の部分は，各ユーザごとの固有のエリアで，主にユーザのワーク・エリア，スタック・エリアとインタラプト・ベクトルから成る．各ユーザはそのワーク・エリアを指定するのにインデックス・モードで指定し，インデックス・レジスタの内容を切り換えることにより，他のユーザのワーク・エリアが指定される．このインデックス・レジスタを，以後ベース・レジスタと呼ぶ．また処理の簡単のため，各ユーザは個別にスタック・エリアを持っている．

最後の4番目の部分は I/O 割込みに関する処理をするもので，入出力ドライバである．

### 3.2 エディタ・マルチ・プログラムの基本的動作

図1に，エディタ・マルチ・プログラムの概略を示す．プログラムの動作が開始されると，まずディストリビュータへ制御が移る．ディストリビュータは定められた手続きに従って制御を渡すべきユーザを決定し，しかるべきユーザに制御を渡す．制御を受け取ったユーザは，入力が完了されていてある処理が開始できる状態にあるならば，再入可能なエディタ本体の部分に入り，自己のための処理を遂行することになる．そうでなければ制御をディストリビュータに返す．処理の遂行の途中で I/O 待ちが生じたり，シリアリ・リユーザブル・ルーチンが使用不可で待ちとされる場合があったとする．このような場合，自らの待ちの状

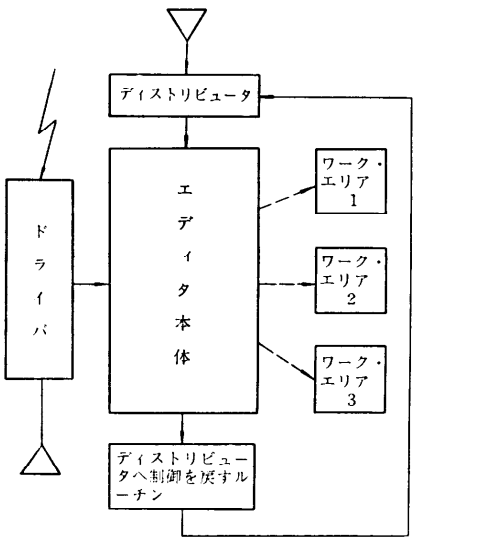


図 1 エディタ・マルチ・プログラムの概略

Fig. 1 General Flow of the Program

態であることをワーク・エリアに記憶して，他のユーザに制御を渡すためにいったん，ディストリビュータに制御を返す．ディストリビュータは再び，その定められた手続きに従って，次に制御を渡すべきユーザを決定し，制御を渡す．制御の渡ったユーザは，前のユーザと同じようにして自己の処理を遂行してゆく．待ちの状態となり，一時制御を離したユーザに再びディストリビュータから制御が渡ってきて，そのときまだ待ちの状態であるときには，制御をディストリビュータに戻す．あるいは既に I/O 処理などを終了している場合には，ユーザは制御を離さず処理を続けてゆくことになる．ある処理を遂行し終えたときは，ディストリビュータに制御を返し，次に処理が開始できる状態となるのを待つ．

割込みに関しては，あるユーザが制御を持っていて，自分自身も含めて他から I/O 割込みがあった場合には，割込みをかけたユーザに強制的に制御を渡し，ドライバにおいて割込みに関する処理を終えたのちもとのユーザに制御を返すか，そのまま制御を持ち処理を遂行し処理完了後もとのユーザに制御を返すかする．

### 3.3 制御の受け渡し

制御の受け渡しの際，切り換えられるものは，ベース・レジスタ，スタック・ポインタ，そして汎用レジスタである．

ユーザに制御が渡っている状態は2つあって，ユーザがエディタ本体を利用して処理をしている状態を内部処理状態，ユーザが入出力ドライバを用いて I/O 割込みを処理している状態を I/O 処理状態とする．I/O 処理状態とすべきところを，内部処理状態で代用でき，ユーザにサービスするには内部処理状態によることのみで済ませることができるが，特に I/O 処理状態を設けた理由は I/O 割込み処理はその回数が多く，しかも処理内容はエディタ本体での処理に比較して簡単で，したがって制御の移行もスタック・ポインタ，汎用レジスタの切換えが不要で簡単に済ませることができるからというものである．以下に2つの状態を詳述する．

