SONAR: Operating Systems for New (Computer) Architectures

Marisa Gil (marisa@ac.upc.es)
Nacho Navarro (nacho@ac.upc.es)

SONAR: Introduction

- Operating systems abstractions
- High performance multiprocessors
- (Micro-Exo) kernel, libraries, environments
- Application/kernel customization, extensibility, middleware, pervasiveness, real time
- Virtual architectures, machines, emulation
- Embedded systems, power management
- Multicore SOC, soft/hard interfaces

We are interested in …

- Embedded systems: cell phones, routers, home gateway, PDA, video/audio players
- 64 bit processors, open source tools and OS optimizations
- Linux and libraries optimization, efficient execution, customization
- Dynamic linking, interposition
- Emulation, VM, dynamic loading
- Sensor Networks, power management

OS and Multiprocessors

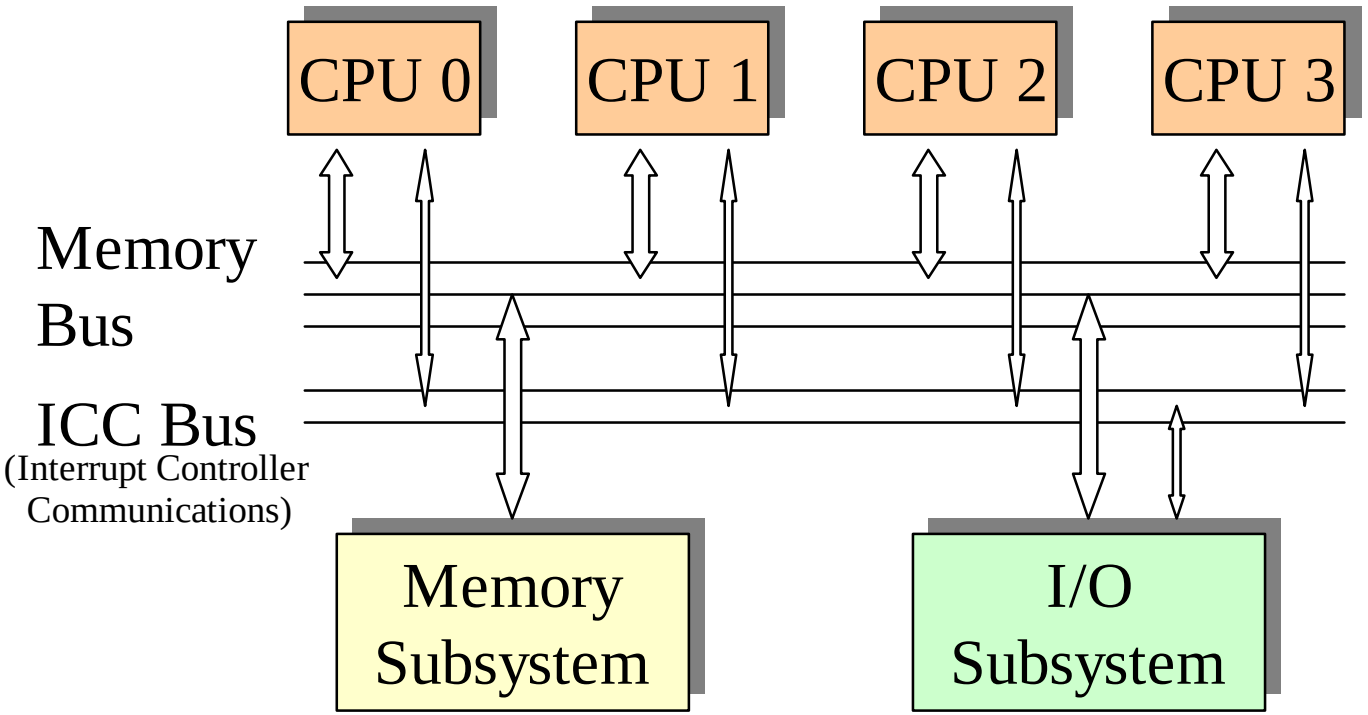
- Multiprocessor systems
  - SMP, NUMA, CC-NUMA
  - Processor pools
    - More CPUS than users; on demand allocation
  - Workstations
    - Processors, memory, display, network, multimedia
    - Diskless, mobility, dynamic loading
  - Heterogeneous
    - General and specific purpose
    - Network processors, DSP, FPGA, reconfigurable logic

