

スーパーコンピュータ VPP5000 利用の手引

2002 年 4 月版

～ 目次 ～

1 . スーパーコンピュータ VPP5000 の特徴.....	1
2 . VPP5000 のソフトウェア.....	1
3 . 利用形態とジョブ種別	2
4 . 翻訳コマンド.....	3
5 . ライブラリの利用.....	4
6 . バッチジョブの投入	4
6.1 バッチリクエスト（スクリプトファイル）の作成例と実行依頼	4
6.1.1 1PE 用のモジュールの実行（メモリ 2GB 以内）	4
6.1.2 1PE 用のモジュールの実行（メモリ 2GB を超え 15.5GB 以内）	5
6.1.3 並列処理用モジュールの実行.....	5
6.2 スーパーコンピュータでのジョブ投入	5
6.3 汎用計算サーバでのジョブ投入.....	6
6.4 NQS のコマンド	7
6.4.1 qsub コマンド	7
6.4.2 qstat コマンド.....	7
6.4.3 qdel コマンド	8
6.4.4 qcat コマンド	8
6.5 Web ブラウザからのジョブ投入.....	9
7 . TSS での実行	11
7.1 TSS での実行環境.....	11
7.2 翻訳処理.....	12
7.2.1 Fortran の翻訳処理と逐次実行.....	12
7.2.2 VPP Fortran による並列処理の翻訳	12
7.3 TSS での並列実行.....	12

8 . 実行時間の計測方法	13
9 . OPEN 文によらないファイルの結合方法 < Fortran >	14
10 . 図形出力ライブラリの利用方法	14
11 . スーパーコンピュータのファイル利用について	14
11.1 高速大容量ファイルの利用について	14
11.2 ファイル入出力のエラー	15
11.3 dpfs_alloc コマンド	15
11.4 mrfs(memory resident file system)の利用	15
12 . MPI の利用について	16
12.1 翻訳	16
12.2 実行	16
12.2.1 full モードでの実行	16
12.2.2 limited モードでの実行	17
12.2.3 TSS での実行	17
12.2.4 実行時の環境変数	17
13 . 必要メモリ量の試算	18
14 . クロスコンパイル	19
15 . 大容量ファイルの退避	20

1 . スーパーコンピュータ VPP5000 の特徴

スーパーコンピュータ VPP5000 は、分散メモリ型のベクトル並列計算機です。VPP5000 の特徴を以下に列挙します。

- ・ PE 台数：64 (8 台は IOPE)
- ・ PE の理論最大性能 9.6Gflops
- ・ 総メモリ容量 1024GB
- ・ 1PE のメモリ容量：16GB
- ・ クロスバネットワークのデータ転送速度：最大 1.6GB / 秒
- ・ スカラ演算の性能向上
- ・ 4 倍精度演算の高速化

2 . VPP5000 のソフトウェア

VPP5000 で利用できるソフトウェアの一覧を表 1 に示します。

表 1：VPP5000 のソフトウェア一覧

種類	ソフトウェア名
プログラミング言語	Fortran (JIS X3001-1:1998 準拠)
	C (ANSI X3.159:1989 準拠)
	C++ (USL release 3.0 準拠)
並列処理言語	VPP Fortran
	HPF (High Performance Fortran Forum 仕様 V2.0 準拠)
	DPCE (Data Parallel C Extension)
数値計算ライブラリ	SSL (科学用サブルーチンライブラリ)
	C-SSL (科学用サブルーチンライブラリ)
	NUMPAC (数値計算ライブラリ)
	BLAS (ベクトル処理向き線形計算ライブラリ)
	LAPACK (ベクトル処理向き線形計算ライブラリ)
	ScaLAPAC (MPI 並列処理向き線形計算ライブラリ)
メッセージパッシングライブラリ	MPI (MPI2.0 仕様準拠)
	PVM (PVM3.3 仕様準拠)
アプリケーション	- FLOW (汎用 3 次元流体解析システム) *
	STAR-CD (非構造格子汎用熱流体解析ソフトウェア)
	POPLAS/FEM5 (有限要素法による構造解析プログラム)
	LS-DYNA3D (非線形動的構造解析ソフトウェア)
	MASPHYC-2, MASPHYC-SP (材料設計システム) *
	Gaussian98 (分子軌道計算プログラム) *
	AMBER (分子構造計算プログラム) *
	FastDNAm1 (最尤法による進化系統樹推定プログラム)
	VisLink (可視化ソフトウェア)
	MOLPRO (分子軌道計算プログラム)

*印のアプリケーションは、並列版が用意されている。

3．利用形態とジョブ種別

スーパーコンピュータ VPP5000 と汎用計算サーバおよびファイルサーバのシステム概略を図 1 に示します。

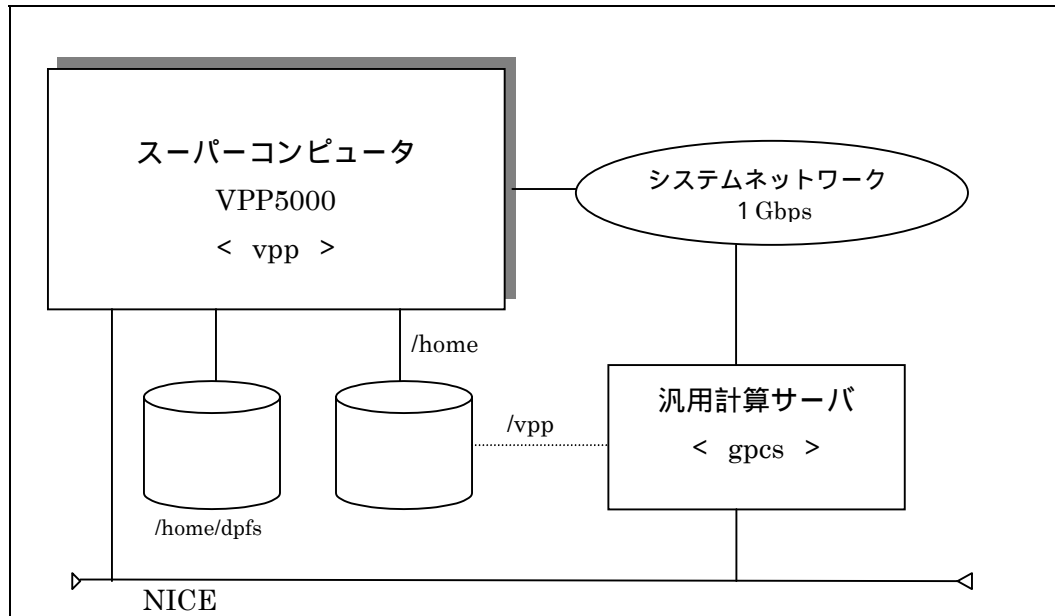


図 1：スーパーコンピュータ VPP5000 のシステム概略

図 1 に示すようにスーパーコンピュータは、ネットワークから直接 TELNET 接続して利用可能です。また、汎用計算サーバ gpcs から NQS により利用することも可能です。