内部処理状態とは，ベース・レジスタの内容がサービスを受けているユーザの固有の値となっており，かつスタック・ポインタの内容，汎用レジスタの内容もそのユーザがエディタ内部での処理をするために，そのユーザ固有のものとなり，ユーザが自分自身のスタックの内容を処理のために参照，変更することができるようになっている状態をいう．

I/O 処理状態とは，ベース・レジスタの内容がサービスを受けるユーザ固有の値となっているが，スタック・ポインタ，汎用レジスタはそのユーザ固有のものに切り換えられてはいない状態をいう．もちろん，このとき汎用レジスタを自由に使用できるようにするため，以前の汎用レジスタの内容は，前に制御が渡っていたユーザのスタックに貯えてしまっておく．スタックを切り換えていないことから，ユーザは他のユーザのスタック・エリアを使用するが，図2に示すようにこの，他のユーザが後でも必要とするエリアの内容は破壊せず，それより他のエリア部を使用している．図3では内部処理状態間の遷移において，スタックを切り換えたときの各ユーザのスタック・エリアの使用状況の一例を示す．

次に，ユーザからユーザへの制御の受け渡しの動作について述べる．制御の受け渡しには3種類の方法がある．

a. 制御の配分を行なうディストリビュータから制御を受ける．

b. 自ら制御を離し，ディストリビュータに渡す．

c. 割込みによる制御の受け渡し．

a, b は内部処理状態間での制御の受け渡しに用いられ，c は I/O 処理状態のための受け渡しの方法である．

a. ディストリビュータによる制御の配分

ディストリビュータは制御を要求しているユーザ，すなわち処理中に出力待ちとなり，いったんディストリビュータに制御を渡したが出力が完了して処理をさらに進めようとしているか，または同じく処理中で

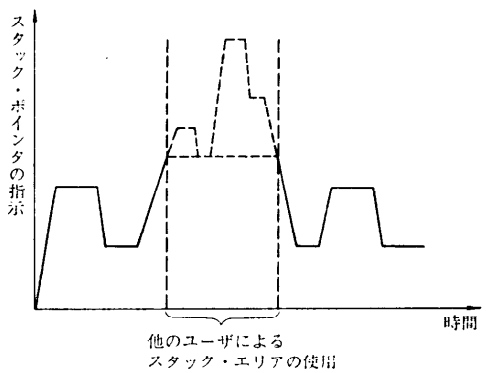


図 2 他のユーザによるユーザのスタック・エリアの使用

Fig. 2 Use of a user's stack area by other user

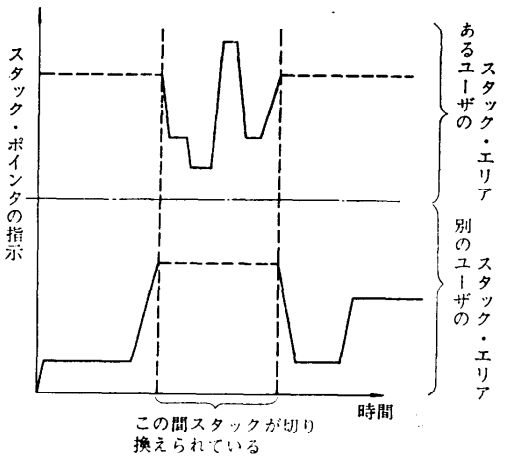


図 3　スタック・ポインタの切換え
Fig. 3　Changing of the content of stack pointer

DOS ルーチンを使用しようとするとき使用待ちとされて，その後そのルーチンが使用可となって処理続行可能となっているようなユーザに制御を渡す．制御を要求しているユーザが全くなければ，制御はディストリビュータに保持されたままとなる．このような機能を果たすために図4のようなテーブルを作り，図中で FLAG 1 はユーザ1がエディタ・マルチの使用にエントリしているかどうかを "0"，"1" で示し，CONM 1は，同じくユーザ1が入力待ちかまたは制御を要求している状態（"W" で示す）か，I/O など処理中の状態（"E" で示す）かのどちらかを示す．FLAG 2, 3, CONM 2, 3 についても同様である．ディストリビュータは，これらのフラッグを見て制御を要求しているユーザに順に制御を渡す．