OS and Multiprocessors

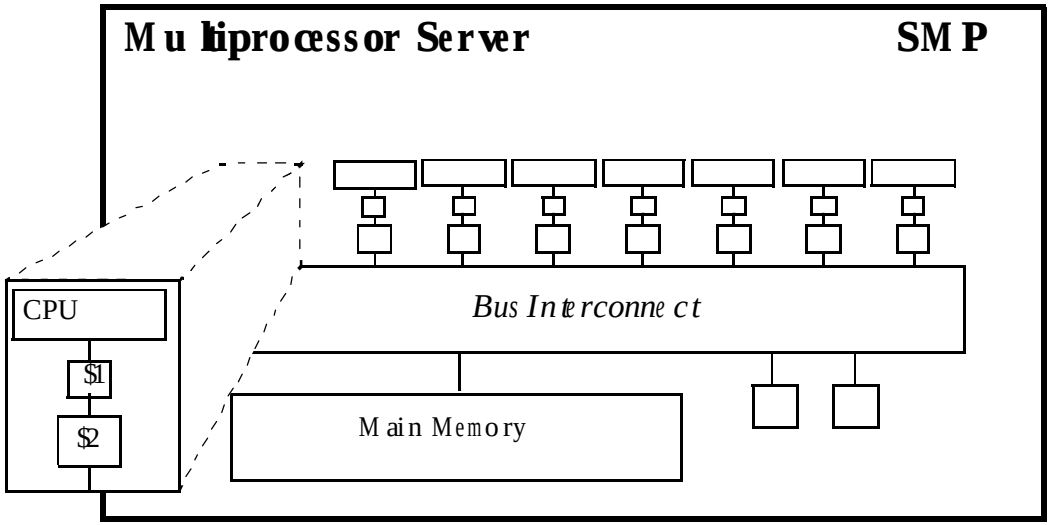
- Operating environments for multiprocessors
  - Monolithic and microkernel technologies
  - Execution abstraction: thread
  - More functionalities at user level, at application level
  - Emulation, extensibility and specialization at multiple software layers
  - Event based programming
  - Exception management
  - Asynchronous, efficient, parallel I/O

Shared memory (SMP)

- Shared CPU, clock, memory hierarchy, I/O
- Cache memory coherence
- Synchronous (IPI) vs. asynchronous (self scheduling) communication

