PULSAR IN EUROPORT ESPRIT PROJECT

PULSAR MAIN GOALS

- **ACHIEVE PORTABLE PARALLELIZATION OF AN ADVANCED SAR IMAGE UNDERSTANDING LIBRARY**
- **BENCHMARKING AND EVALUATION ON DIFFERENT PLATFORMS**
- **IDENTIFY POTENTIAL BENEFITS OF THE PARALLEL PACKAGE**
- **DISSEMINATE RESULTS**

PULSAR PARTNERS

- **ELSAG BAILEY - Genova, ITALY -** ***COORDINATOR + PARALLEL EXPERT + MACHINE PROVIDER***
- **NA SOFTWARE - Liverpool, UK -** ***SW PACKAGE PROVIDER***
- **UNIVERSITY OF LIVERPOOL - UK -** ***PARALLEL EXPERT + PARALLEL TOOL PROVIDER***
- **SCOT CONSEIL - Ramonville Cedex , FRANCE -** ***APPLICATION USER EXPERT***

PULSAR WORKING PERIOD

24 MONTHS, from JAN 1994 to DEC 1995

Elsag Bailey

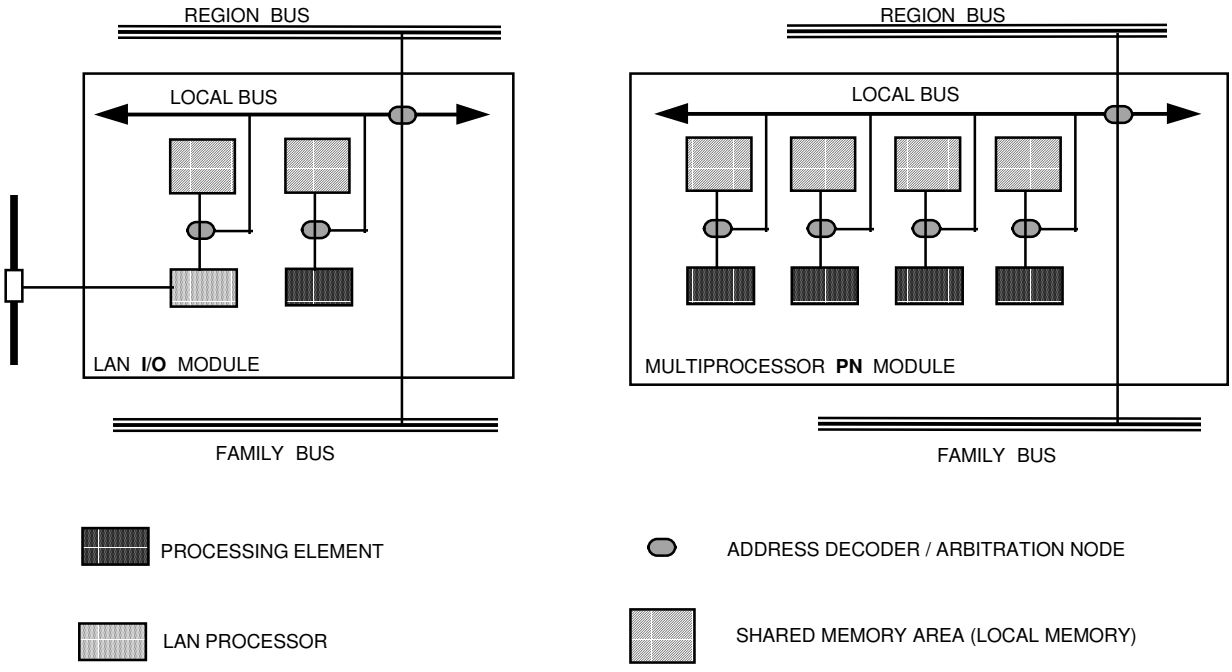
PULSAR - PVM FOR EMMA2E

INTRODUCTION TO PORTING CRITERIA

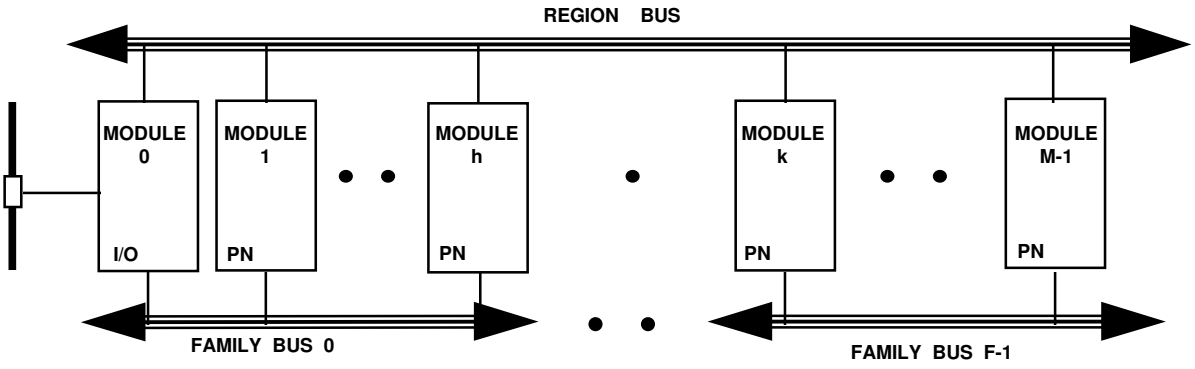
MAIN OBJECTIVES FOR PORTING:

- **Do as simple work as possible.** Within the scope of PULSAR, a complex porting would be risky for achieving success before the end of 1994.
- I**f possible, do not modify EROS up to user interface.** Developing PVM as a user library + service tasks decouples it from EROS complexity, provides better means for debugging and implies use of more reliable services.
- **In the trade-off between standard implementation and efficiency, choose the first.** The main PULSAR objective is parallel application portability, for which getting a partial PVM on EMMA2E could lead to miss the target. Instead, getting a relatively high overhead could be acceptable, since benchmarks will imply high parallel granularity. Efficiency may be increased in the future, only if industrial application for PVM is forecasted.
- **Avoid custom implementation of C library and Unix interface.** This could lead to a bigger job than expected, with limited benefits for EROS interface. Only an effort of POSIX porting would be worth doing, in the meantime it is better to implement strictly necessary primitives.