スーパーコンピュータで利用できるファイルは、2 種類あります。一つは、スーパーコンピュータに login したときのホームディレクトリのファイルで、/home のファイル¹です。このファイルは、汎用計算サーバ gpcs からは、/vpp のディレクトリとして見えます。このファイルは、fpfs(Flexible and high performance file system)と呼ばれるファイルシステムで、小容量のファイルから大容量のファイルまでを高速に処理することができます。一方、/home/dpfs のディレクトリのファイルは、大容量のファイルや並列ジョブからの入出力に適したファイルシステムです。dpfs の利用については、11 章を参照してください。

スーパーコンピュータ VPP5000 の利用形態には、バッチジョブを投入して利用する方法と TSS で会話的に利用する方法があります。実行が長時間にわたるプログラムはバッチ処理がおすすめです。一方、短時間で終了するものや、翻訳処理やデバッグは TSS が便利です。

バッチジョブの投入経路としては、

1. スーパーコンピュータの TSS からコマンドによりバッチジョブを投入する。
2. 汎用計算サーバ gpcs から NQS のコマンドによりバッチジョブを投入する。
3. 本センターのホームページにアクセスし、Web ブラウザからバッチジョブを投入する。

¹ pwd コマンドを実行するとホームディレクトリが、/G/pe0M/usrN/login-name と表示される場合がありますが、ホームディレクトリを指定する場合には、/home/usrN/login-name と指定してください。

があります。どの経路を選択するかは、各自の利用環境にもよりますが、スーパーコンピュータだけを利用する場合には、1.の経路が、ポスト処理等で汎用計算サーバを利用する場合には、2.の経路が良いでしょう。3.の経路は、手元のパソコンから利用し、UNIX のファイル編集に馴染みの薄い利用者には便利な方法です。

スーパーコンピュータ VPP5000 のホスト名は、`vpp.cc.nagoya-u.ac.jp`²、汎用計算サーバのホスト名は、`gpcs.cc.nagoya-u.ac.jp` です。本センターのホームページの URL は、<http://www.cc.nagoya-u.ac.jp>です。

スーパーコンピュータで利用可能な PE 数、CPU 時間やメモリ量などの計算資源はジョブの種別により異なります。表 2 にジョブ種別を示します。

表 2：スーパーコンピュータ VPP5000 のジョブ種別

利用形態	キュー名	使用可能 PE 数	CPU 使用時間	メモリサイズ		経過時間	利用目的
				標準値	制限値		
バッチジョブ	c	1	60 分	500MB	2GB	-	非並列ジョブ
	x	1	1200 分	2GB	15.5GB	-	非並列ジョブ
	z	2 ~ 16	600 分	1PE あたり 2GB	15.5GB	-	並列ジョブ
	ze	17 ~ 32	600 分	1PE あたり 2GB	15.5GB	-	並列ジョブ
TSS	-	1	60 分	500MB	2GB	-	会話型非並列
	-	1 ~ 8	300 分	1PE あたり 2 GB	15.5GB*	720 分	会話型並列

*) 使用 PE 数が 5 以上の場合には、1 台の PE に 2 個の仮想プロセッサを割り当てるため、使用できるメモリの最大は、7.5GB の 1 / 2 になります。

4．翻訳コマンド

VPP で利用できる言語と翻訳コマンド³を表 3 に示します。なお、`frt` コマンドでは、Fortran95 の言語仕様が標準で動きます。

表 3：翻訳コマンド

言語	コマンド
Fortran/VP(ベクトル化)	<code>frt</code>
Fortran/VPP(並列化)	<code>frt -Wx</code>
C	<code>cc</code>
C/VP	<code>cc -Kvp</code> または <code>vcc</code>
C++/VP(ベクトル化)	<code>CC</code>
C++	<code>CC -Knovp</code>

翻訳コマンドのオプションは、VPP に login して、`man` コマンドで確認してください。

² `vpp.cc.nagoya-u.ac.jp` に TELNET 接続した場合、端末への表示は、"UXP/V TELNET (pe2c)"となります。

³ 汎用計算サーバ `gpcs` で VPP 用のクロスコンパイルができます。詳細は、14 章を参照してください。

5. ライブラリの利用

ライブラリを使用する場合には、`frt` コマンドの `-l` オプションの指定が必要です。使用するライブラリと指定するオプションを表 4 に示します。

表 4：ライブラリとオプション

ライブラリ名	オプション
NUMPAC：数値計算ライブラリ	<code>-l numpac</code>
SSL：科学用サブルーチンライブラリ	<code>-l ssl2vp</code>
SSL：科学用サブルーチンライブラリ並列版	<code>-l ssl2vpp</code>
SSL：科学用サブルーチンライブラリ HPF 版	<code>-l ssl2hpf</code>
ユーザ登録のセンターライブラリ	<code>-l ulib</code>
図形出力ライブラリ	<code>-l ps</code>
BLAS（ベクトル処理向き線形計算ライブラリ）	<code>-l blasvp</code>
LAPACK（ベクトル処理向き線形計算ライブラリ）	<code>-l lapackvp -l blasvp</code>

6. バッチジョブの投入

NQS のコマンドによるバッチジョブの投入手順を以下に示します。

1. 実行するコマンドをバッチリクエスト（スクリプトファイル）としてファイルに作成する。
2. `qsub` コマンドでバッチリクエストの実行依頼をする。
3. `qstat` コマンドでジョブの実行状況の確認をする。
4. ジョブの実行が終了していたら、出力結果のファイルをチェックする。

6.1 バッチリクエスト（スクリプトファイル）の作成例と実行依頼

以下の例では、スクリプトファイルは `nqs` のディレクトリに作成するものとする。なお、ここでは `qsub` のオプションはスクリプトファイルに記述する。`qsub` のオプションについては 6.4.1 を参照されたい。

6.1.1 1PE 用のモジュールの実行（メモリ 2GB 以内⁴）

実行キューとして `x` を指定する。

- 1) スクリプトファイル `nqs/exec_s.sh` の内容
実行可能ファイル名：`pg/vtest_s`

```
# @$-q x -eo -o vtest_s.out
cd pg
./vtest_s
```

2) 1PE 用モジュールの実行依頼

スクリプトファイル：`exec_s.sh`

`qsub` コマンドで依頼する。

```
vpp% qsub exec_s.sh
```

⁴ 実行時にメモリが不足した場合には、“Not enough space” のメッセージが出力される。

6.1.2 1PE 用のモジュールの実行 (メモリ 2GB を超え 15.5GB 以内)

実行キューとして `x` を指定し、`qsub` の `-lM` オプションで必要メモリ量を指定する。

- 1) スクリプトファイル `nqs/exec_s5g.sh`
実行可能ファイル名: `pg/vtest_s5g`

```
# @$-q x -eo -o vtest_s.out -lM 5gb
cd pg
./vtest_s5g
```