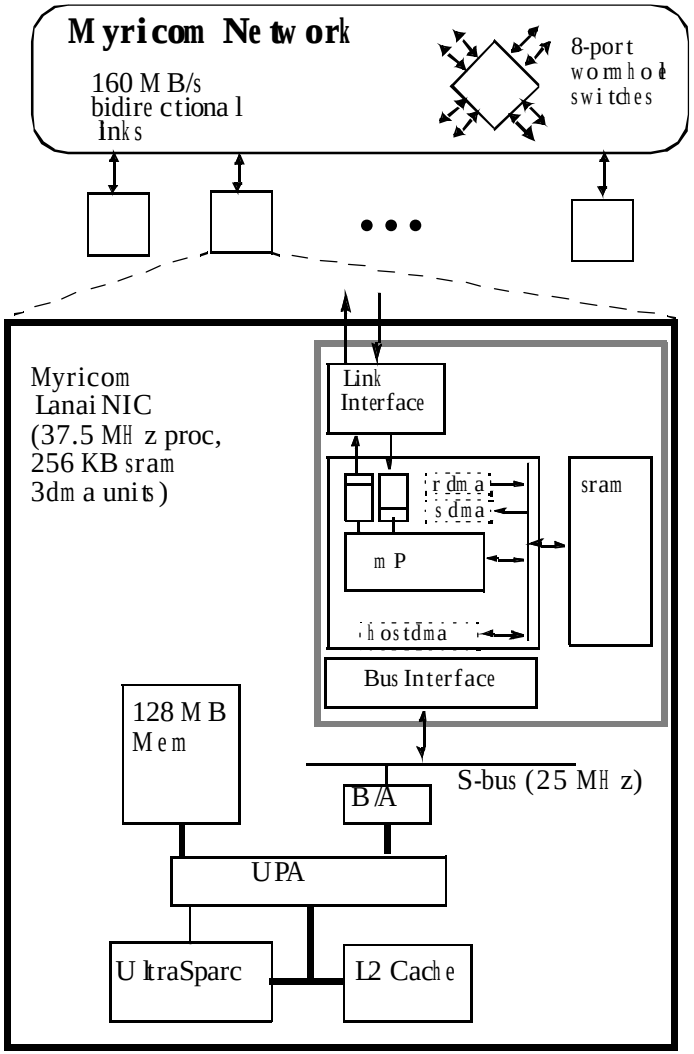


But SMP memory interconnection…



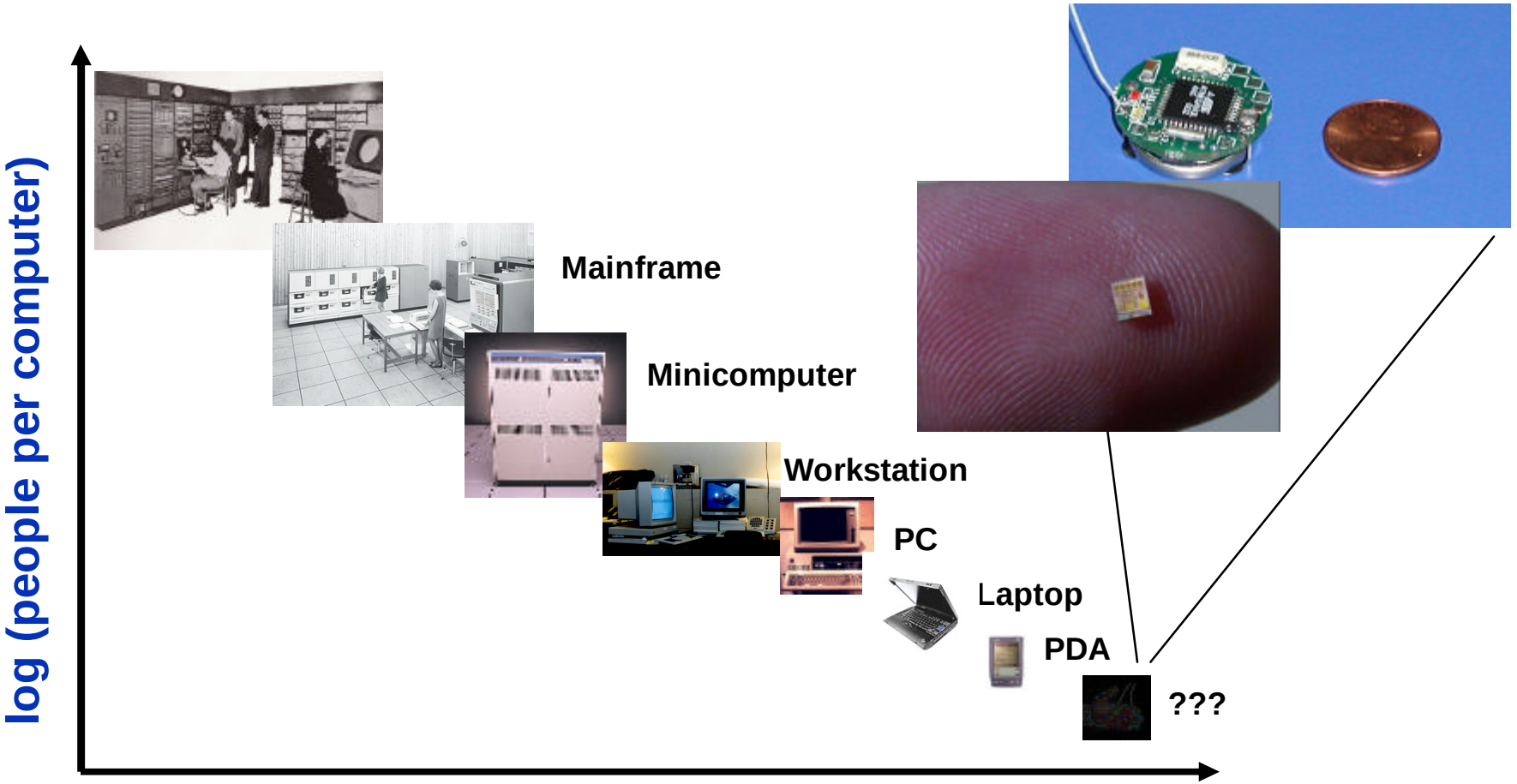
- Cache coherent NUMA
- Local and remote memory latency
- Thread migration and/or data migration
- PIM (processor in memory) proposals

Clusters , Networks



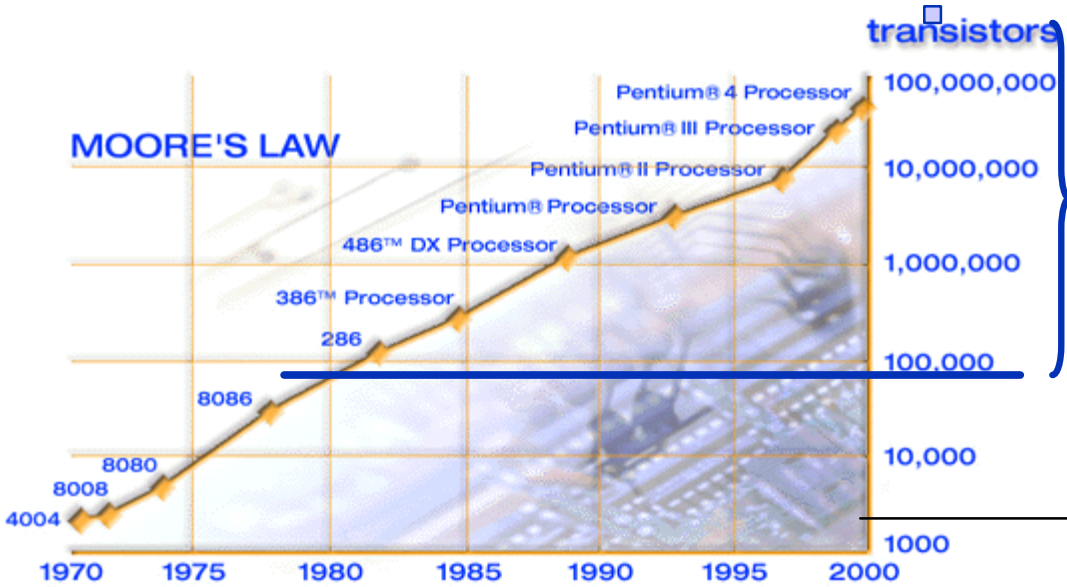
- High speed interconnections
- User-level interfaces
- Network processors
- Message passing programming models