ディストリビュータによる制御の配分の手続きを，図5のフロー・チャートに示す．図5でコントロールを持っていた以前のユーザは，ある処理が完了されて入力待ち，もしくは出力，DOS の使用などの待ちの

| | フラッグ | 内　容 |
|---|---|---|
| ユーザ 1 | FLAG 1 | {0, 1} |
| | CONM 1 | {W, E} |
| ユーザ 2 | FLAG 2 | {0, 1} |
| | CONM 2 | {W, E} |
| ユーザ 3 | FLAG 3 | {0, 1} |
| | CONM 3 | {W, E} |

図 4　ユーザの状況を示すテーブル
Fig. 4　Table for Users' Status

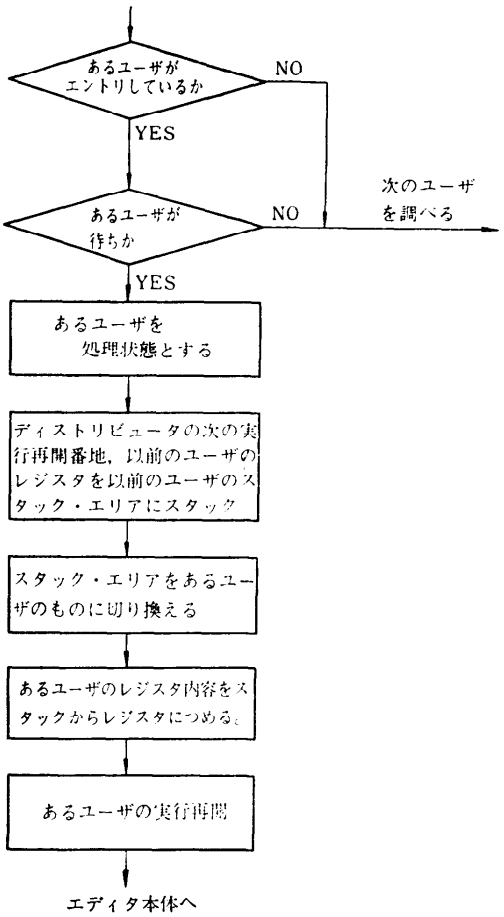


図 5　ディストリビュータによるあるユーザへの制御の配分
Fig. 5　Distribution of Control to a User by the "Distributor"

状態となっている．

b.　自ら制御を離しディストリビュータに制御を渡す方法

ユーザがある処理を完了して入力待ち，または処理中で，I/O 待ち，DOS ルーチンの使用待ちの場合，他のユーザに制御を渡す必要があるが，これはディストリビュータを介して行なわれる．この手続きを図6に示す．図7は，自ら制御を離して待ちの状態となっているユーザのスタック・エリアの先頭部の状況を示している．

c.　割込みによる制御の受け渡し

入力・出力時，入出力割込みによって他のユーザから強制的に制御を奪い，割込みについての処理が終わ

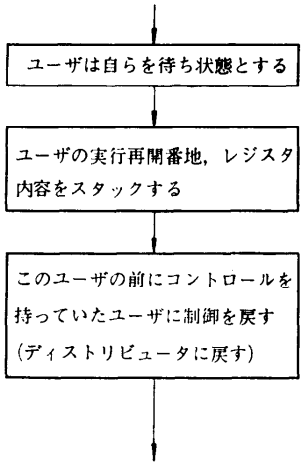


図 6 自ら制御を離しディストリビュータに渡す方法
Fig. 6 Giving the Control to the "Distributor"