- 2) 1PE 用モジュールの実行依頼

スクリプトファイル: `exec_s5g.sh`

`qsub` コマンドで依頼する。

```
vpp% qsub exec_s5g.sh
```

6.1.3 並列処理用モジュールの実行

並列化処理を行う場合には、実行キューとして `z` (16PE まで利用可能) または `ze` (32PE まで利用可能) を指定する。PE 数の指定は、`-lP` オプションで行う。使用するメモリが 2GB を超える場合には、`-lM` オプションで必要メモリ量を指定する。以下の例では、実行の終了をメール⁵ で通知するように `-me` オプションを指定している。また、`-lT` オプションに使用する CPU 時間 (この例では 1 時間 30 分) を指定している。

- 3-1) スクリプトファイル `nqs/exec_p5.sh` の内容

実行可能ファイル名: `pg/vtest_p5`

```
# @$-q z -lP 5 -eo -o vtest_p5.out
# @$-me -lT 1:30:00 -lM 5gb
cd pg
./vtest_p5
```

- 3-2) 並列用モジュールの実行依頼

スクリプトファイル: `exec_p5.sh`

`qsub` コマンドで依頼する。

```
vpp% qsub exec_p5.sh
```

6.2 スーパーコンピュータでのジョブ投入

スーパーコンピュータ VPP5000 ではバッチジョブの投入に関して、次に示すコマンド⁶が利用できます。

バッチリクエストの実行依頼

`qsub` コマンド

バッチリクエストの実行状態の確認

`qstat` コマンド

⁵ `qsub` を実行しているホスト以外のシステムにメールで通知したい場合には、`qsub` を実行しているホストの `forward` ファイルにメールアドレスを記述する。

⁶ これらのコマンドは、`/usr/local/bin` にあります。これらのコマンドの動きがおかしい場合には、`which qsub` としてコマンドのパス名を確認してください。

バッチリクエストの削除

qdel コマンド

実行中のバッチリクエストの詳細情報の表示

jstat コマンド

上記コマンドのうち qsub、qstat、qdel コマンドの使用方法和オプションは、NQS のコマンドと同じです。コマンドのオプションの詳細については、man コマンドで確認してください。

実行中のジョブの状況を見るために jstat コマンドが用意されています。jstat コマンドでは、実行中のバッチリクエストの経過時間、CPU 時間、VU 時間、メモリ量、I/O 量が確認できます。なお、並列処理では、表示される CPU 時間と VU 時間はいずれも最大のものであります。

(使用例)

```
vpp% qsub exec_p.sh
Request 291.pe00 submitted to queue: x.
vpp% qstat x
x@pe00; type=BATCH; [ENABLED, RUNNING]; pri=31
  0 exit;  1 run;  0 queued;  0 wait;  0 hold;  0 arrive;
  REQUEST NAME  REQUEST ID  USER  PRI  STATE  JOB-ID  PHASE
  1:      exec      291.pe00  a49999a  31  RUNNING   137  RUN
vpp% jstat
Job Information on vpp5000 (ver 1.00)      2000/04/19-15:27
-----
QUEUE USER REQUEST-No  PE  ELAPS      CPU      VU      %      MEM  I/O
-----
x  a49999a   291.pe00   1  0:55:25  0:50:18  0:43:08  88  384MB 84KB
-----
vpp% qdel -k 291.pe00
Request 291.pe00 is running, and has been signalled.
vpp% qstat x
x@pe00; type=BATCH; [ENABLED, INACTIVE]; pri=31
```

【留意点】TSS から qsub コマンドでジョブの実行依頼をすると、標準出力ファイルに下記のメッセージが出力されますが、実行には影響ありませんので、無視してください。

```
Warning: no access to tty; thus no job control in this shell...
```

6.3 汎用計算サーバでのジョブ投入

スーパーコンピュータで使用するファイルはすべて /vpp/home/usrN/login-name のディレクトリの下に作成してください。gps の cdvpp コマンドにより /vpp のディレクトリに移動できます。

バッチリクエストの実行依頼

qsub コマンド

バッチリクエストの実行状態の確認

qstat コマンド

バッチリクエストの削除

qdel コマンド

汎用計算サーバ gps から qstat、qdel コマンドを実行する場合にはシステム名も付けて指定してください。

【例】 `gpcs% qstat x@vpp-g`

キューxの状態表示

`gpcs% qdel -k -r vpp-g` リクエスト ID

バッチリクエストの削除

6.4 NQS のコマンド

6.4.1 qsub コマンド

機能：バッチリクエストの実行依頼をする。

形式： `qsub` [オプション] [スクリプトファイル]

qsub コマンドの主なオプション：

- q バッチリクエストを依頼するキュー名を指定する。キュー名については表2を参照のこと。
- e 標準エラー出力ファイルを指定する。
- o 標準出力ファイルを指定する。
- eo 標準エラー出力を標準出力ファイルに出力する。-eを指定した場合にはこのオプションは指定できない。
- lP 並列処理の場合に使用するPE数を指定する。
- lT 使用するCPU時間を指定する。
時間の記述形式は、[[時間:]分:]秒[.ミリ秒]である。
- lM 使用するメモリ量を指定する。記述の形式は、整数[.小数]gbである。
- mb バッチリクエストの実行を開始したときにメールを送る。
- me バッチリクエストの実行を終了したときにメールを送る。
- mu バッチリクエストに関するメールを指定ユーザに送る。
- s バッチリクエストシェルスクリプトを解釈するのに用いるシェルのフルパスを指定する。デフォルトではCシェルが用いられる。Bシェルを用いるときは次のように指定する。

`-s /usr/bin/sh`

使用例：

`qsub -q キュー名 スクリプトファイル名`

なお、スクリプトファイル内には `qsub` コマンドのオプションも記述することができます。`qsub` コマンドのオプションの詳細は、`man` コマンドで確認してください。

6.4.2 qstat コマンド

機能：バッチリクエストの実行状態を確認する。

形式： `qstat` [キュー名]

オプションの説明：

汎用計算サーバ `gpcs` からキューの状態を表示する場合はシステム名も付けて以下の形式で指定する。

`x@vpp-g` キューxの状態を表示

`z@vpp-g` キューzの状態を表示

使用例：

qstat x@vpp-g

6.4.3 qdel コマンド

機能：バッチリクエストの削除

形式：qdel [-k|-シグナル番号] [-r システム名] リクエスト ID

オプションの説明：

-k | -シグナル番号

-k 指定したバッチリクエストのうち走行中のバッチリクエストに対してシグナル SIGKILL を送信する。

-シグナル番号 指定したバッチリクエストのうち走行中のバッチリクエストに対してシグナル SIGKILL 以外のシグナルを送信する。

このオプションを指定しない場合は走行待ちのバッチリクエストを削除する。

-r システム名

汎用計算サーバ gpcs から行う場合は、システム名として vpp-g を指定する。

使用例：

```
qdel -k -r vpp-g リクエスト ID
```

【注意】バッチリクエストが swap out されている場合には、qdel コマンドが効かない旨のメッセージが表示されるが、そのバッチリクエストが swap in された時点で、qdel コマンドが有効となり、バッチリクエストは削除される。