- **Even using proprietary inter-processor communication facilities, hide details in PVM interface.** Whichever means of fast data exchange among processors is chosen, it should not influence PVM interface. Only quantitative or dynamic limitations are acceptable, if reasonable in using EMMA2E.

EMMA2E PROCESSING MODULES



EMMA2E ARCHITECTURE



EXAMPLE SCHEME OF AN EMMA2E TARGET NODE WITH

- **ONE** I/O MODULE FOR LAN INTERFACE
- **M-1** PN MODULES, FOR A TOTAL OF **M** MODULES AND **4M - 3** PROCESSORS
- **F** FAMILY BUSSES

PVM PORTABILITY

PVM claims to be portable, since it relies on C library, some Unix standard services and communication Socket Library in Stream (TCP) and Datagram (UDP) modes.
In addition, inter-task communication within a node may be substituted by a proprietary implementation, especially in the case of multiprocessors.

However, being EROS completely proprietary except for inter-node communication, we must carefully examine all dependencies, listed below.

a) System-independent C libraries or macros: *qsort, strcmp, memset, memcmp, bcmp, memcpy, ispace, strncmp, atoi, ntohl, strncpy, strcpy, FD_SET, FD_RESET, FD_CLR, FD_ZERO, strcat, tnpman, index, strncat, strchr, isdigit, isupper, htonl, htons, htohs, random, dup2 (**28**).*

b) System-dependent C libraries: *fopen, fgets, flcose, fprintf, malloc, realloc, free, fcntrl, fflush, exit, printf, ctime, fputc, fputs (**14**).*

c) Socket and communication services: *gethostbyname, socket, gethostname, bind, getsockname, setsockopt, connect, listen, select, accept, sendto, recvfrom, getservbyname (**13**).*

d) Unix system services: *fork, kill, perror, ioctl, close, write, abort, read, open, getenv, sleep, getpid, unlink, authfn, stat, execvp, execv, _exit, wait, va_arg, va_start, va_end, sysconf, getpwuid, endpwent, end_servent, setsid, setpgid, setpgrp, signal, chdir, waitpid, wait3, gettimeofday, pipe, time, localtime, rexec (**38**).*

