

**Kris Gaj¹, Tarek El-Ghazawi², Nikitas Alexandridis²,
Jacek R. Radzikowski¹, Mohamed Taher²,
and Frederic Vroman²**

¹ George Mason University

² George Washington University

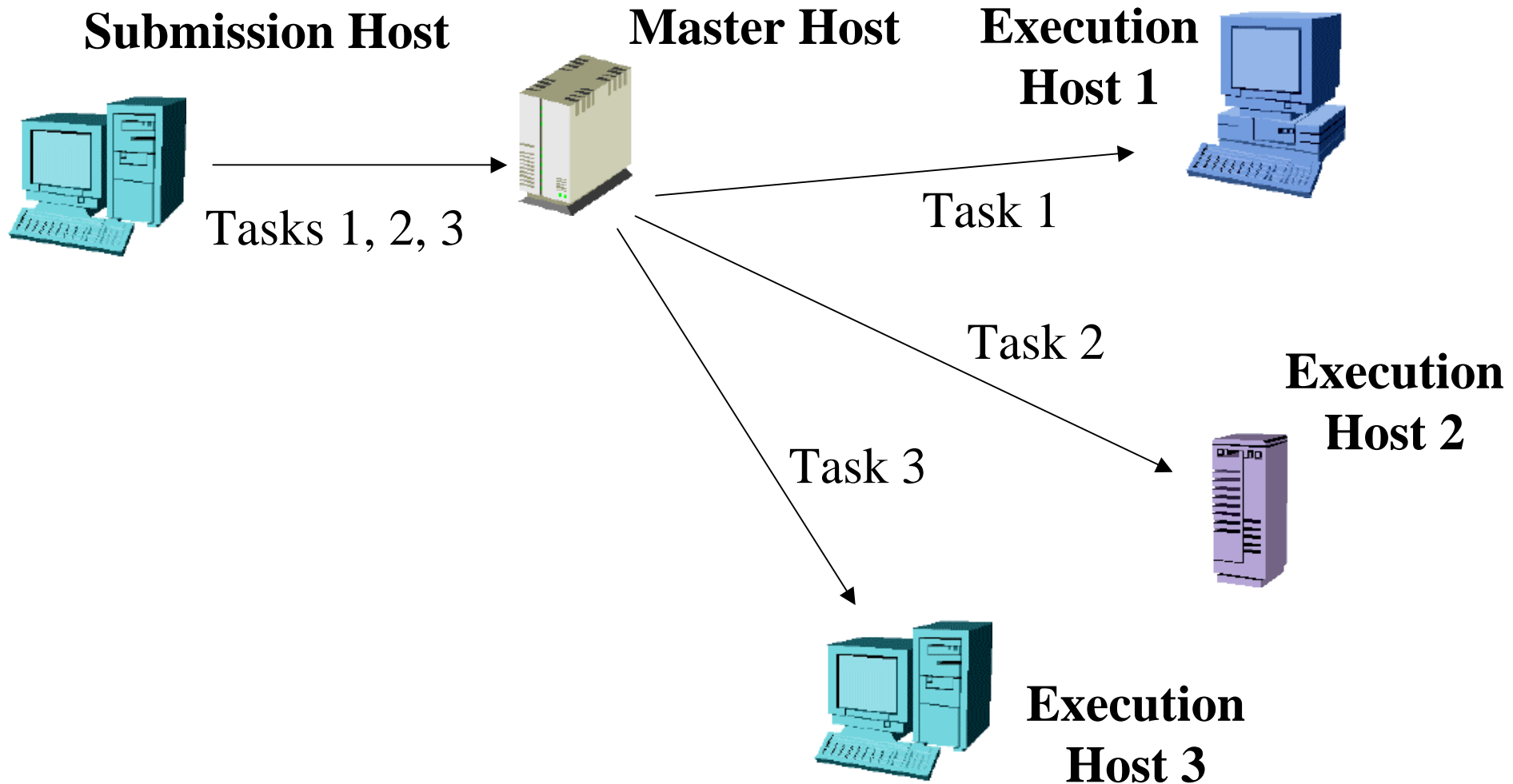
**Effective Utilization and Reconfiguration
of Distributed Hardware Resources
Using Job Management Systems**

<http://ece.gmu.edu/lucite>

Problem:

- **Reconfigurable resources expensive and underutilized**
- **Many of these resources available over the network**
- **It is desirable to leverage networked reconfigurable resources to help other users within the same organization**