【qstat コマンドと qdel コマンドの使用例】

```
gpcs% qstat x@vpp-g
x@pe00; type=BATCH; [ENABLED, RUNNING]; pri=31
  0 exit;   1 run;   0 queued;   0 wait;   0 hold;   0 arrive;
    REQUESTNAME    REQUESTID    USERPRI    STATE    JOB-ID    PHASE
      1:  exec_s.sh  70494.gpcs    a49999a  31  RUNNING      173  RUN
gpcsf% qdel -k -r vpp-g 70494.gpcs
Request 70494.gpcs is running, and has been signalled.
```

6.4.4 qcat コマンド

機能：バッチリクエストのスク립トファイル、出力ファイルの表示

形式：qcat [-e|-o|-s] [-u ユーザ名] リクエスト ID

オプションの説明：

-e リクエストの標準エラー出力ファイルを表示。

-o リクエストの標準出力ファイルを表示。

-s リクエストのスク립トファイルを表示。

-e、-o、-s オプションはいずれか一つのみが指定可能。これらのオプションを省略した場合には、-o が指定されたものとみなされる。

6.5 Web ブラウザからのジョブ投入

本センターのホームページからスーパーコンピュータへジョブを投入することができます。このウィンドウでは、ジョブ投入の他にジョブの実行状況の表示、ファイルの編集、ファイルの内容表示、ファイルの複写・移動、ファイルの消去、ブラウザを起動しているパソコンとの間でのファイルのアップロード/ダウンロードなどもできます。パソコンをホームにして仕事をしている利用者には便利な機能です。ジョブの投入手順を次に示します。

- 1) 「システム利用案内」の「スーパーコンピュータ VPP5000/64 利用案内」の項目をクリックする。
- 2) 「Web ブラウザからのジョブ投入」をクリックする。このページを表示するには Plug-in が必要である。Plug-in が必要な場合は、該当するアイコンをクリックして Plug-in のソフトを入手する。なお、汎用計算サーバ gpcs で netscape を立ち上げる場合には、実行に先立ち mkplugin コマンドを実行すると Plug-in の環境が整う。
- 3) 次に示すウィンドウが表示されるので、“vpp”の項目をチェックし、“User ID”、“Password”の項目を入力する。更に“Language”を“Japanese”にして、“OK”をクリックする。

Group Name	Status	User ID	Password
<input type="checkbox"/> gpcs	NO CONNECTION		
<input checked="" type="checkbox"/> vpp	NO CONNECTION		