PVM PORTABILITY FOR EROS

With EROS of EMMA2E, things are quite different. Having neither POSIX nor Unix services, most PVM dependencies are not implementable. With more details:

a) System-independent C libraries or macros: availability should be straightforward, just depending on libraries available with Intel or (in case of adoption) Metaware C compiler.

b) System-dependent C libraries: Standard I/O could be implemented on top of EROS file I/O, but it would be heavy and with uncomplete results. Instead, local services like *malloc, exit, ctime* are implementable with little effort.

c) Socket and communication services: they should be available on SLIC processors, but currently they are internal to EROS. With the aim of not modifying EROS structure, the custom user interface of CE_xxx primitives can be used, probably with limitations but also interoperability with Socket library on a standard node. Communcation should rely on UDP/IP, so far used.

d) Unix system services: maybe some of them are easy to implement (for instance, *signal, wait, sleep, getpid*) but the job is clearly not possible for all dependencies emerged.

IT IS EVIDENT THAT THE INTEGRAL PORTING OF PVM CODE TO EROS THROUGH RESOLUTION OF ALL RUN-TIME SYSTEM DEPENDENCIES IS NOT POSSIBLE.

PVM PORTING STRATEGY FOR PULSAR

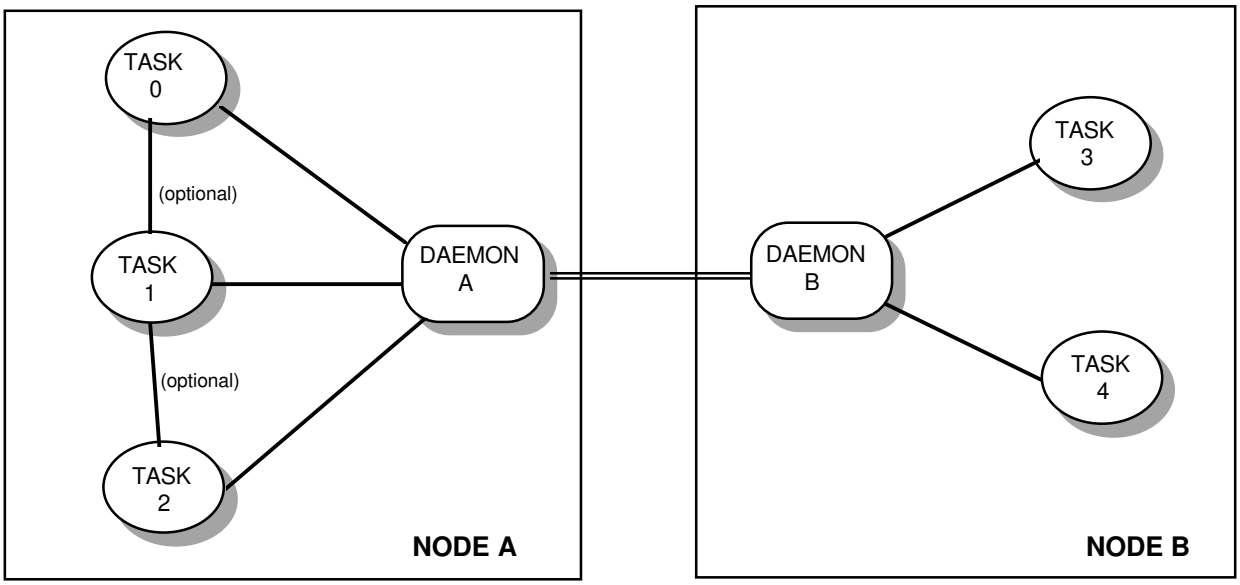
With the initial objectives and system dependencies emerged, our porting strategy for PULSAR will be based on the following points:

- **try to just port primitives required for DDL.** So far, they seem to be (Liverpool Univ.):
  - *pvm_initsend, pvm_pkbyte, pvm_pkint, pvm_upkbyte, pvm_upkint*
    probably just dependent on **memory management;**
  - *pvm_mytid, pvm_send, pvm_mcast, pvm_recv, pvm_probe*
    probably just dependent on **memory management** and **point-to-point communication,** even if the effects of Daemon involvment could enlarge the dependencies;
  - *pvm_bufinfo, pvm_getrbuf, pvm_serror*
    probably just independent of system services, and
  - *pvm_spawn*
    surely dependent on **Unix system services for task creation, I/O, environment, remote commands...** but avoidable if implementation is too complex (suggestion of Liverpool Univ.);

- **try to separate PVM Library from PVM Daemon.** Strictly they should reside on the same node, but also **remoting the EMMA2E-PVM Daemon on a Unix node** may be the key to reduce porting effort while maintaining most functionalities.