A New Computer Class Emerging



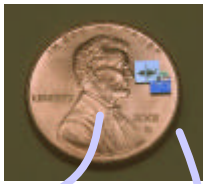
(David Culler, 2003)

CMOS Trends: miniaturization

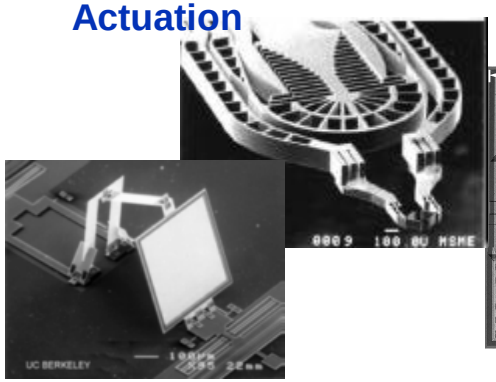


Itanium2 (241M)

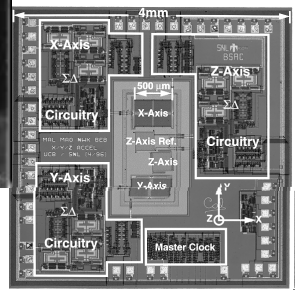
nearly a thousand 8086's would fit in a modern microprocessor



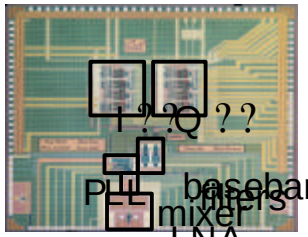
Actuation