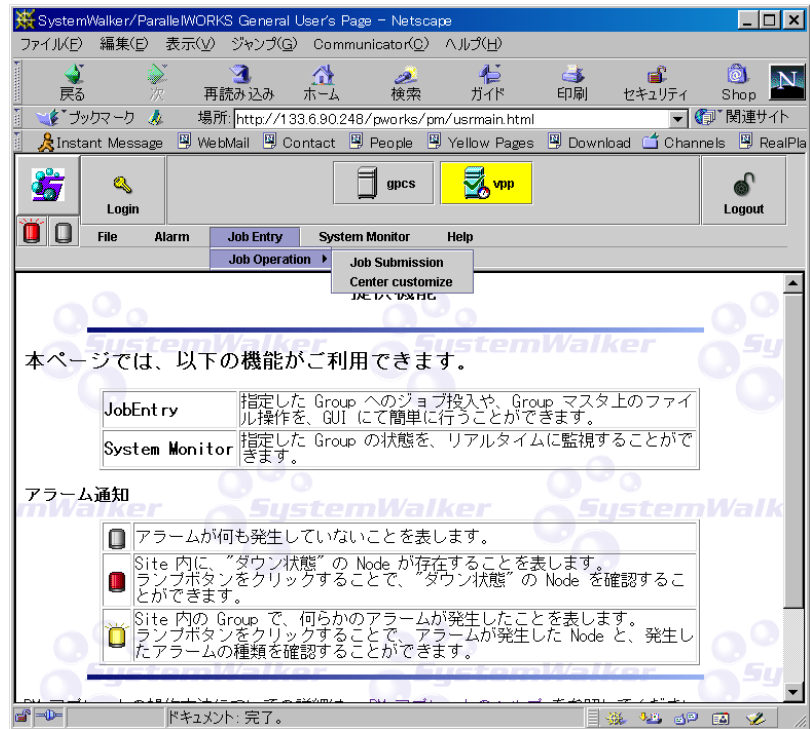
Help
OK
Cancel

Options
☐ Use one User ID for all
☐ Use one Password for all

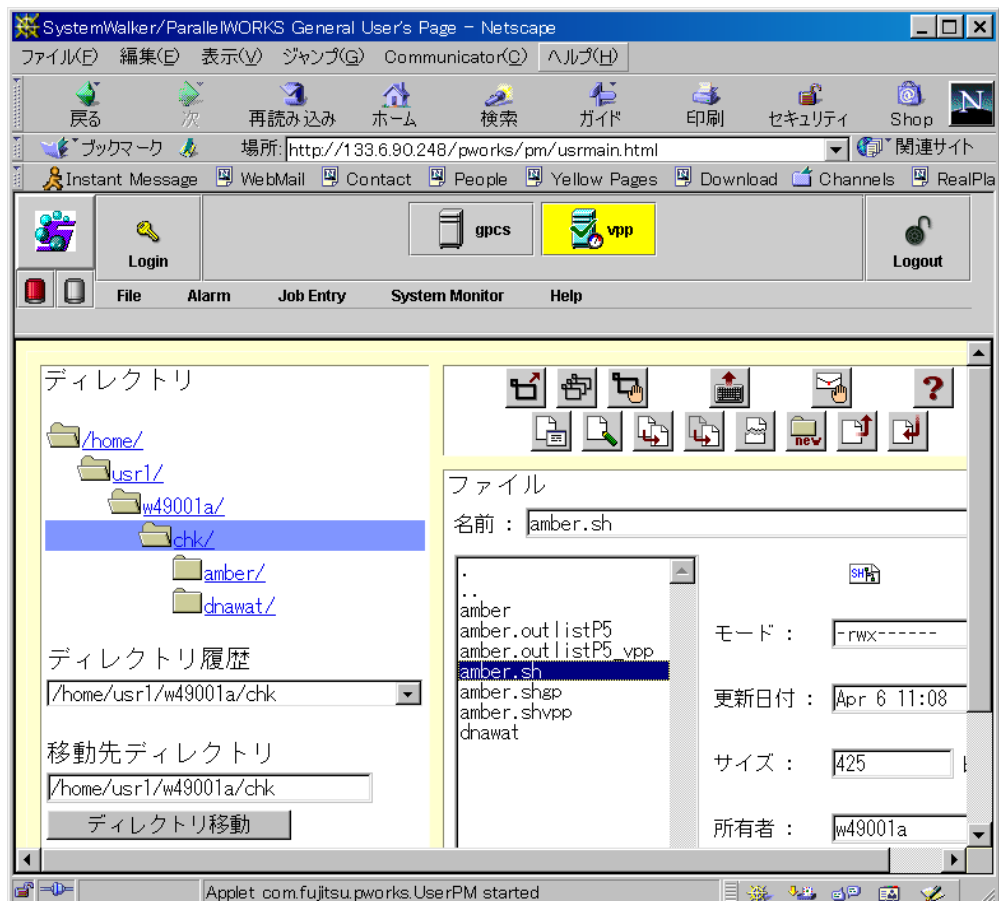
Language
Japanese

Warning: Applet Window

- 4) 次に示すウィンドウが表示されるので、“Job Entry”の“Job Submission”の項目をクリックする。“Job Entry”の項目がクリックできないときは、その上の“vpp”のアイコンをクリックする。



- 5) 次に示すウィンドウが表示される。このウィンドウで、ファイルを選択し、ウィンドウ右上の”Submit job”のアイコンを選択することにより、スーパーコンピュータにジョブを投入することができる。このとき、指定したファイル名のサフィックスが .sh の場合には、qsub コマンドが実行され、ジョブの投入が完了する。ファイル名のサフィックスが .sh 以外のときは、指定したファイルはロードモジュール(実行可能ファイル)と判断され、実行依頼を行うためのシェルスクリプトファイルを作成するウィンドウが表示されるので、必要な項目を入力する。



- 6) “Operate own job”のアイコンで、投入したジョブの状況を表示する。
- 7) ジョブが終了したら、“Browse file”のアイコンで結果を確認する。結果のファイルをブラウザを起動しているパソコンにダウンロードする場合には、“Transfer file to browser computer”のアイコンでファイルを転送する。
- 8) すべての処理が終了したら、“logout”のアイコンをクリックする。

7. TSS での実行

7.1 TSS での実行環境

VPP5000 で利用できるエディタ⁷には以下のものがあります。

コマンド名	エディタ名	備考
emacs	Emacs-20.4	日本語入力システムは SKK。
ng	miniEMACS	日本語表示可能。 日本語の入力は、パソコンまたは CDE の日本語入力システムを使用してください。
vi	vi	

また、less、nkf などのコマンドも利用できます。

⁷ VPP5000 の/home のファイルは、gps から/vpp のファイルとしてみえますので、それらのファイルは、汎用計算サーバ gps で編集することも可能です。

TSS では、CPU 使用時間は 60 分、メモリは標準で 500MB まで、指定をすれば 2 GB まで使用できます。現在のシステム資源は、limit コマンドで表示できます。これらの設定を変更する場合にも limit コマンドで行います。limit コマンドの使用例を以下に示します。

(例) limit コマンドによりメモリを 2 GB まで使用可能にする。

```
vpp% limit
cputime      1:00:00
filesize     2097152 kbytes
datasize     2097152 kbytes
stacksize    16384 kbytes
coredumpsize 16384 kbytes
memoryuse    524288 kbytes
descriptors  512
vpp% limit memoryuse 2097152kb
vpp% limit
cputime      1:00:00
filesize     2097152 kbytes
datasize     2097152 kbytes
stacksize    16384 kbytes
coredumpsize 16384 kbytes
memoryuse    2097152kb kbytes
descriptors  512
```

7.2 翻訳処理

VPP5000 では TSS で翻訳・実行ができます。翻訳コマンドについては“4. 翻訳コマンド”を参照ください。

7.2.1 Fortran の翻訳処理と逐次実行

Fortran のソースのファイル名 : prog.f

実行可能ファイル名 : prog

```
vpp% frt -o prog prog.f
vpp% ./prog
```

7.2.2 VPP Fortran による並列処理の翻訳

VPP Fortran で並列処理を行うためには、コンパイラオプション -Wx を指定します。

Fortran のソースのファイル名 : prog_p.f

実行可能ファイル名 : prog_p

```
vpp% frt -Wx -o prog_p prog_p.f
```

7.3 TSS での並列実行

VPP5000 の TSS では、並列実行ができます。TSS の並列処理で利用できる PE は 4 台ですが、折り返し機能により 8 並列まで実行可能です⁸。デバッグやインタラクティブに結果を表示したい場合には便利です。課金の対象となる CPU は、バッチ型処理と同じで使

用した CPU の演算時間のうち最大のものです。演算負担経費は、会話型処理と同じで、1 秒につき 2 円です。TSS の並列処理での利用可能なシステム資源については “ 3 . 利用形態とジョブ種別 ” の項を参照してください。

TSS での並列実行は、`jobexec` コマンドで行います。`jobexec` コマンドの主なオプションを以下に挙げておきます。オプションの詳細は、`man` コマンドで確認してください。

- vp 使用する PE 数を指定する。
- ct CPU 使用時間を指定する。省略値は、300 分。
値は、[[時 :] 分 :] 秒の形式で指定する。
- et 経過時間を指定する。省略値は、720 分。
値は、[[時 :] 分 :] 秒の形式で指定する。
- mem 使用するメモリサイズを指定する。省略値は 2048MB。
値はメガバイト単位で指定する。

なお、`jobexec` コマンドのオペランドで指定する スクリプトファイル には、実行権 が必要です。また、このスクリプトファイルにはホームからのパス名を指定してください。

以下に使用例を示します。

【使用例】4 台の PE を使用し、メモリサイズを 3GB に、CPU 使用時間を 30 分に、経過時間を 60 分に指定して、スクリプトファイル(ファイル名: `$HOME/ex/para_tss`) に指定されたジョブを実行する。

```
vpp% jobexec -vp 4 -mem 3072 -ct 30: -et 60: ex/para_tss
```

8 . 実行時間の計測方法

実行に要した CPU 時間は、`timex` コマンドにより知ることができます。VU (ベクトルユニット) の使用時間も表示されます。

```
cd pg
timex ./vtest_s
```

`timex` コマンドの出力例を次に示します。ここで、`real` は経過時間を示しています。実行に要した CPU 時間は、`user` と `sys` の項目を合わせたものです。時間はいずれも秒単位です。並列処理の場合には、`user` と `sys` の項目は、すべての PE の合計値となります。並列実行の場合 `-P` オプション⁹を指定すると各 PE での CPU 時間が表示されます。この `timex` コマンドの出力は標準エラー出力に出力されます。

real	6:31.22
user	6:23.52
sys	2.11
vu-user	6:23.24
vu-sys	0.00

⁸ 現在の運用では、1 台の PE に 2 個の仮想プロセッサを割り当てることも可能になっているため。

⁹ Gaussian98 で並列処理を行う場合には `-P` オプションを指定すると出力が多くなるので、`-P` オプションの指定は行わない方がよい。

9 . OPEN 文によらないファイルの結合方法 < Fortran >

Fortran のプログラムで装置番号とファイルを結合する場合、OPEN 文ではなく環境変数によりファイルの対応づけを行なうことができます。環境変数によるファイル名の指定は、次のように行います。

setenv fuxx ファイル名

ここで、fu : 固定、xx : 装置番号(00 ~ 2147483647)