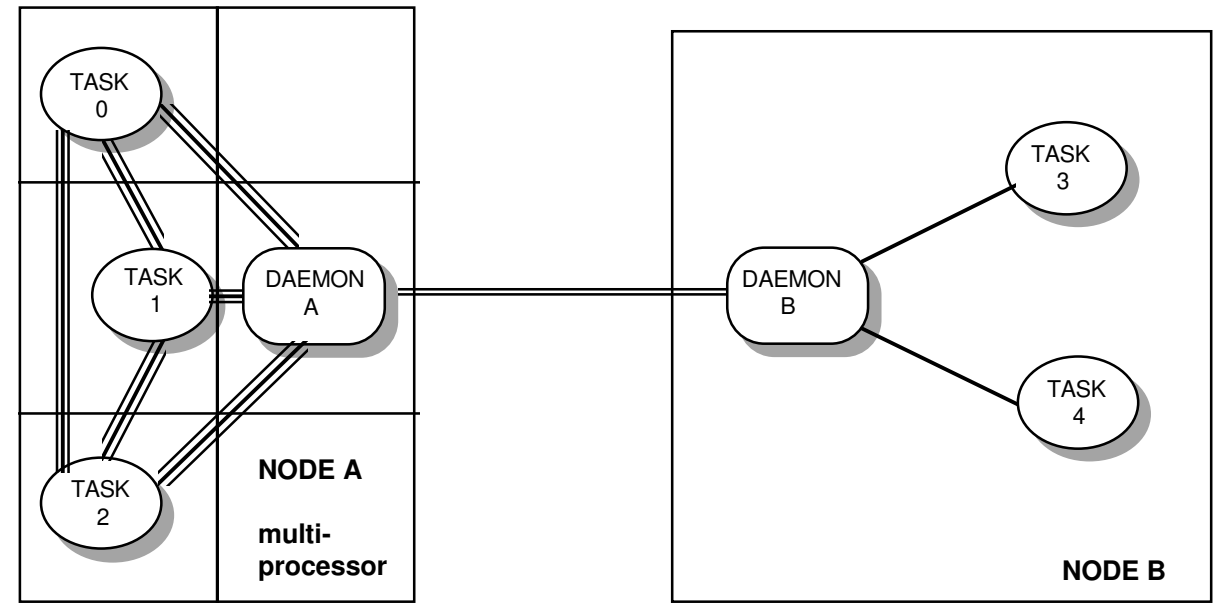
RESULTING PORTING STRATEGY: **SEPARATE PVM LIBRARY, TO BE PORTED ON EMMA2E, FROM DAEMON, TO BE REMOTED ON A UNIX NODE, IN THE SAME WAY USED FOR EROS REMOTE I/O [inter-processor communication + routing tasks on SLIC + server task on Host] .**