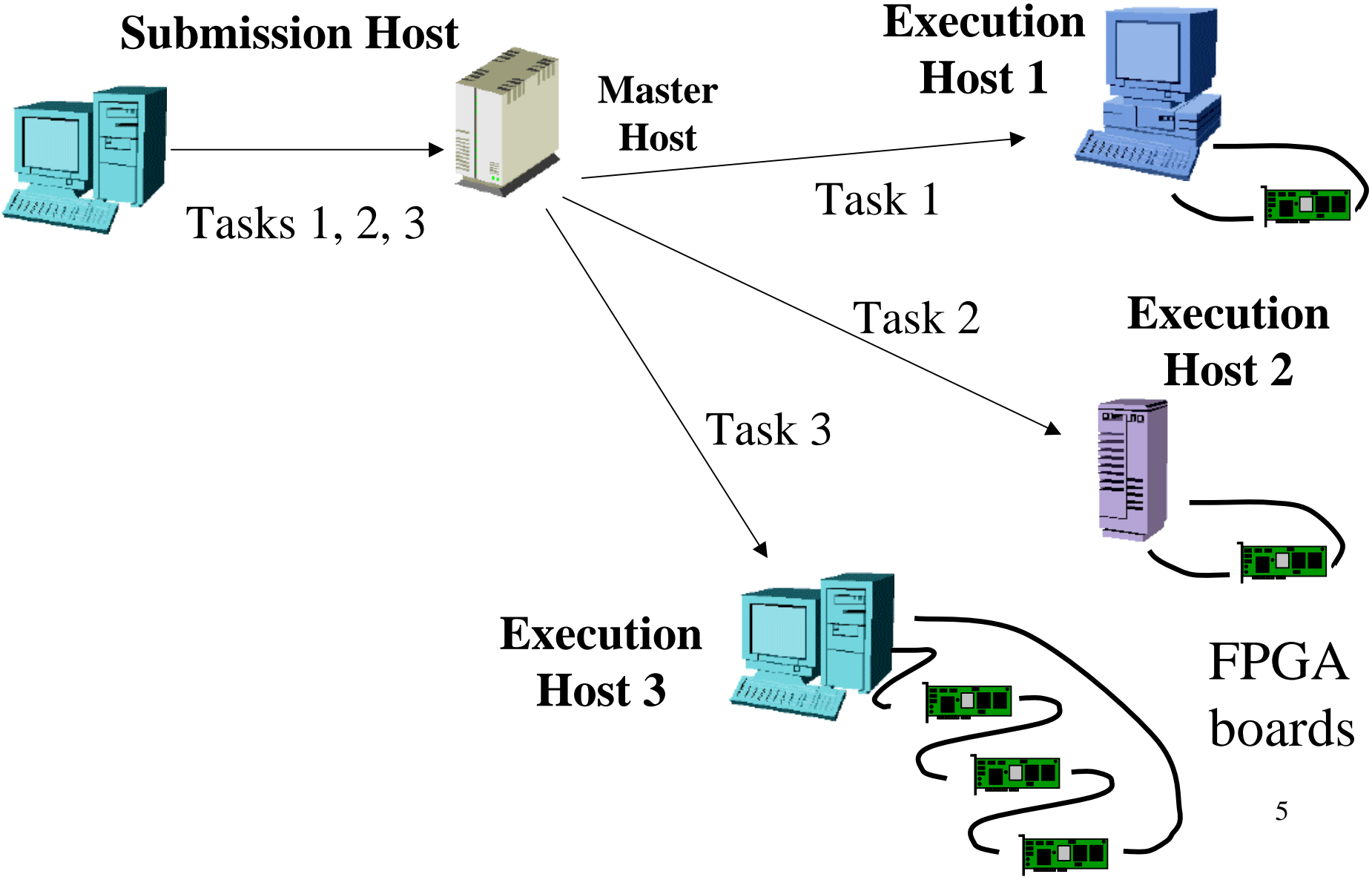
Approach: Adapt and Use a Job Management System



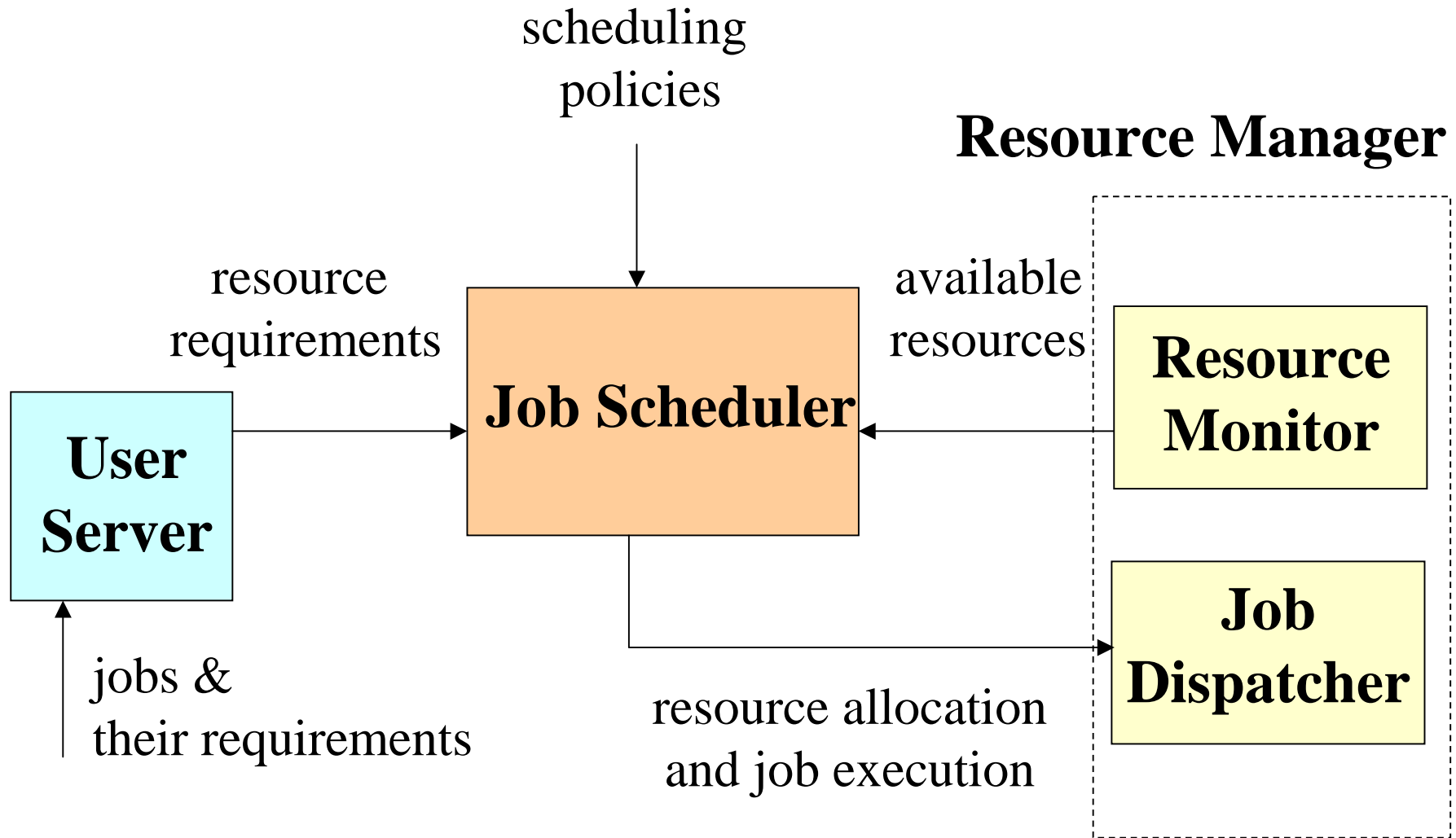
Approach:

- **Select the most suitable existing Job Management System (JMS)**
 - identify and define functional requirements
 - rank known systems according to these requirements
 - identify which JMS is the easiest to extend
- **Extend this JMS to recognize and utilize reconfigurable resources**
 - add new dynamic resources
 - configure scheduling to be based on these new resources

Networked Reconfigurable Resource Management System



Functional units of a typical Job Management System



Job Management Systems Compared in our Study

LSF - Load Sharing Facility

Sun Grid Engine / Codine

PBS - Portable Batch System

CONDOR

Platform Computing Corp.