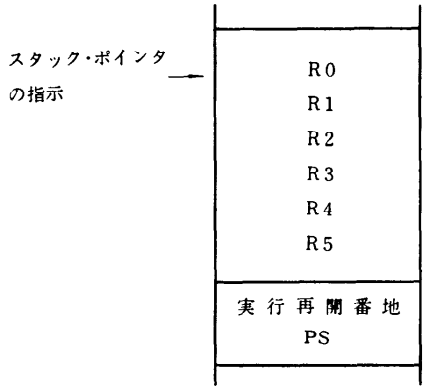


図 7 待ちのユーザのスタックの先頭部
Fig. 7 Top of the stack area of waiting user

ると，制御を奪ったもとのユーザに直接制御を戻すようにしている．この手続きを図8に示す．

この I/O 割込みによって制御を強制的に奪い，直接もとのユーザに戻す方法は，キーボード I/O を行なうときにしか用いず，その場合，割込み処理ルーチンではスタックを利用しても，その処理終了後スタックは完全に使用開始前の状態に戻される．ゆえにこの場合，制御が移ってもスタックの切換えを行なう必要はない．なお図8中の割込み処理ルーチンとは，ドライバ中の共通ルーチンのことで，ここで述べていることは次のドライバの節と関係が深い．

## 4. 多ユーザ用エディタのためのドライバ

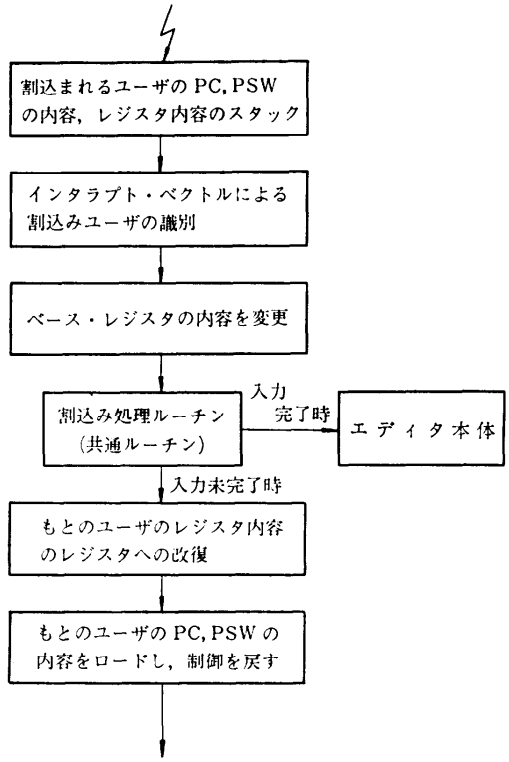


図 8 割込みによる制御の受け渡し
Fig. 8 Handling of Control at the Occurence of Interrupt

エディタ・マルチ・プログラムのドライバの機能は，1ユーザを対象としたドライバの機能に加えて，どのユーザからの割込みかを判断し，そのユーザに対して I/O サービスを開始することである．入出力ドライバの構成としては，まず各々のユーザのための処理ルーチンを別々に作り，各々のユーザがその処理ルーチンを通り抜けた後，各ユーザに共通した処理を行なう共通ルーチンへ制御が移るようにしている．これらの関係を図9に示す．各々のユーザのための処理ルーチンでは，ベース・レジスタの内容の変更などユーザに制御を渡す作業を行ない，共通ルーチンでは入力の受け入れ，出力動作を行なう．ドライバの作業が終わると，制御を奪われたもとのユーザに制御を返すようにしている．

### 4.1 各々のユーザのための処理ルーチン

ユーザから割込みがあると，PDP-11/20 計算機の割込み処理機構により，ユーザ固有の処理ルーチンへ制御を移し，ユーザへのサービス開始，すなわち以前

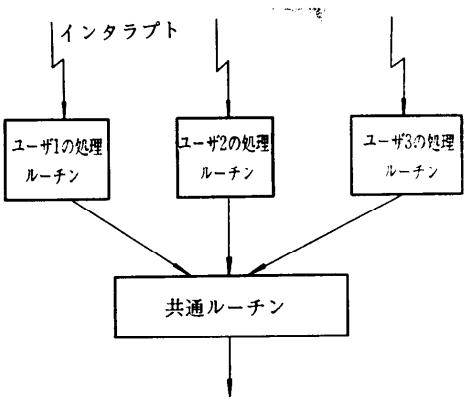


図 9 各ユーザの処理ルーチンと共通ルーチンの関係
Fig. 9 Relation between routines for each user and roouine for all users in driver section.