PVM ARCHITECTURE ON STANDARD NODES



Elsag Bailey

PVM ON STANDARD AND MULTIPROCESSOR NODES

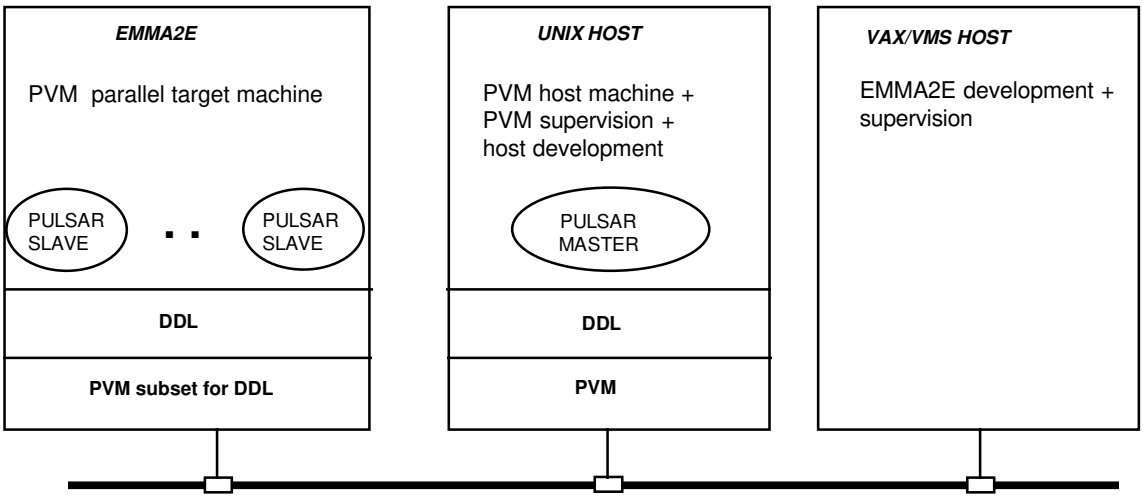


SOCKET / UDP link

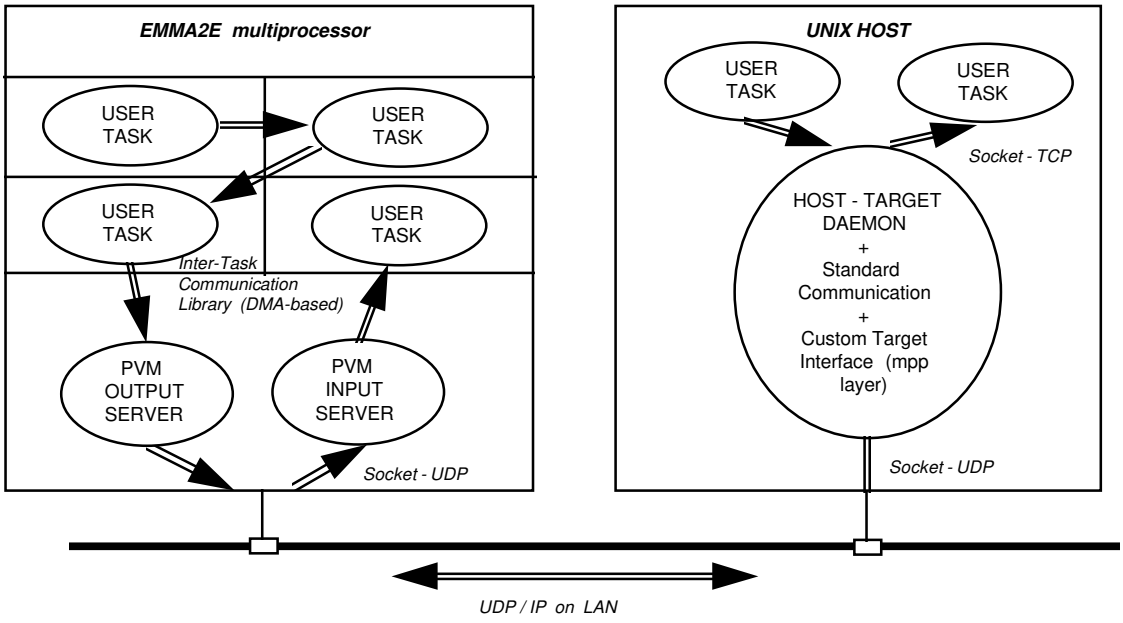
SOCKET / TCP link

MPP (multiprocessor point-to-point) link