Sun Microsystems

















Veridian Systems

University of Wisconsin





















Operating system, flexibility, user interface

	LSF	Codine	PBS	CONDOR
Distribution	com	pub	pub/com	pub
Source code	○	●	◐	●
OS Support				
Solaris	●	●	●	●
Linux	●	●	●	●
Tru64	●	●	●	○
NT	●	○	○	◐
User Interface	GUI & CLI	GUI & CLI	GUI & CLI	GUI & CLI

















Scheduling and Resource Management

	LSF	Codine	PBS	CONDOR
Batch jobs				
Interactive jobs				
Parallel jobs				
Accounting				









Efficiency and Utilization

	LSF	Codine	PBS	CONDOR
Stage-in and stage-out				
Timesharing				
Process migration				
Dynamic load balancing				
Scalability				

Fault Tolerance and Security

	LSF	Codine	PBS	CONDOR
Checkpointing				
Daemon fault recovery				
Authentication				
Authorization				

Documentation and Technical Support

	LSF	Codine	PBS	CONDOR
Documentation				
Technical support				

JMS features supporting extension to reconfigurable hardware

- capability to define **new dynamic resources**
- strong support for **stage-in and stage-out**
 - configuration bitstreams
 - executable code
 - input/output data
- support for **Windows NT and Linux**

Ranking of Centralized Job Management Systems (1)

Capability to define new dynamic resources:

Excellent:	LSF, PBS, CODINE
More difficult:	CONDOR

Stage-in and stage-out:

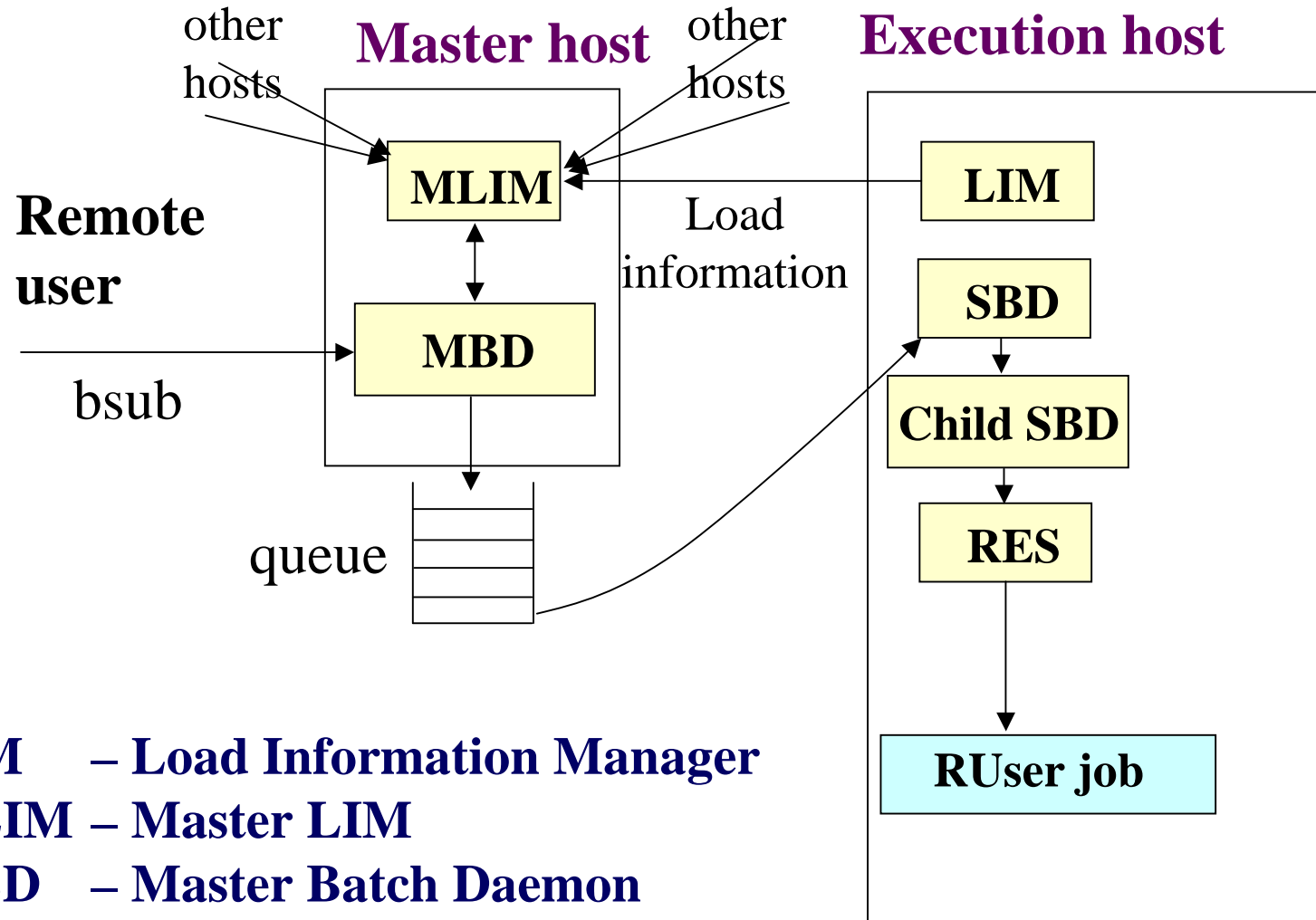
Excellent:	LSF, PBS
Limited:	CONDOR
No:	CODINE