装置番号とファイルの結合例を次に示します。

```
setenv fu01 testin.dat
setenv fu02 testout.dat
./a.out
```

この例では、装置番号 1 に対して testin.dat を、装置番号 2 に対して testout.dat を割り当てています。なお、環境変数によるファイルの結合と OPEN 文の両方が指定されている場合には、OPEN 文が優先されます。

10 . 図形出力ライブラリの利用方法

図形出力ライブラリを用いて PS ファイルを作成する場合には、firt コマンドの -l オプションで ps を指定してください。使用例を次に示します。

```
cd pg
firt -o xyout xyout.f -l ps
```

なお、PS 用図形出力ライブラリでは装置番号 99 に対して PS ファイルを出力しているので、プログラムの実行に際しては PS ファイルの出力先の指定をしてください。指定方法には、以下の 3 つがあります。

- ・ OPEN 文
- ・ 環境変数による 99 へのファイル割り当て
- ・ XINT サブルーチンの引数の file= で指定

なお、上記のいずれの指定もしなかった場合には、fort.99 という名前のファイルが作成されます。

11 . スーパーコンピュータのファイル利用について

11.1 高速大容量ファイルの利用について

スーパーコンピュータ VPP5000 には、高速処理できる大容量向きのファイルシステム dpfs(distributed parallel file system)が用意されています。これは大容量のファイルや並列ジョブからの入出力に適したファイルシステムです。

dpfs を利用する場合には、利用に先立ちディレクトリを作成する必要があります。ディレクトリの作成は、dpfsdir コマンドで行います。dpfsdir コマンドを実行すると /home/dpfs/usrN/login-name のディレクトリが作成されますので、これらのディレクトリの下にファイルを作成してください。dpfs のファイルの課金や保存期間は、通常のファ

イルと同様です。

また、dpfs では実行時に入出力作業領域を指定すると読み書きの処理が更に高速になります。入出力作業領域の指定方法を以下に示します。

【例】実行時に 10MB の入出力作業領域を指定する。

```
./a.out -W1, -g10000
```

11.2 ファイル入出力のエラー

ファイルの入出力関係でエラーが出た場合、エラー番号は表示されますが、エラーの内容までは表示されません。エラー番号の内容は、ferrno コマンドで表示できます。

<エラーメッセージ>

```
jwe0022i-sline10 入出力エラーが発生しました( systemcall=open, errno=2, unit=
1) .
  error occurs at MAIN__      line 9  loc  00000000000000830  offset
000000000000002b0 (2c,2c)
  MAIN__      at loc 00000000000000580 called from o.s.
jwe0903i-u エラー識別番号 0022 のエラーが発生し、エラー打切り回数に達しました .
error summary (Fortran)
error number  error level  error count
jwe0022i      s          1
total error count = 1
```

【ferrno コマンドの使用例】

```
vpp% ferrno 2 ▲
      #define ENOENT 2 /* No such file or directory */
```

これがエラーの内容

11.3 dpfs_alloc コマンド

実行に先立ちあらかじめ dpfs のファイルの領域を確保しておきたい場合に利用します。

形式 : dpfs_alloc -i 初期割当て量 -e 拡張割当て量 ファイル名

ここで、割当量は、いずれも 2 MB 単位のデータブロック数で指定します。1 以上 8388608 の値が指定できます。

11.4 mrfs(memory resident file system)の利用

VPP5000 では、mrfs と呼ばれるメモリ常駐ファイルシステムが他のファイルシステムと同じように利用できます。mrfs は、一時的にメモリ上にファイルを作成しますので、その入出力性能は数百 MB/sec と高速です。しかし、ジョブの終了時点で自動的に開放され削除されます。mrfs のファイルは、NQS の qsub コマンドの -cc または -lv オペランドの指定により利用できます。

1. qsub コマンドの -cc オペランドを指定することによりバッチリクエストの実行ディレクトリを mrfs 上とする。

【例】

```
vpp% qsub -cc go.sh
```

2. qsub コマンドの -lV オペランドで、使用する mrfs のファイルサイズを指定する。このファイルサイズは、-lM オペランドで指定されたメモリ量から確保される。-cc オペランドを指定しない場合は、環境変数 MRFSDIR によりファイルの入出力をおこなう。

【例】

```
# @$-q x -eo -o prog1.out -lV 1gb -lM 3gb
setenv fu01 prog1/testin.dat
cd $MRFSDIR
setenv fu10 temp.dat
$HOME/prog1/a.out
```

12 . MPI の利用について

12.1 翻訳

Fortran mpifrt

C mpicc

【例】 vpp% mpifrt -o mpi_prog mpi_prog.f

12.2 実行

vpp で MPI を実行する場合には、full モードと limited モードの 2 つの機能モードがある。

12.2.1 full モードでの実行

mpiexec コマンドで起動し、MPI2 規格のすべての言語仕様のプログラムを実行することが可能であるモードである。full モードでの実行に必要なプロセッサ数は、mpiexec の使用するプロセッサ分 (1) 増加したものになる。

12.2.1.1 full モードの実行方法

mpiexec コマンドにより行う。mpiexec コマンドの形式を次に示す。

```
mpiexec -n np prog args
```

ここで、

np : ユーザの実行可能プログラムが最初に静的に必要とするプロセッサ数

prog : 最初に静的に生成される実行可能プログラムのパス名

args : *prog* への引数

バッチジョブとして動作させるには、qsub コマンドにより実行依頼する。次に示すようなスクリプトファイルを作成し、qsub コマンドで実行する。

スクリプトファイル名 : mpi_full.sh

```
# @$-q z -eo -o mpi_full.out
# @$-lP 5
mpiexec -n 4 mpi_prog
```

→ キューの指定
→ PE 数の指定
→ プロセッサ数の指定

```
vpp% qsub mpi_full.sh
```

12.2.2 limited モードでの実行

limited モードでは、MPI2 規格の言語仕様のうち、プロセス管理および MPI I/O 機能が実行できない。

12.2.2.1 limited モードの実行方法

バッチジョブとして動作させるには、qsub コマンドにより実行依頼する。次に示すようなスクリプトファイルを作成し、qsub コマンドで実行する。

スクリプトファイル名: mpi_limited.sh

# @\$-q z -eo -o mpi_full.out	→	キューの指定
# @\$-lP 4	→	PE 数の指定
./mpi_prog -np 4	→	プロセッサ数の指定

```
vpp% qsub mpi_limited.sh
```

