

# Advanced Computer Systems Architecture

## Chip-Multiprocessors: Applications and Architectures

CSE 526M

Prof. Patrick Crowley

## Plan for Today

- Questions
- Today's discussion

## Outline

- Background
- Successes/Failures
- Small Address Space Study

## What is C.mmp/Hydra?

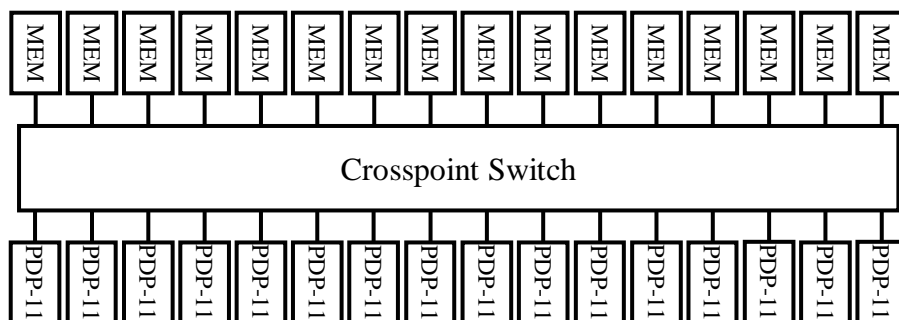
- C.mmp
  - CMU MultiMiniProcessor (built with PDP-11s)
  - No caches, uniform memory access from all CPUs
  - Goal: *Cost-effective* general-purpose shared-memory multiprocessor
- Hydra
  - Kernel-based OS for C.mmp
  - Goal: OS services can be built by users to enable experimentation

## What is a PDP-11?

- 16-bit computer from DEC
  - late '60s, early '70s
- CISC
- Tidbit: first machine with only a memory bus, no bus for I/O.
  - All I/O was handled through special memory addresses
- More info: <http://en.wikipedia.org/wiki/PDP-11>

## C.mmp Organization

- 16 processors, 16 memory banks
- Total performance: 6 MIPS, 500 Mbps



## Hydra

- OS kernel on top of which OS services can be built, by users
- Protection and object reference based on *capabilities*
  - i.e., the only references available are capabilities, which encode the access rights of the holder

## Successes and Failures

- |  |   |
|--|---|
| <ul style="list-style-type: none"><li>• <b>Successes:</b> (1 of 5 are HW)<ul style="list-style-type: none"><li>– Cost-effective, symmetric MP</li><li>– Hydra is user-extensible</li><li>– Hydra is distributed</li><li>– Reliability in software</li><li>– Good Hydra engineering</li></ul></li></ul> | <ul style="list-style-type: none"><li>• <b>Failures:</b> (3 of 5 are HW)<ul style="list-style-type: none"><li>– Hardware is not reliable</li><li>– PDP-11 has small address space</li><li>– C.mmp cannot be partitioned</li><li>– Poor user interface</li><li>– Poor project management</li></ul></li></ul> |
|--|---|

## A Cost-effective Multiprocessor

- Design goals
  - Speed
  - Simplicity
  - Off-the-shelf components
- System complaints
  - PDP-11 has too few addresses (64K)
  - High OS overhead
  - Memory contention

## Extensible OS

- The extension abstraction is the protected subsystem
  - Implemented over 20 subsystems (i.e., services)
- Verdict
  - Abstraction was good for fast implementation, revision and integration with existing system
  - Development environment poor
  - It was difficult to build schedulers outside the kernel

## The Distributed OS

- All Hydra instances on all processors are peers
  - no client/server relationships needed
- System tasks can run anywhere
  - Except for I/O tasks which must run on a processor attached to the particular device
- Provides a range of synchronization primitives
  - Fast locks
  - Slow semaphores

## Reliability

- Goal: provide software error detection and recovery, regardless of error source
- Error detection
  - HW: parity checking on all memory transactions
  - SW: redundant representations, etc.
- Error recovery
  - *Suspect-monitor* paradigm
  - When one processor gets hung, another re-boots it

## Hardware Reliability

- Mean-time-between-failures for C.mmp/Hydra: 2-6 hours
  - 2/3 of failures were 100% HW
- Culprits
  - PDP-11 UNIBUS
  - Drums
  - Crosspoint switch can be hung by misbehaving clients
  - HW group wrote poor diagnostic code

## Address Space

- Only 64K addresses
- Consequences
  - OS had to help extend memory size
  - Programs/data must be fractured
- Realizations
  - Users did not write small program pieces
  - OS-managed paging was expensive

## System Partitioning

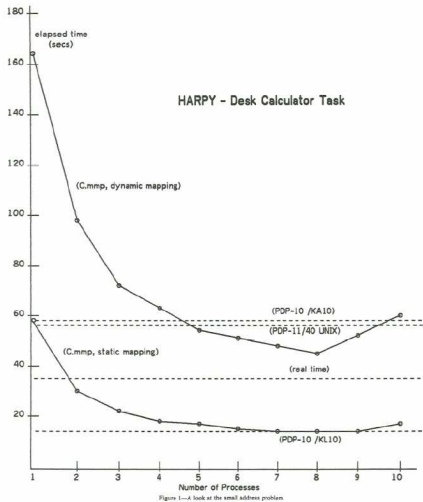
- Idea: disjoint subsystems will enable coincident usage among: users, diagnostics, maintenance, upgrades, etc.
- Infeasible on C.mmp/Hydra
  - Although some maintenance can be partitioned
  - Not enough I/O devices to go around
  - Design does not account for the sharing or communication of capabilities between partitions

## Human Interface Engineering

- User interface did not receive much attention
- Consequences
  - Long learning process
  - System lacked conventional conveniences and languages
  - However, the Command Language was a big hit (allowed complete access to Hydra environment)

## Address Space Problem

- Program run on several PDP-based machines
- Dynamic vs. static
  - Paging is never needed, but dynamic version checks anyway



## Project Management

- Medium-sized team, 15 or so
- Failures
  - Too few people, wrong personnel
  - Novel aspects of system were over-emphasized, essential (and mundance) tasks were neglected
  - Informal management style felt right, but, in fact, led to problems in specifications, documentation and coding standards
    - Hydra team did not use C.mmp for development
    - Lack of specs prohibited parallel HW/SW development

## Intellectual Digestion

- What purpose does an OS serve?
- Does the IXP have an OS? Does it need one?
- Could you build something like the C.mmp in your spare time? How much would it cost?

## Purpose of OS

- To provide
  - Common services
  - Rights-based access to resources
  - Protection between different programs
  - Resource sharing to applications
  - A record of resource usage and system activity
  - A higher-level implementation target for applications

## OS on the IXP

- OS existence
  - XScale certainly has one
  - MEs do not
    - Although, certain tasks are supported in HW, such as synchronization and thread scheduling
- OS demand
  - Do MEs need separation?
  - Do MEs need to share resources?
  - Do MEs need to protect resources?
- Put another way, how would you use the IXP if it had this support?

## Assignment

- Thursday (4/29)
  - **Commentary:** The Cosmic Cube
  - 15 minute perspective: DDOS Shield/Firewall
- Tuesday (**Last class, 5/4**):
  - **Commentary:** The Burroughs Scientific Processor
  - 15 minute perspective: FAM