のレジスタ(R 0～R 7)のプッシュ・ダウンとベース・レジスタの内容を新しいユーザ固有の値に変えることを行ない，その後共通ルーチンへ入る．

### 4.2 共通ルーチン

a. 出力ドライバ

出力用バッファにたくわえられた出力文字系列を出力する．出力ドライバで出力中か，出力を完了してしまっているかを識別するために，フラッグを設けている．このフラッグを見ることにより，出力ドライバから次のルーチンに入るか，また出力ドライバに入ってよいかどうかを決定する．

b. 入力ドライバ

割込みによる入力を，出力用バッファ，内部処理用バッファに同時につめる．入力がバッファにつめられると出力ドライバが使用中かどうかを見て，出力ドライバに制御を移し，そのエコーを行なう．エコーが完了してしまってから，入力の1行分の終りを示す CR (Carriage Return) 入力が入力されているかどうかを調べ，入力されているならばエディタ本体に制御を移し，内部処理用バッファにたくわえられている入力の処理を行なう．

## 5. その他プログラム作成のために行なった処理

この節では，今までに述べたこと以外に行なったことの主なものについて述べる．まずユーザ識別に用いるベース・レジスタに汎用レジスタ (R 0～R 7) の中のどれを当てるかを決定することであるが，既存のプログラムで R 0～R 5 の中で最も使用ひん度の小さいものを選ぶことにした．この使用ひん度はクロス・レファレンス・テーブルで参照した．そしてベース・レジスタと決定されたものは，もはや汎用レジスタとして用いられてはならず，それが使用されている箇所はあるワーク・エリアを設けて代替えした．

また全体のワーク・エリアとしては，あるユーザが必要とするワーク・エリアを集めてひとまとまりとし，それと全く同じ形をしたものを他のユーザに対しても持たせている．アドレス方式をインデックス・モードとし，ワーク・エリアのユーザごとの切換えをインデックス・レジスタの切換えによって行なっている．

## 6. むすび

われわれは全く新たにマルチユーザ用のプログラムを作るのではなく，既存のプログラムに適切な改造を施すことによって，そのプログラムを同時に複数のユーザにサービスすることを試みた．改造を施すについては，対象プログラムの大体のフローを知っておく必要があった．EDIT-11 エディタをこのように改造することにより，今まで CPU 使用効率の非常に低かった，オン・ライン・エデット作業での CPU 使用効率向上に成功した．

今回の作業で用いた改造の方法は，対象がインタアクティブなプログラムであれば，マルチユーザ用のプログラムに改造する手段として有効である．ゆえに，この方式でさまざまな既存のインタアクティブなプログラムを，多くのユーザが同時に使えるようにレベル・アップすることが望ましい．

### 参考文献

1) 加藤道子，“TOSBAC-40 マルチ・ユーザ・サービス”，情報処理学会大会 '72 予稿集．
2) 葛山，本田，“あるミニコン用 TSS の試作について”，情報処理学会大会 '73 予稿集．
3) PDP-11 Peripherals and Interfacing Handbook : Digital Equipment Corporation, 1971.
4) PDP-11 Processor Handbook : Digital Equipment Corporation, 1971.
5) PDP-11 Edit-11 Text Editor : Digital Equipment Corporation, 1971.
6) PDP-11 Disk Operating System Monitor : Digital Equipment Corporation, 1971.
7) PDP-11 Macro-11 Assembler : Digital Equipment Corporation, 1972.

(昭和 49 年 3 月 9 日受付)
(昭和 49 年 6 月 12 日再受付)