Ranking of Centralized Job Management Systems (2)

Overall suitability to extend to reconfigurable hardware:

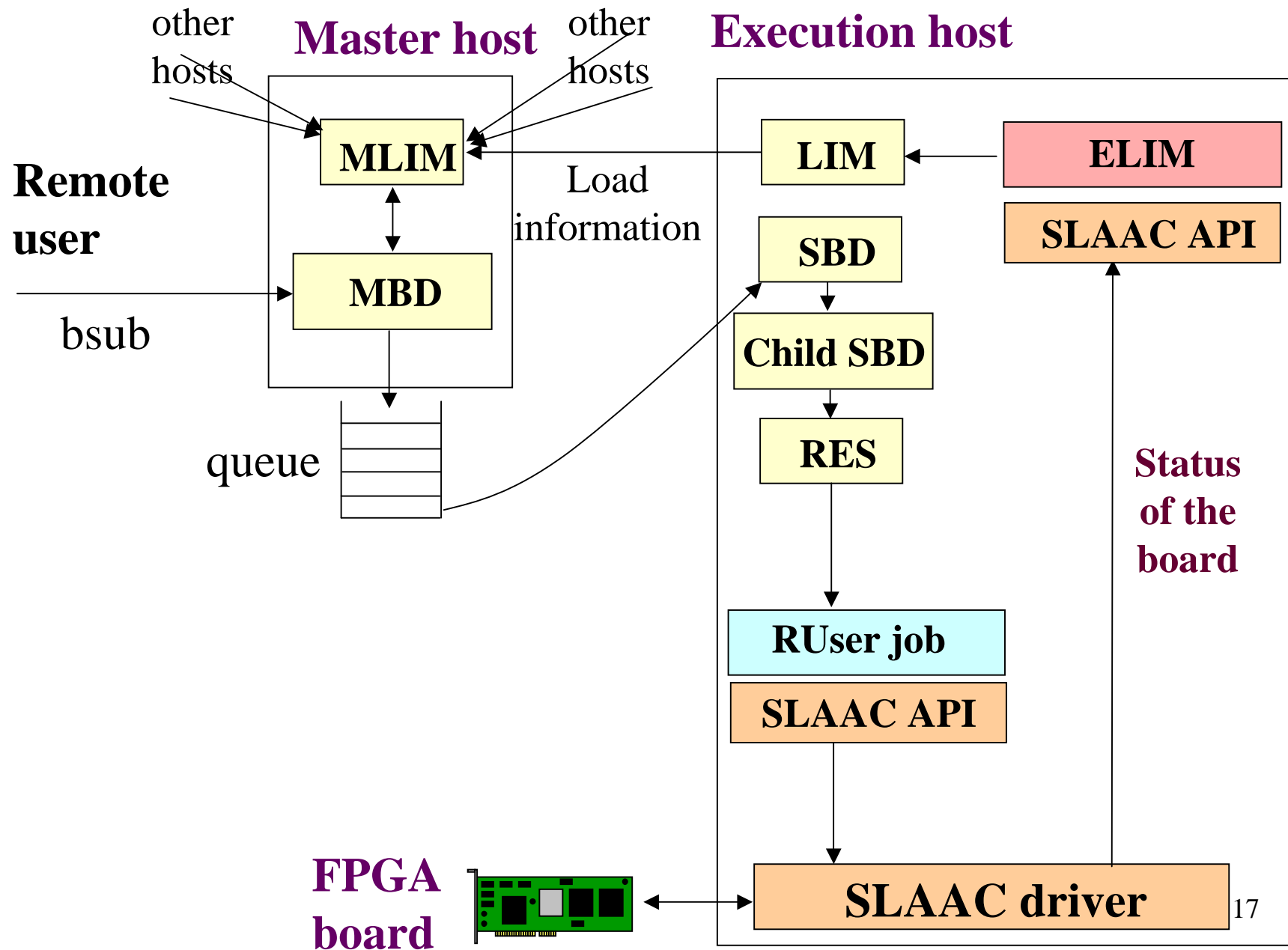
- | | | |
|------------------|---|--|
| 1. LSF | } | without changing the JMS source code |
| 2. CODINE | | |
| 3. PBS | | |
| <hr/> | | |
| 4. CONDOR | } | requires changes to the JMS source code |

