Fortran Interface

```
include 'fpvm3.h'
```

Process Control

```
call pvmfmytid( tid )
call pvmfexit( info )
call pvmfkill(  tid, info )
call pvmfaddhost( host, info )
call pvmfdelhost( host, info )
call pvmfnotify(about, msgtag, ntask, tids, info)
call pvmfspawn( task, flag, where,
                ntask, tids, numt )
```

Information

```
call pvmfparent( tid )
call pvmfperror( msg, info )
call pvmftidtohost( tid, dtid )
call pvmfconfig( nhost, narch, dtid, host,
                 arch, speed, info)
call pvmftasks(  which, ntask, tid, ptid, dtid,
                 flag, task, info)
call pvmfgetopt( what, val )
call pvmfsetopt( what, val, oldval )
```

Group Operations

```
call pvmfjoingroup( group, inum )
call pvmflvgroup(   group, info )
call pvmfgsize(     group, size )
call pvmfgettid(    group, inum, tid )
call pvmfgetinst(   group, tid, inum )
call pvmfbarrier(   group, count, info )
call pvmfbcast(     group, msgtag, info )
call pvmfreduce( op, xp, nitem, type, tag,
                 group, root, info )
```

op options

PvmMax	PvmMin	PvmSum	PvmProduct

Message Buffers

```
call pvmfmkbuf( encoding, bufid )
call pvmffreebuf( bufid, info )
call pvmfgetsbuf( bufid )
call pvmfgetrbuf( bufid )
call pvmfsetsbuf( bufid, oldbuf )
call pvmfsetrbuf( bufid, oldbuf )
call pvmfinitsend( encoding, bufid )
```

Encoding options		MEANING
PvmDataDefault	0	XDR
PvmDataRaw	1	no encoding
PvmDataInplace	2	data left in place

Sending

```
call pvmfpack( type, xp, nitem, stride, info )
```

Pack/unpack and Reduce type options

STRING	0	REAL4	4
BYTE1	1	COMPLEX8	5
INTEGER2	2	REAL8	6
INTEGER4	3	COMPLEX16	7

```
call pvmfsend(  tid,   msgtag, info )
call pvmfmcast( ntask, tids, msgtag, info )
call pvmfpsend( tid, msgtag, xp, nitem, type, info )
```

Receiving

```
call pvmfrecv(  tid, msgtag, bufid )
call pvmfprobe( tid, msgtag, bufid )
call pvmfnrecv( tid, msgtag, bufid )
call pvmftrecv( tid, msgtag, sec, usec, bufid )
call pvmfprecv( tid, msgtag, xp, nitem, type,
                rtid, rtag, ritem, info )
call pvmfbufinfo(bufid, bytes, msgtag, tid, info)

call pvmfunpack( type, xp, nitem, stride, info )
```

Declarations

```
INTEGER    about, bufid, count, dtid, encoding
INTEGER    flag, info, inum, msgtag, mstat
INTEGER    narch, nhost, nitem, ntask, numt
INTEGER    oldbuf, oldset, oldval, pstat, ptid
INTEGER    ritem, root, rtag, rtid, sec, set
INTEGER    signum, speed, stride, tid
INTEGER    tids(ntask), type, usec, which
CHARACTER  arch, group, host, msg, task, where
'type'     xp(nitem*stride)
```

ERROR CODE		MEANING
PvmOk	0	okay
PvmBadParam	-2	bad parameter
PvmMismatch	-3	barrier count mismatch
PvmNoData	-5	read past end of buffer
PvmNoHost	-6	no such host
PvmNoFile	-7	no such executable
PvmNoMem	-10	can't get memory
PvmBadMsg	-12	can't decode received msg
PvmSysErr	-14	pvmd not responding
PvmNoBuf	-15	no current buffer
PvmNoSuchBuf	-16	bad message id
PvmNullGroup	-17	null group name is illegal
PvmDupGroup	-18	already in group
PvmNoGroup	-19	no group with that name
PvmNotInGroup	-20	not in group
PvmNoInst	-21	no such instance in group
PvmHostFail	-22	host failed
PvmNoParent	-23	no parent task
PvmNotImpl	-24	function not implemented
PvmDSysErr	-25	pvmd system error
PvmBadVersion	-26	pvmd-pvmd protocol mismatch
PvmOutOfRes	-27	out of resources
PvmDupHost	-28	host already configured
PvmCantStart	-29	failed to exec new slave pvmd
PvmAlready	-30	already doing operation
PvmNoTask	-31	no such task
PvmNoEntry	-32	no such (group,instance)
PvmDupEntry	-33	(group,instance) already exists

Parallel Virtual Machine

Quick Reference Guide

Release 3.3

March 21, 1994

University of Tennessee

Oak Ridge National Laboratory

Emory University

Obtaining PVM

ftp: `netlib2.cs.utk.edu` directory pvm3/
email: `netlib@ornl.gov` with the message
`send index from pvm3 .`

C Interface

```
#include "pvm3.h"
```

Process Control

```
int tid  = pvm_mytid(void)
int info = pvm_exit( void)
int info = pvm_kill( int tid)
int info = pvm_addhosts(char **hosts, int nhost,
                        int *infos)
int info = pvm_delhosts(char **hosts, int nhost,
                        int *infos)
int numt = pvm_spawn(char *task, char **argv,
                     int flag, char *where,
                     int ntask, int *tids)
```

Spawn flag options		MEANING
PvmTaskDefault	0	don't care where
PvmTaskHost	1	"where" contains host
PvmTaskArch	2	"where" contains arch
PvmTaskDebug	4	start tasks with debug on
PvmTaskTrace	8	start tasks with trace on
PvmHostCompl	32	use complement host set