MINIMAL SYSTEM CONFIGURATION FOR PULSAR

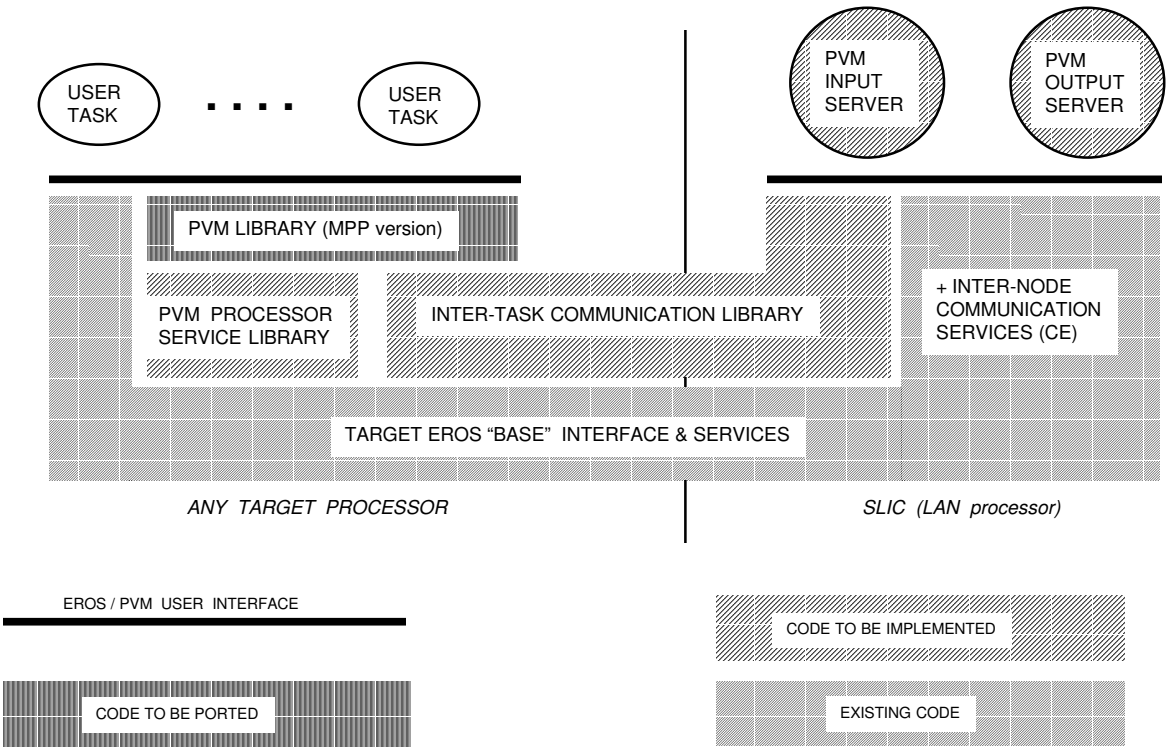
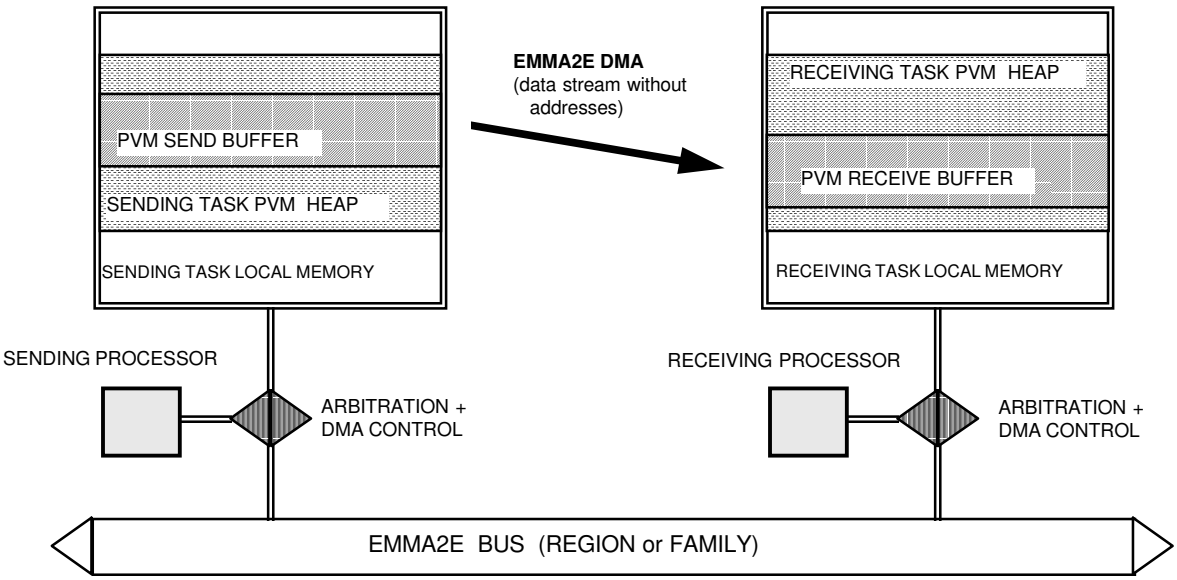


PVM HOST - TARGET ARCHITECTURE FOR EMMA2E



Elsag Bailey

PVM ARCHITECTURE ON EMMA2E NODES



DIRECT DATA MOVING BETWEEN SHARED BUFFERS