Regular Operation of LSF

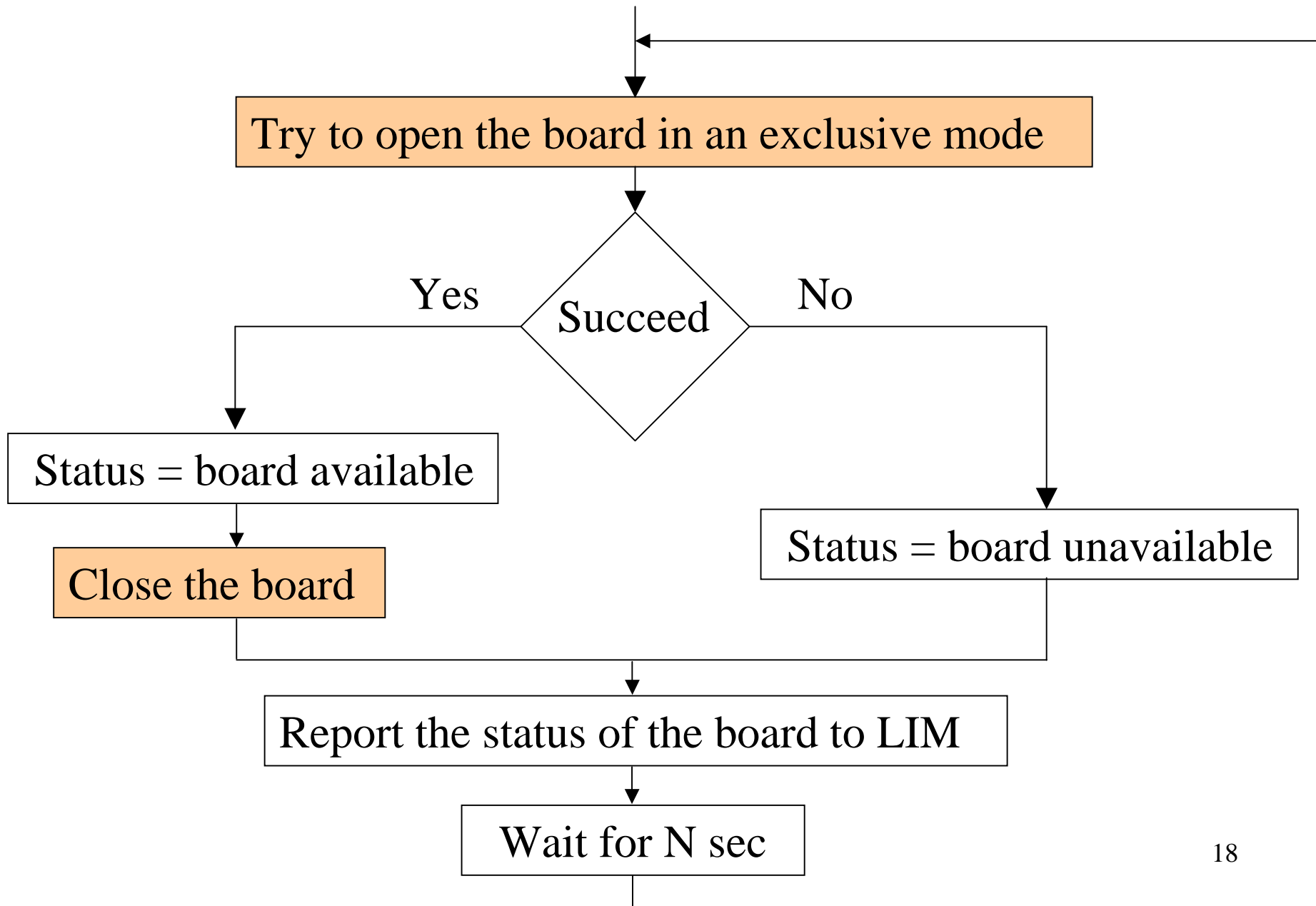


- LIM** – Load Information Manager
- MLIM** – Master LIM
- MBD** – Master Batch Daemon
- SBD** – Slave Batch Daemon
- RES** – Remote Execution Server

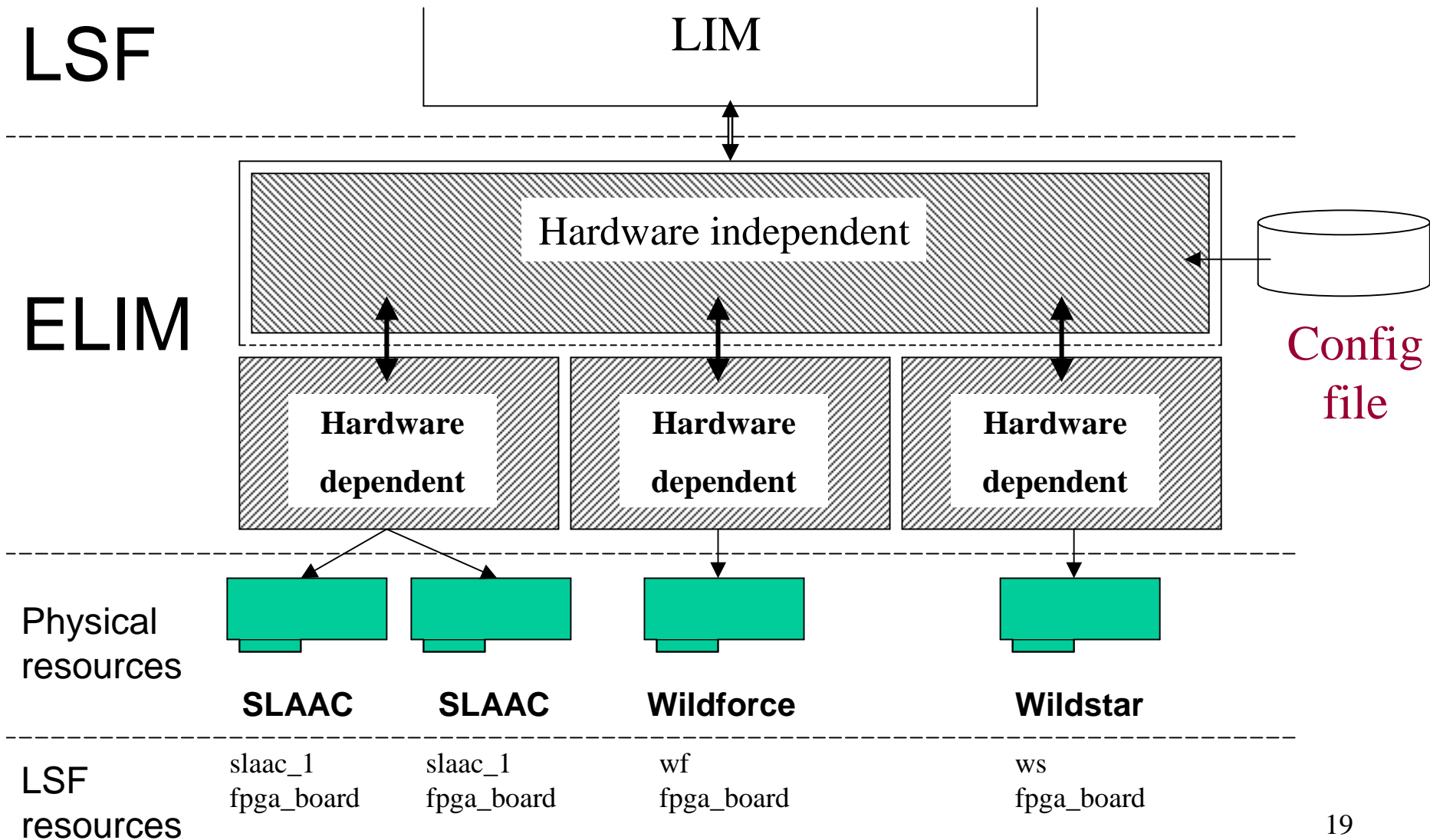
Extension of LSF to reconfigurable hardware