Information

```
int tid  = pvm_parent(void)
int dtid = pvm_tidtohost(int tid)
int info = pvm_perror(char *msg)
int info = pvm_config(int *nhost, int *narch,
                      struct hostinfo **hostp)
int info = pvm_tasks(int which, int *ntask,
                     struct taskinfo **taskp)
int    val = pvm_getopt(int what)
int oldval = pvm_setopt(int what, int val)
```

what option		SETS/GETS This	
PvmRoute	1	routing policy :	PvmRouteDirect
PvmDebugMask	2	debug level	PvmAllowDirect
PvmAutoErr	3	auto error reporting	
PvmOutputTid	4	stdout device for children	
PvmOutputCode	5	output msgtag	
PvmTraceTid	6	trace device for children	
PvmTraceCode	7	trace msgtag	
PvmFragSize	8	message fragment size	

Signalling

```
int info = pvm_sendsig(int tid, int signum)
int info = pvm_notify(int about, int msgtag,
                      int ntask, int *tids)
```

About options		MEANING
PvmTaskExit	1	notify if task exit
PvmHostDelete	2	notify if deletion
PvmHostAdd	3	notify if addition

Message Buffers

```
int bufid  = pvm_mkbuf(int encoding)
int info   = pvm_freebuf(int bufid)
int bufid  = pvm_getsbuf(void)
int bufid  = pvm_getrbuf(void)
int oldbuf = pvm_setsbuf(int bufid)
int oldbuf = pvm_setrbuf(int bufid)
int bufid  = pvm_initsend(int encoding)
```

Encoding options		MEANING
PvmDataDefault	0	XDR
PvmDataRaw	1	no encoding
PvmDataInPlace	2	data left in place

Sending

```
int info = pvm_packf(   printf-like format... )
int info = pvm_pkbyte(  char   *cp, int cnt, int std )
int info = pvm_pkcplx(  float  *xp, int cnt, int std )
int info = pvm_pkdcplx( double *zp, int cnt, int std )
int info = pvm_pkdouble(double *dp, int cnt, int std )
int info = pvm_pkfloat( float  *fp, int cnt, int std )
int info = pvm_pkint(   int    *np, int cnt, int std )
int info = pvm_pklong(  long   *np, int cnt, int std )
int info = pvm_pkshort( short  *np, int cnt, int std )
int info = pvm_pkstr(   char   *cp )

int info = pvm_send(  int  tid, int msgtag )
int info = pvm_mcast( int *tids, int ntask, int msgtag )
int info = pvm_psend(  int tid, int msgtag,
                       void *vp, int cnt, int type )
```

Receiving

```
int bufid = pvm_recv(   int tid, int msgtag )
int bufid = pvm_probe(  int tid, int msgtag )
int bufid = pvm_nrecv(  int tid, int msgtag )
int bufid = pvm_precv(  int tid, int msgtag,
                        void *vp, int cnt, int type
                        int *rtid, int *rtag, int *rlen )
int bufid = pvm_trecv(  int tid, int msgtag,
                        struct timeval *tmout )
int info = pvm_bufinfo( int bufid, int *bytes,
                        int *msgtag, int *tid)

int info = pvm_unpackf(   printf-like format... )
int info = pvm_upkbyte(  char   *cp, int cnt, int std )
int info = pvm_upkcplx(  float  *xp, int cnt, int std )
int info = pvm_upkdcplx( double *zp, int cnt, int std )
int info = pvm_upkdouble(double *dp, int cnt, int std )
int info = pvm_upkfloat( float  *fp, int cnt, int std )
int info = pvm_upkint(   int    *np, int cnt, int std )
int info = pvm_upklong(  long   *np, int cnt, int std )
int info = pvm_upkshort( short  *np, int cnt, int std )
int info = pvm_upkstr(   char   *cp )
```

Group Operations

```
int inum = pvm_joingroup(char *group)
int info = pvm_lvgroup(  char *group)
int size = pvm_gsize(    char *group)
int tid  = pvm_gettid(   char *group, int inum)
int inum = pvm_getinst(  char *group, int tid)
int info = pvm_barrier(  char *group, int count)
int info = pvm_bcast(    char *group, int msgtag)
int info = pvm_reduce( void *op, void *vp, int cnt,
       int type, int msgtag, char *group, int root)
```

op options	vp type options	
PvmMax	PVM_BYTE	PVM_FLOAT
PvmMin	PVM_SHORT	PVM_DOUBLE
PvmSum	PVM_INT	PVM_CPLX
PvmProduct	PVM_LONG	PVM_DCPLX

Starting PVM

```
pvmd [-nhostname] [-d<debugmask>] [hostfile]
pvm [hostfile] (starts console)
```

PVM Console Commands

```
help [command] – get information about commands
conf           – lists hosts in virtual machine
add    host(s) – add host(s) to virtual machine
delete host(s) – delete host(s)
spawn [opt] file  – spawn process
    -<count>   – number of tasks to spawn
    -<host>    – host to spawn on
    ->         – redirect task output to console
    ->file     – redirect task output to file
    ->>file    – append task output to file
ps [-a]        – lists processes on virtual machine
alias          – define/list command aliases
unalias        – undefine command alias
setenv         – set/show environment variables
echo           – echo arguments
version        – print libpvm version
id             – print console tid
sig num tid    – send signal num to process
kill tid       – terminate a process
reset          – kill all processes and reset PVM
quit           – exit console (PVM continues)
halt           – kill all pvmds and console
```

Compiling PVM Applications

```
cc -o task myprog.c libpvm3.a
f77 -o task myprog.f libfpvm3.a libpvm3.a
```

For groups add `libgpvm3.a` before `libpvm3.a`