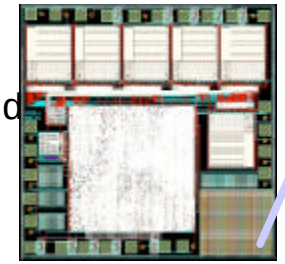
Sensing



Communication

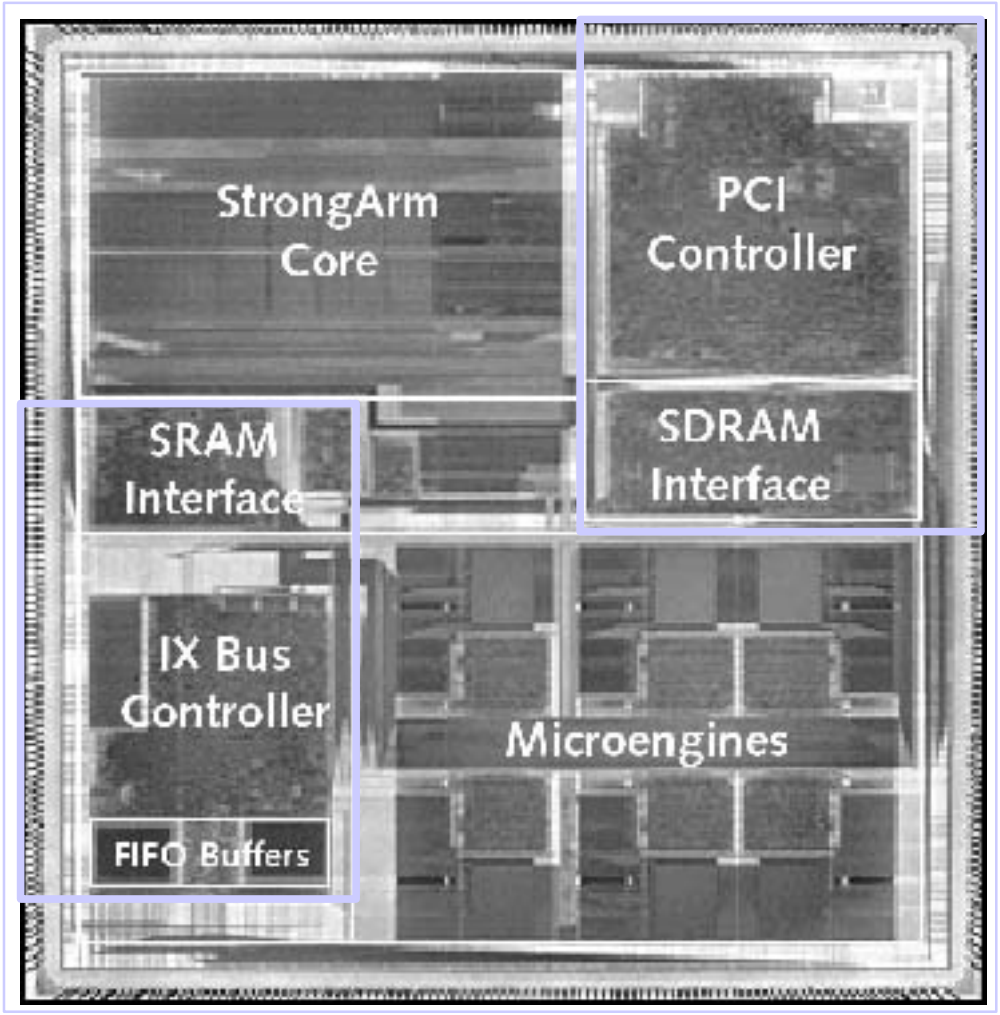


Processing & Storage



(David Culler, 2003)

Intel IXP1200 Network Processor



MIMD Network Processor for switching, routing, data processing

6 Micro engines (RISC 32b based) that are controlled by StrongARM

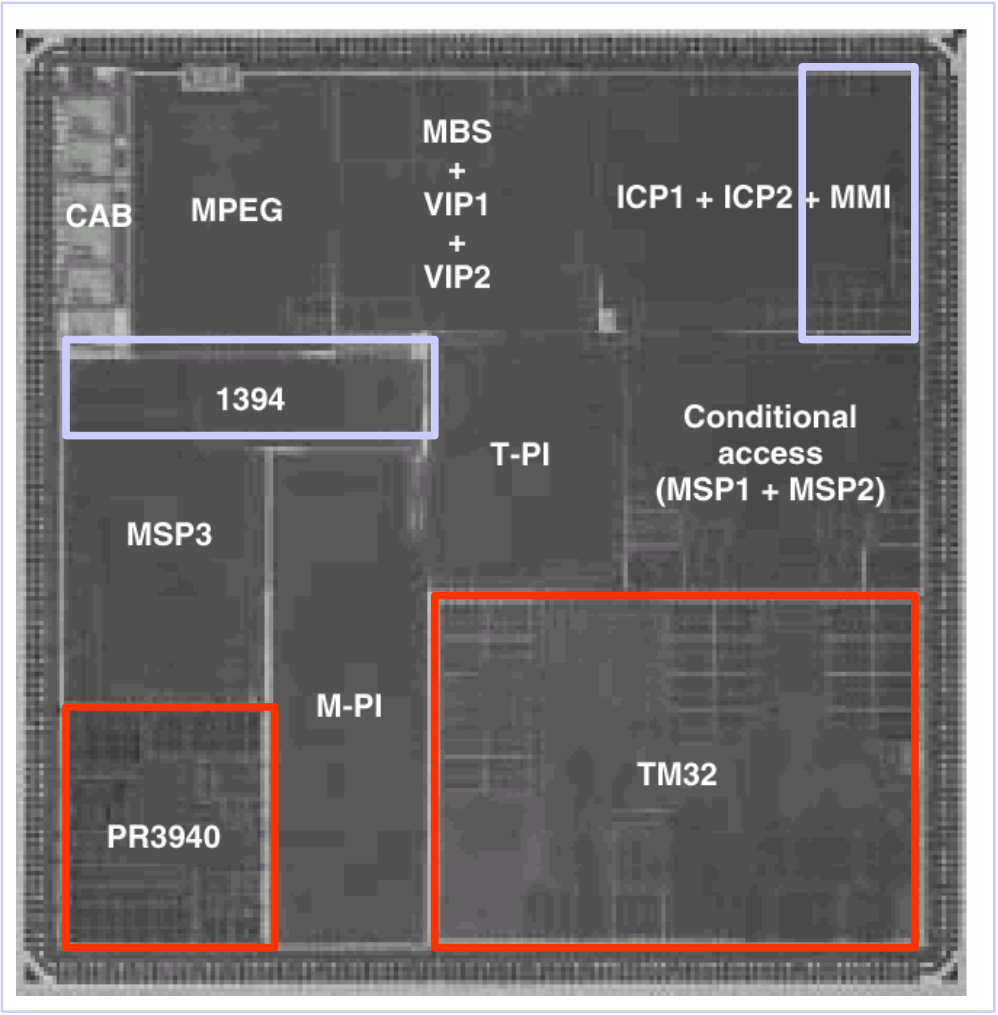
Specialized units:

- Hash unit for table lookup
- Proprietary IX Bus interface

Clock frequency	200 MHz
Power	5 W

~35% of die area for I/O interfaces/functions

Philips Nexperia (Viper)



Multimedia processor platform
e.g. Set Top Boxes

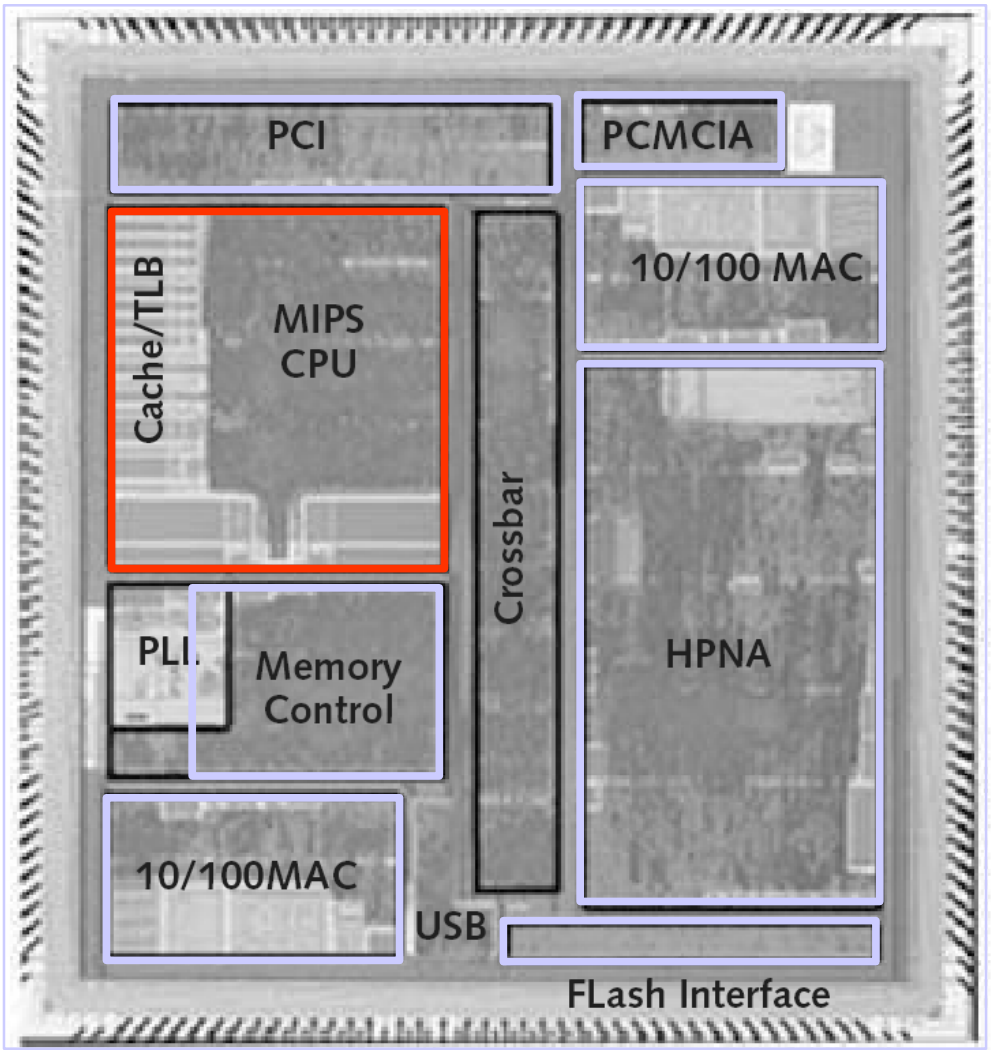
MIPS 32b RISC CPU (PR3940)
for general purpose Tasks
Trimedia VLIW CPU (TM32)
for media processing tasks

Specialized units:
- MPEG2 video decoder/ MPEG processor
- 2D rendering engine
- Audio I/O
- USB, SSI, GPI, MMI, DMA
- Interrupt controller

Clock frequency	200/150 MHz
Power	4.5 W @ 1.8V

50 heterogeneous peripherals
82 Clock domains

Broadcom BCM 4710



Access Network Processor for Home gateway application, Wireless access point

R3000 MIPS Core system
w/ on-chip caches/TLB, Timer/Interrupt
Linux/Vxworks kernel

Integrated Peripherals:

- PCI 2.2 Host, PCMCIA Interface
- Flash
- 2x 10/100 Eth. MAC
- HPNA 2.0 base band signal processing
- USB 1.1 Host/device
- UART/Modem/IR, I^2S