12.2.3 TSS での実行

vpp では、TSS で並列実行ができる。TSS の並列処理で利用できる PE は 4 台である。デバッグやインタラクティブ実行に便利である。TSS の並列実行は jobexec コマンドで行う。PE 数の指定は -vp オプションで、プロセッサ数の指定は -np オプションで行う。

```
vpp% jobexec -vp 4
jobexec start
cd MPI_pro
./mpi_prog -np 4
```

12.2.4 実行時の環境変数

環境変数一覧 『UXP/V MPI 使用手引書』 表 2-3 (p.27)

通常は、環境変数を意識する必要はないが、転送するメッセージが多くなると環境変数 VPP_MBX_SIZE で、メッセージプールの大きさ(バイト単位)を指定する必要がある。なお、省略値は、4,194,304 バイトである。

【例】 setenv VPP_MBX_SIZE 10485760

上記の環境変数でメッセージプールを大きく指定することにより回避できたエラー

(その1)

```
MPLIB-sr2.3.1 Exception Handler
DeadLock detected during parallel execution,
By process 119369939, running as MASTER on processing element 0
Time Stamp : Fri Dec 15 20:11:30 2000
Deadlock Diagnostic : Process is part of cycle.
:
```

(その2)

```
MPLIB-sr2.3.1 Signal Handler
Signal 6 (code 6 - errno 16) received during parallel execution,
By process 1180762371, running as NODE on processing element 2
Executable : ./mpi_prog
Time Stamp : Wed Dec 13 15:11:30 2000
Diagnostic : Abort
    Receive queue/mailbox overflow
    Interrupt vpid is 2
    :
```

1 3 . 必要メモリ量の試算

VPP5000 上で実行するプログラムの必要メモリ量は、次の手順で算出¹⁰することにより知ることができます。

(1) size コマンドにより data と bss のサイズを知る。

【例】

```
vpp% size vtest_p32
2547536 + 355248 + 819120208 = 822022992
  ↑       ↑       ↑       ↑
text    data    bss    total
```

(2) gsize コマンドにより PE 数と max_global サイズを知る。

【例】

```
vpp% gsize vtest_p32
File vtest_p32 global array information is following.
```

global array information :

total size	-	62812624KB (4) on	32 pe's
partitioned	-	62812624KB (4)	
non-partitioned	-	0KB (0)	
max. size / pe	-	1962894KB	

これが max_global
サイズ

(3) LOCAL のメモリ量を次式より求める。

$LOCAL = [data + bss + 1.5MB + 22KB * PE \text{ 数} + i/o \text{ buffer 域}] + 128MB$

ここで、[]内は 128MB に切り上げる。なお、i/o buffer 域の default 値は、ファイルシステムにより異なり、次のようになっている。

fpfs(/home のファイル)	512KB
dpfs(/home/dpfs のファイル)	2MB

¹⁰ ここでは、計算を簡単にするために 1 K=1000、1 M=10⁶ で計算していますが、正確には 1 K=1024、1 M=1024² で計算してください。

【例】

LOCAL = [355KB + 819MB + 1.5MB + 22KB*32 + 512KB] + 128MB
= [822MB] + 128MB
= 896MB + 128MB = 1024MB

(4) GLOBAL のメモリ量を次式より求める。

GLOBAL = [max_global + 256KB + 2KB*PE 数]

【例】

GLOBAL = [1962MB + 256KB + 64KB]
= 2048MB

(5) 必要メモリ量 = LOCAL + GLOBAL

【例】

必要メモリ量 = 1024MB + 2048MB = 3072MB

14 . クロスコンパイル

汎用計算サーバ gpcs で VPP 用のクロスコンパイルができます。クロスコンパイラのコマンド名と機能を表 5 に示します。

表 5：クロスコンパイラのコマンド名と機能

コマンド名	機 能
frtpx	Fortran/VP、Fortran/VPP、HPF のコンパイル
fccpx	C (VPP 上の C) のコンパイル
CCpx	C++ (VPP 上の C++) のコンパイル
mpifrtpx	MPI(Fortran 用)のコンパイル
mpiccpix	MPI(C 用)のコンパイル

これらのクロスコンパイラを利用するためには、以下の環境設定が必要です。

1) 環境設定ファイルの .cshrc の変更

通常的环境設定では、クロスコンパイラが動作する環境になっていません。そこで、ホームディレクトリの下での .cshrc のファイルの内容を以下のように変更してください。

```
source /etc/skel/local.cshrc    source /etc/skel/local.cshrc.all
```

2) VPP システム利用者情報の設定

gpcs の vppnetpx コマンドで利用者情報の設定を行います。以下に使用例を示します。

【使用例】

```
gpcs% vppnetpx
```

```
Enter VPP Host name      : vvp
```

```
Enter VPP Login name     : a49999a
```

```
Enter VPP Password       :
```

```
Enter VPP output directory : /home/usr9/a49999a/pg
```

オブジェクトと実行可能ファイルを格納する VPP 上のディレクトリを指定する。

利用者情報に変更がある場合には、再度 vppnetpx コマンドで変更してください。

- 3) frtpx コマンドでクロスコンパイルする場合には、環境変数 FORT90C の設定を削除¹¹してから行ってください。なお、VPP との Fortran の環境を合わせるためコンパイラ オプション -X9 (Fortran95 の言語仕様を採用) の指定をしてください。

(使用例)

```
gpcs% unsetenv FORT90C
gpcs% frtpx -X9 -o prog1 prog.f
```

クロスコンパイラの各コマンドのオプションは、man コマンドで確認してください。

15. 大容量ファイルの退避

計算結果のファイルが大量にある場合、その退避装置として、DVD の貸出しボリュームが利用できます。一旦ファイルを DVD へ転送する手間が要りますが、ディスクファイルに比較して 1 桁程度低額になっています。DVD の利用申請は、センター4 階事務受付 (052-789-4355) で行います。利用の概要は次のとおりです。

1. 利用負担金

- ・ 1 ボリューム 5GB とし、1 ボリュームにつき月額 200 円とする。
- ・ 利用申請は、5 ボリューム (25GB) 以上とする。
- ・ DVD の利用負担金額は、DVD 利用申請時および月初めに請求を行う。

2. DVD 貸出しボリュームシステムの利用方法

- ・ ホスト名

NICE (FastEther) 経由: dvdserv.cc.nagoya-u.ac.jp

システムネットワーク (GigaEther) 経由: dvdserv-g

- ・ ログイン名、パスワード: 他のシステム (vpp、gpcs、nucc 等) と同じ。
- ・ DVD 利用申請ごとに DVD ボリュームが作成される。ボリューム名は、登録番号 + 1 桁。