Operation of ELIM



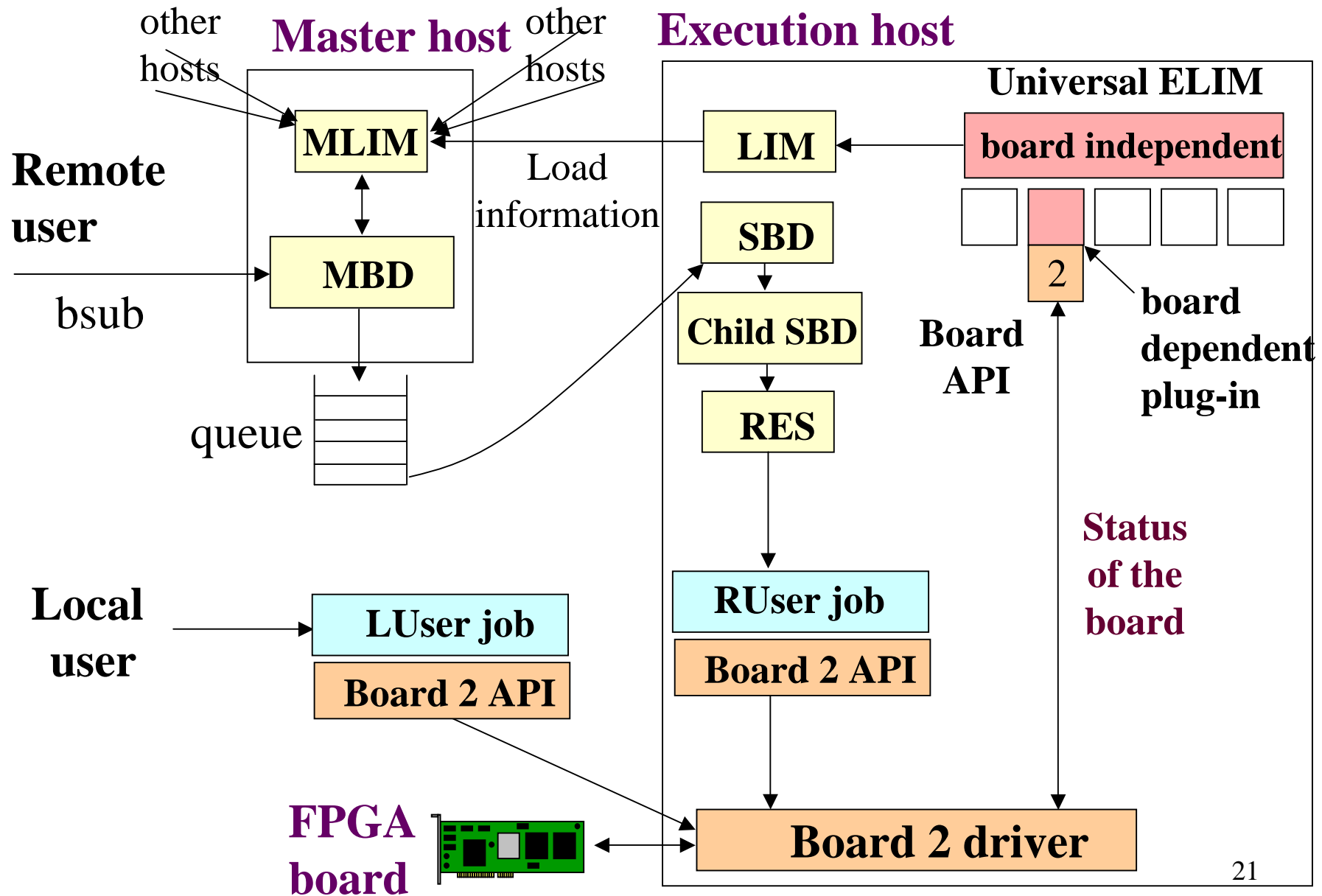
Modular ELIM structure



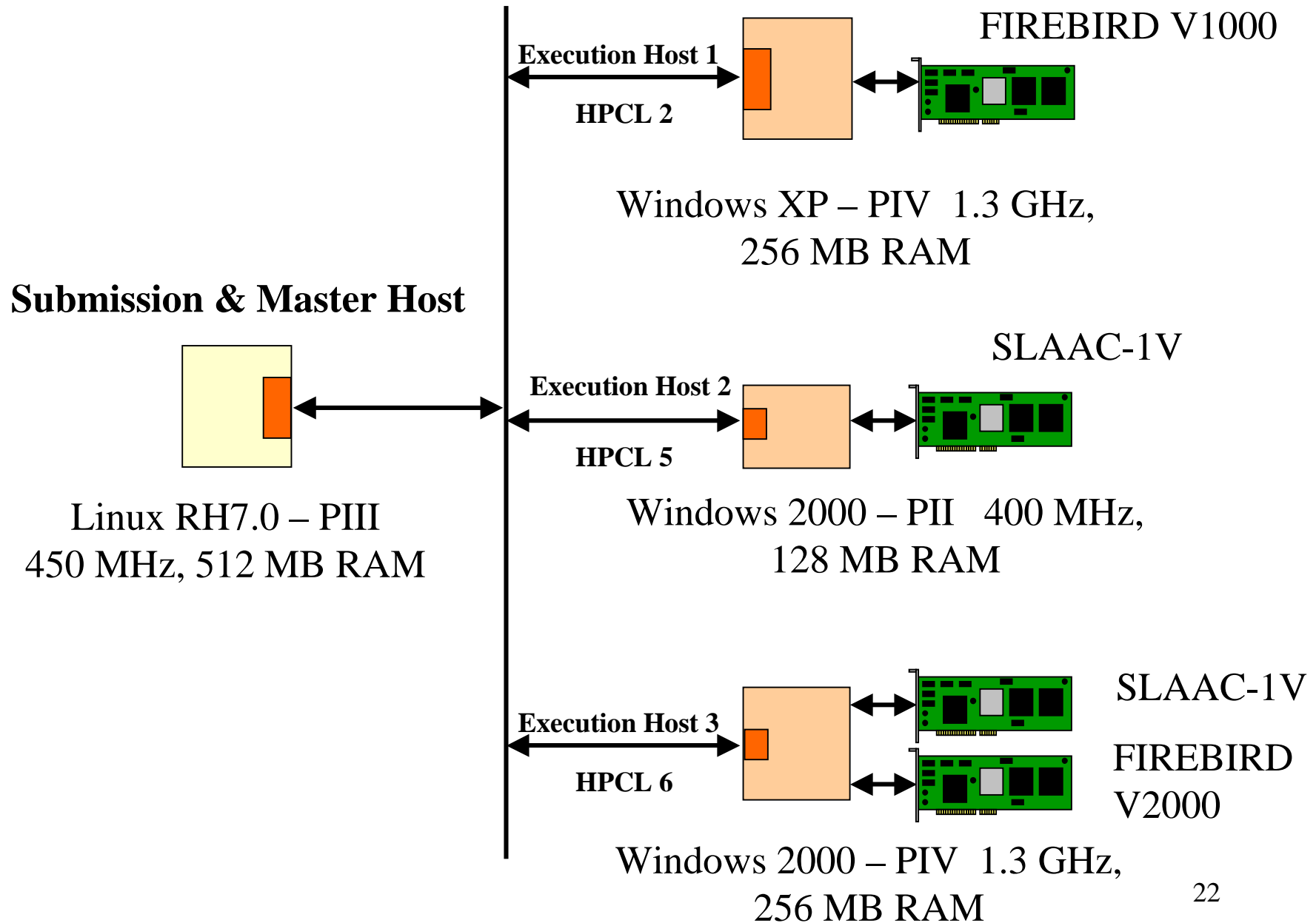
Modular ELIM - implementation

- Hardware independent part supports multiple resources and multiple instances of a single resource.
- One physical resource can be assigned to several logical resources
- One logical resource can be assigned to several physical resources
- Configuration stored in an XML file
- Plugins implemented as run-time loaded modules
- The interface between parts implemented as function calls

