Embedded Distributed Systems: A Case of Study

Victor Rodriguez
The rise of embedded & IoT ....

The rise of data ....

The rise of real problems …

“The total amount of user data (data payload) to be stored or processed doubles every two years”

“Boeing 787s to create half a terabyte of data per flight, says Virgin Atlantic”
We are not the first one.... LMPI

An example of a traditional MPI.

An example of the LMPI system.

The CPU time with different number of nodes in Jacobi with 4 processes
We *might* not need a server

The main objective of this work will be to prove that some distributed embedded system can coordinate itself to process its own data without the need of an external HPC system.

![Diagram](image-url)
By definition: A distributed system consists of a collection of autonomous computers, connected through a network and distribution middleware, which enables computers to coordinate their activities and to share the resources of the system, so that users perceive the system as a single, integrated computing facility.

Advantages

• Partitioned Workload:
• Heterogeneous HW:

Disadvantages

• Network:
• Lack of optimized OS:
Results

All Reduce

Performance of MPI Allreduce

- KB/sec vs. Message Size in Bytes
- Graph shows the performance of MPI Allreduce for different message sizes.
Results

Bidirectional Bandwidth

```c
if (am_i_the_master()){
    TIMER_START;
    for (i=0; i<cnt; i++){
        mp_Irecv(dest_rank, 2, destbuf, bytes, &requestarray[1]);
        mp_isend(dest_rank, 1, sendbuf, bytes, &requestarray[0]);
        MPI_Waitall(2, requestarray, statusarray);
    }
}
else if (am_i_the_slave()){
    for (i=0; i<cnt; i++) {
        mp_Irecv(source_rank, 1, destbuf, bytes, &requestarray[0]);
        mp_isend(source_rank, 2, sendbuf, bytes, &requestarray[1]);
        MPI_Waitall(2, requestarray, statusarray);
    }
}
```
Results

All to All

Performance of MPI Alltoall

MPLComm_size(comm, &n);
for (i = 0, i < n; i++)
    MPLSend(sendbuf + i * sendcount * extent(sendtype),
             sendcount, sendtype, i, ..., comm);
for (i = 0, i < n; i++)
    MPLRecv(recvbuf + i * recvcount * extent(recvtype),
             recvcount, recvtype, i, ..., comm);
Results

Latency and round trip
Future Work

Experiment 1

HPC Workloads ➔ MPI & P thread

清楚 Linux OS

Yocto
MPI patches on RFC

PnP Analysis
PERF

open source software
toolkit used for
building grids
Everybody wants the control...