(例) 登録番号: a49999a ボリューム名: a49999a1

- ・ DVD ボリュームのマウントポイントは、\$HOME/ボリューム名。

(例) 登録番号 (login 名): a49999a ボリューム名: a49999a1

マウントポイント: /home/usr9/a49999a/a49999a1

- ・ DVD 貸出しボリュームシステムでは、2GB を超えるファイルは扱えないので、ファイルを圧縮したり、split コマンドでファイルを分割すること。
- ・ ホームディレクトリには、環境設定ファイル以外のファイルは作成しないこと。

3. 利用例

【例 1】chkdvd.dat のファイルを DVD ボリューム a49999a1 へ転送する。ホスト名は、dvdserv-g を指定する。

```
vpp% ftp dvdserv-g
```

¹¹ 環境変数 FORT90C の設定を戻すときは、再 login してください。

```
Connected to 192.168.1.19.
220 dvdserv FTP server (Version wu-2.6.0(1) Fri Apr 21 12:59:33
JST 2000) ready.
Name (dvdserv-g:a49999a):
331 Password required for a49999a.
Password:
230 User a49999a logged in.
ftp> cd a49999a1
250 CWD command successful.
257 "/home/usr9/a49999a/a49999a1" is current directory.
ftp> bin
200 Type set to I.
ftp> put dvdchk.dat
200 PORT command successful.
150 Opening BINARY mode data connection for dvdchk.dat.
226 Transfer complete.
local: dvdchk.dat remote: dvdchk.dat
367048480 bytes sent in 82 seconds (4369.69 Kbytes/s)
```

【例 2】2GB を超えるファイル (chk2g.dat) を DVD ボリュームへ転送する。

(その 1) 2GB を超えるファイルは、DVD ボリュームには転送できないので、split コマンドでファイルを分割する。ファイルの大きさを分割する場合には、-b オプションでその大きさを指定する。この例では、2000MB 単位に分割し、分割されるファイルの接頭辞を chk2g.datX と指定している。なお、この接頭辞のオペランドを省略すると、x となる。split コマンドのオペランドの詳細については、man コマンドで確認すること。

```
vpp% split -b 2000m chk2g.dat chk2g.datX
vpp% ls
chk2g.dat  chk2g.datXaa  chk2g.datXab  chk2g.datXac
```

(その 2) 分割されたファイルを DVD ボリュームに転送する。

```
vpp% ftp dvdserv
Connected to 133.6.90.19.
220 dvdserv FTP server (Version wu-2.6.0(1) Fri Apr 21 12:59:33
JST 2000) ready.
Name (dvdserv:a49999a):
331 Password required for a49999a.
Password:
230 User a49999a logged in.
ftp> cd a49999a1
```

```
250 CWD command successful.
257 "/home/usr9/a49999a/a49999a1" is current directory.
ftp> bin
200 Type set to I.
ftp> put chk2g.datXaa
200 PORT command successful.
150 Opening BINARY mode data connection for chk2g.datXaa.
226 Transfer complete.
local: chk2g.datXaa remote: chk2g.dataXaa
2097152000 bytes sent in 7.6e+02 seconds (2.7e+03 Kbytes/s)
:
:
```

DVD ボリュームにファイルを転送する。

(補足)

分割されたファイルを1つのファイルに連結するのは、cat コマンドで行う。

```
vpp% cat chk2g.datXaa chk2g.datXab chk2g.datXac >chk2g.dat
```


【参考資料】

< Fortran 関係 >

- (1) UXP/V Fortran 使用手引書 V20 用(J2U5-0410)
- (2) UXP/V Fortran/VPP 使用手引書 V20 用(J2U5-0430)
- (3) UXP/V Fortran メッセージ説明書 V20 用(J2U5-0460)
- (4) UXP/V HPF 使用手引書 V20 用(J2U5-0450)

< C 関係 >

- (5) UXP/V C 言語使用手引書 V20 用(J2U5-0121)
- (6) UXP/V DPCE 使用手引書 V20 用

< C++ >

- (7) UXP/V C++使用手引書 V20 用 (J2U5-0311)

< ライブラリ >

- (8) 富士通 SSL 使用手引書(99SP-4020)
- (9) FUJITSU SSL 拡張機能使用手引書(99SP-4070)
- (10) FUJITSU SSL 拡張機能使用手引書 (J2X0-1360)
- (11) FUJITSU SSL /VPP 使用手引書(J2X0-1372)
- (12) FUJITSU C-SSL 使用手引書(J2X0-3330)
- (13) UXP/V BLAS/VP LAPACK/VP ScaLAPACK 使用手引書 V20 用(J2U5-0480)

< Fortran 並列処理技法 >

- (14) UXP/V VPP Fortran プログラミングハンドブック V20 用
- (15) VPP Fortran 入門 (改定版) 名古屋大学大型計算機センターニュース
Vol.31, No.3, 2000.8
- (16) VPP Fortran から HPF へ, 名古屋大学大型計算機センターニュース
Vol.31, No.4, 2000.11

< ベクトル化技法 >

- (17) UXP/V Fortran プログラミングハンドブック V20 用

< プログラミング支援 >

- (18) UXP/V アナライザ使用手引書 V20 用

< メッセージパッシングライブラリ >

- (19) UXP/V MPI 使用手引書 V20 用(J2U5-0272)
- (20) UXP/V PVM 使用手引書 V20 用(J2U5-0141)
- (21) FUJITSU MPTools 使用手引書(J2X0-2641)

なお、上記参考資料のうち(15)、(16)以外は、名古屋大学大型計算機センターの利用者に限りホームページで参照することができます。

【VPP のデータ形式】

VPP の浮動小数点の形式は、IEEE 形式（ANSI/IEEE Std 754-1985 により規格化されているもの）と呼ばれるもので、多くのワークステーションで採用されているものと同じ形式である。単精度(REAL*4)の形式と倍精度(REAL*8)の形式を図 2 に示す。ここで、s は符号部、e は指数部、f は仮数部とする。



図 2：VPP の浮動小数点の形式

この形式で表される単精度の浮動小数点の値を表 6 に、倍精度の浮動小数点の値を表 7 に示す。表中で INF は *Infinity* を NaN は *Not a Number* を表す。

表 5：単精度の浮動小数点の値

指数部と仮数部の値	表現される浮動小数点の値
$0 < e < 255$	$(-1)^s 2^{e-127}(1.f)$
$e = 0, f \neq 0$	$(-1)^s 2^{e-126}(0.f)$
$e = 0, f = 0$	$(-1)^s(0.0)$
$e = 255, f \neq 0$	$(-1)^s(INF)$
$e = 255, f = 0$	NaN

表 6：倍精度の浮動小数点の値

指数部と仮数部の値	表現される浮動小数点の値
$0 < e < 2047$	$(-1)^s 2^{e-1023}(1.f)$
$e = 0, f \neq 0$	$(-1)^s 2^{e-1022}(0.f)$
$e = 0, f = 0$	$(-1)^s(0.0)$
$e = 2047, f \neq 0$	$(-1)^s(INF)$
$e = 2047, f = 0$	NaN