Implementation of ELIM using plugins

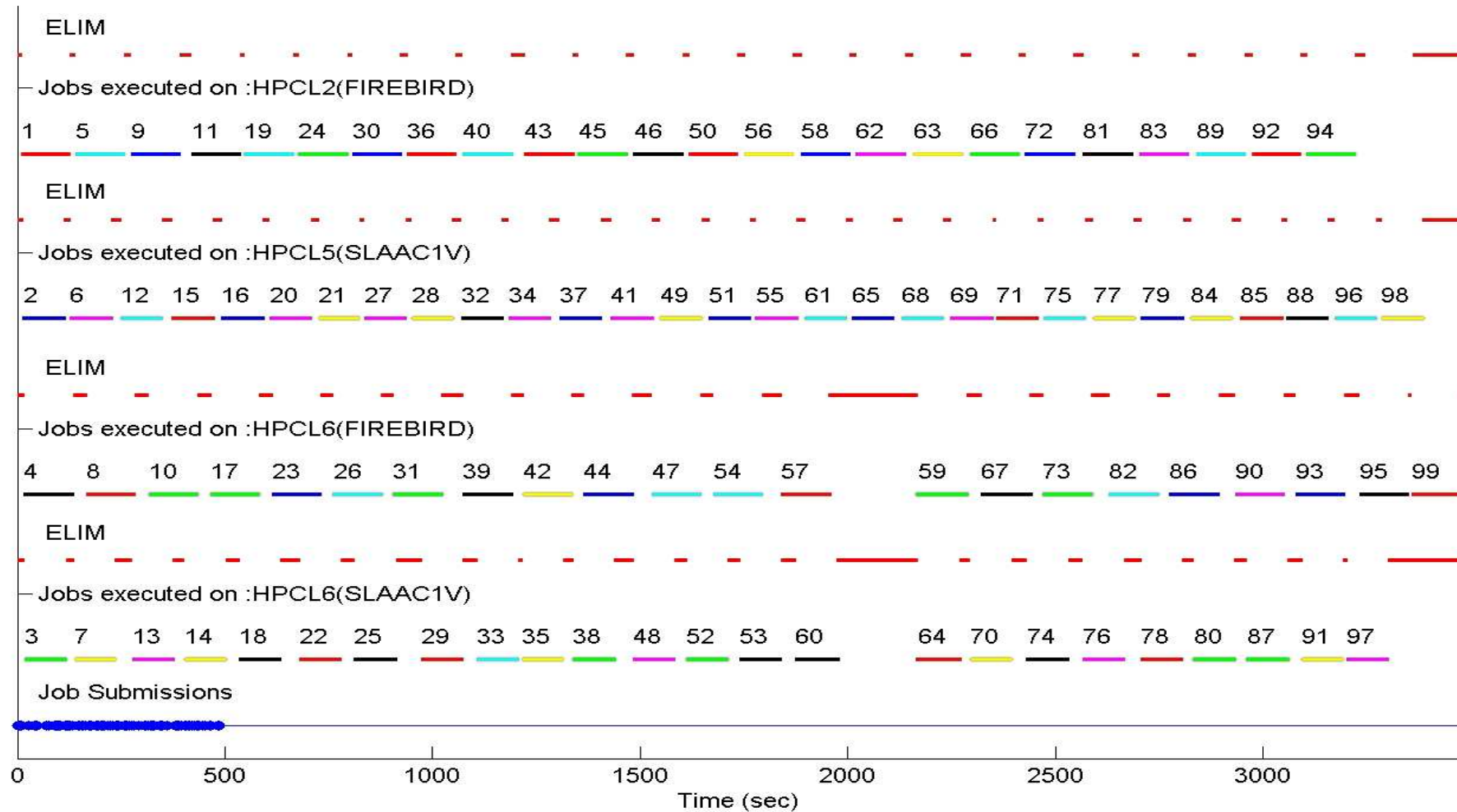


Experimental Testbed



DES Breaker
Ciphertext=0X95F8A5E5DD31D90A
Plaintext=0X8000000000000000

Search Space from 0X 0000000000000000 to 0X000000E8D4A50F9C
Key not found



Estimated speed-up compared to software

Network with four distributed FPGA boards

31,362,155,470 keys / 3,500 sec \approx 8,960,615 keys/sec

Software implementation of the DES breaker running on Pentium 4

100,000,000 keys / 198 sec \approx 505,050 keys/sec

Estimated speed-up compared to software- cont.