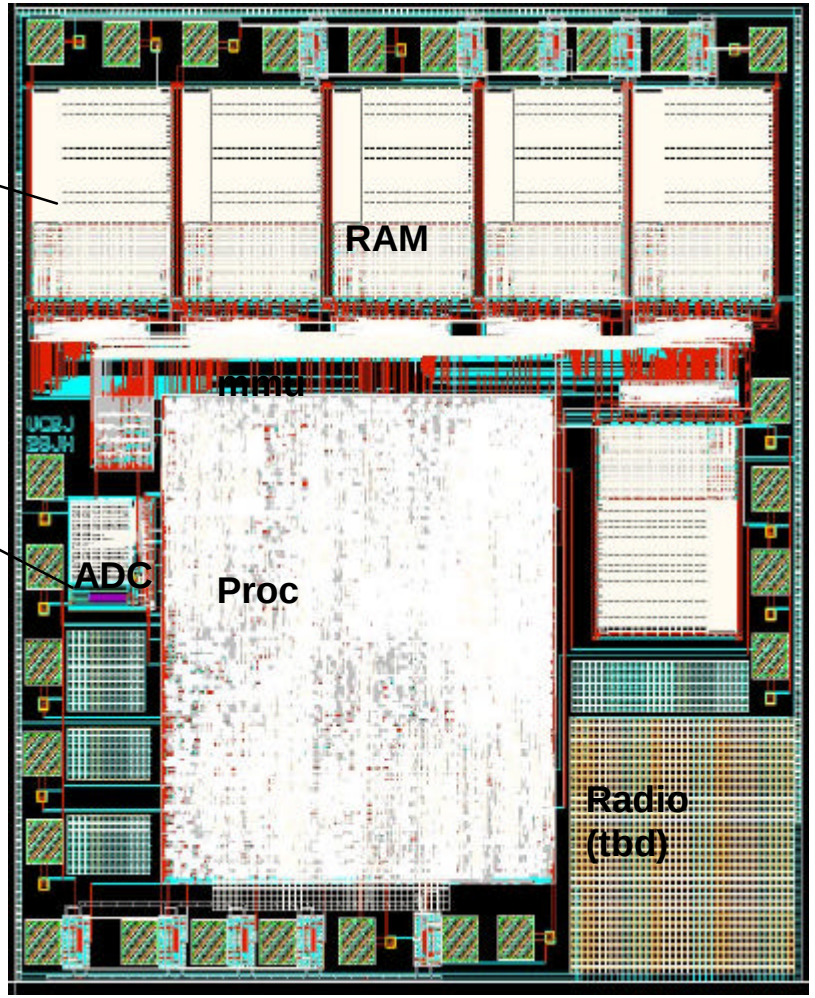
Clock frequency	125 MHz
Power	??? W @ 1.8 V

MIPS system < 25% of die
25-30% Home PNA

What integration buys...

- 3K RAM = 1.5 mm^2
- CPU Core = 1mm^2
- RF COMM stack = .5mm^2
- RADIO = .25 mm^2
- ADC 1/64 mm^2
- I/O PADS

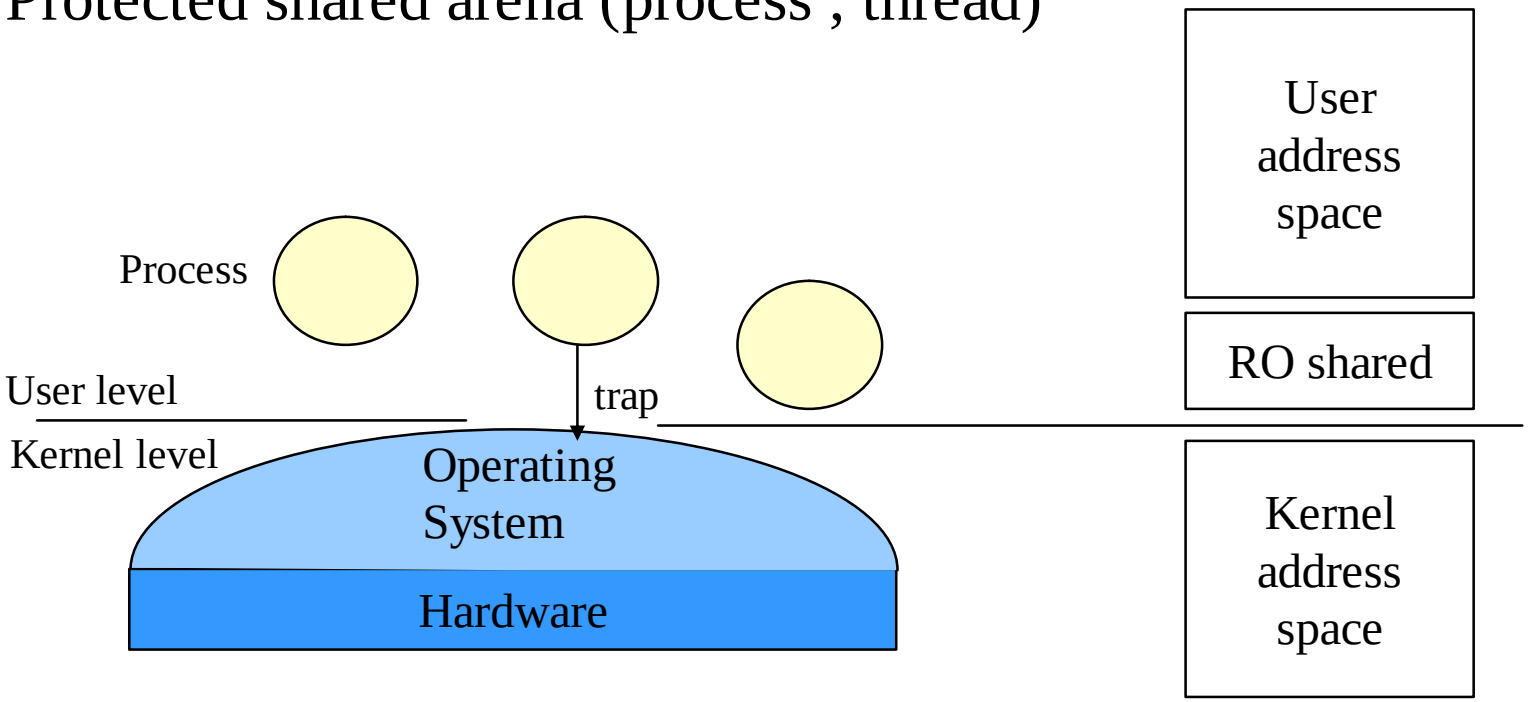
- Expected sleep: 1 uW
  - 400+ years on AA
- 4 Mhz < 1 mW
- Radio:
  - .5mm2, -90dBm receive sensitivity
  - 1 mW power at 100Kbps
- ADC:
  - 20 pJ/sample
  - 10 Ksamps/second = .2 uW.