Measured speed-up

≈ 18

Potential for much higher speed-ups:

- multiple DES breaking engines on a single FPGA device (only about 7% of the FPGA resources utilized to date)
- fully pipelined architecture
- full usage of all FPGA devices on the accelerator boards

Estimated practical speed-up: > 500

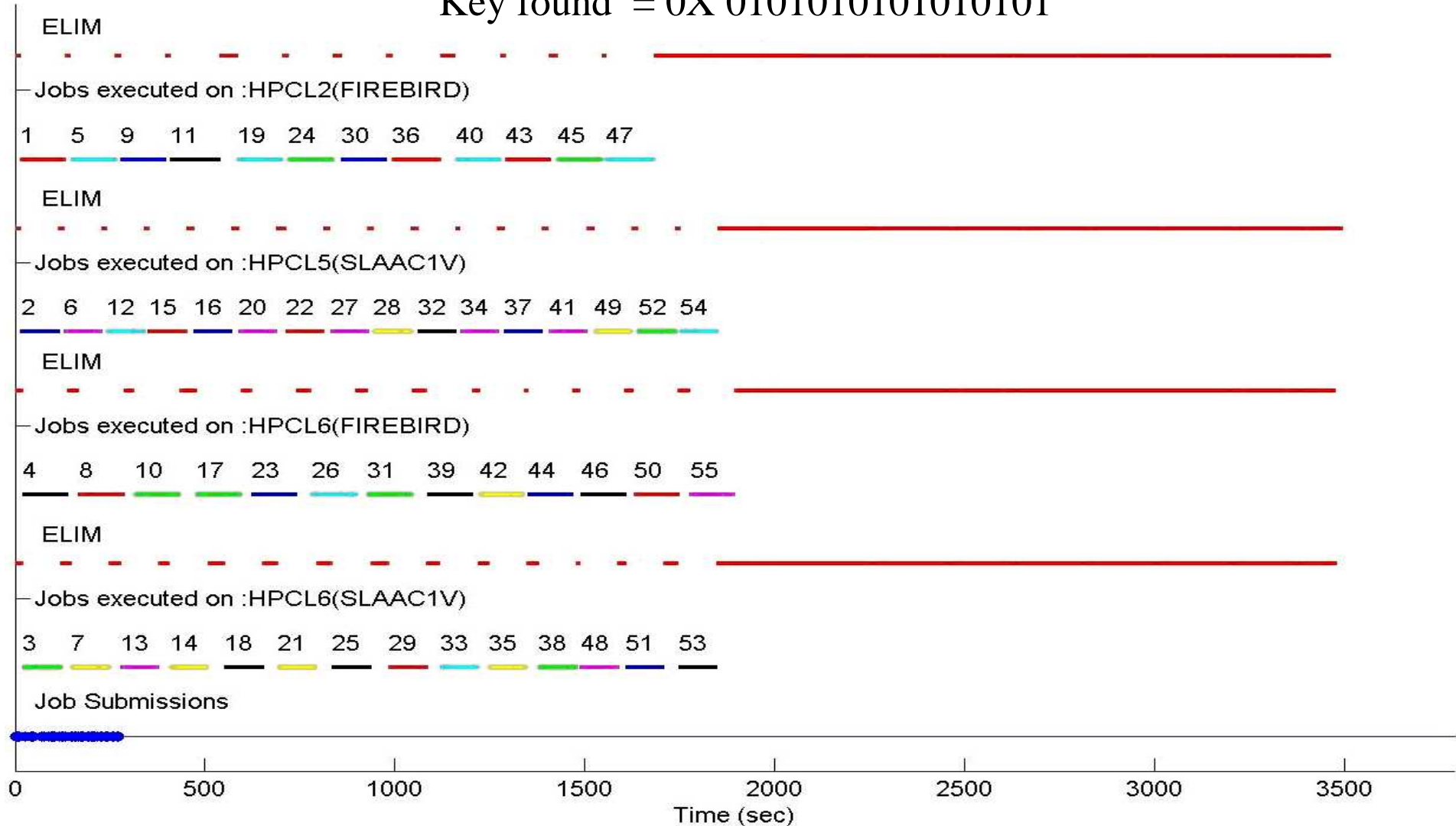
DES Breaker

Ciphertext=0X 8CA64DE9C1B123A7

Plaintext=0X 0000000000000000

Search Space from 0X 1010100C5663702 to 10101013C9BCB00

Key found = 0X 0101010101010101



Utilization of FPGA boards during unsuccessful attempts to find the key

Iteration	utilization
1	85.5 %
2	82.3 %
3	82.4 %
4	80.8 %
5	82.7 %

Conclusions

- **Several popular Job Management Systems compared and evaluated**
- **LSF, PBS Pro, Sun Grid Engine / CODINE shown to be easily extendable to support FPGA boards**
- **The general architecture of the extended system developed**
- **An extension of LSF developed and tested experimentally**

Conclusions (cont)

- **Experimental verification in a testbed consisting of Windows and Linux workstations, and three types of FPGA boards**
- **Benchmark based on the exhaustive key search for the DES cipher**
- **Experiments proved the correctness of our concept and the feasibility of its implementation**
- **The efficiency of the extended system, measured in terms of the average utilization of reconfigurable resources, reaching 86%**