(David Culler, 2003)

Operating environment

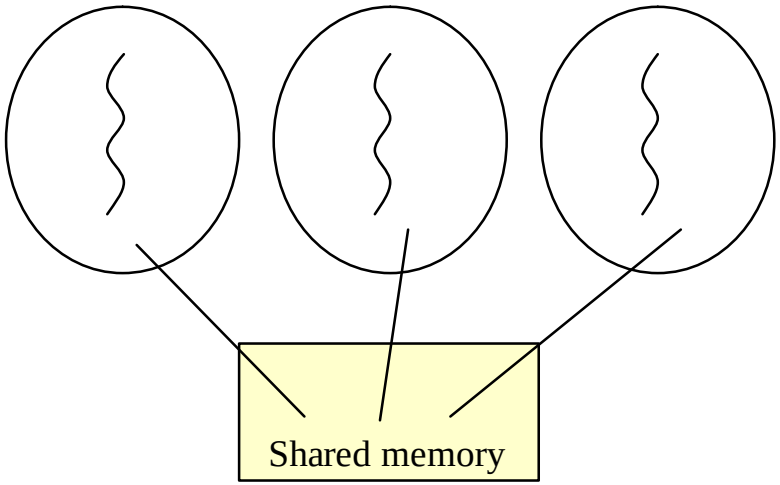
- Traditional monolithic systems
- Processes, trap mechanism, isolation, virtualization
- Hardware control: CPU, memory, devices
- Protected shared arena (process , thread)



Program and Process / Thread Model

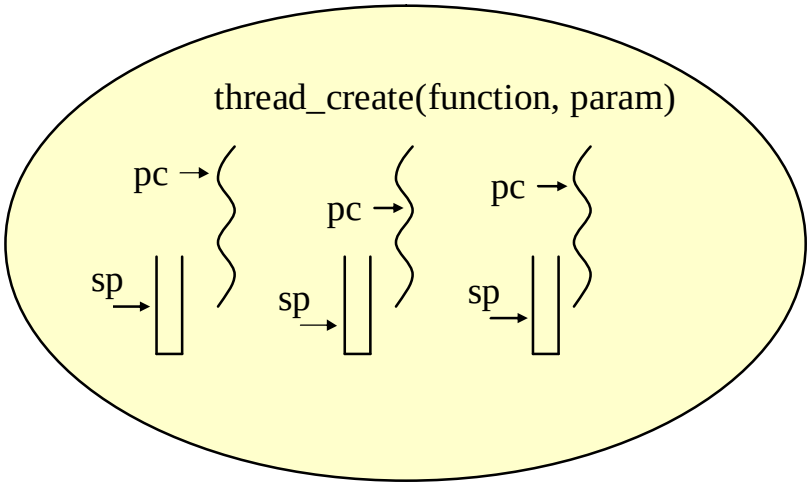
fork();

exec(program, parameters);



Unix process

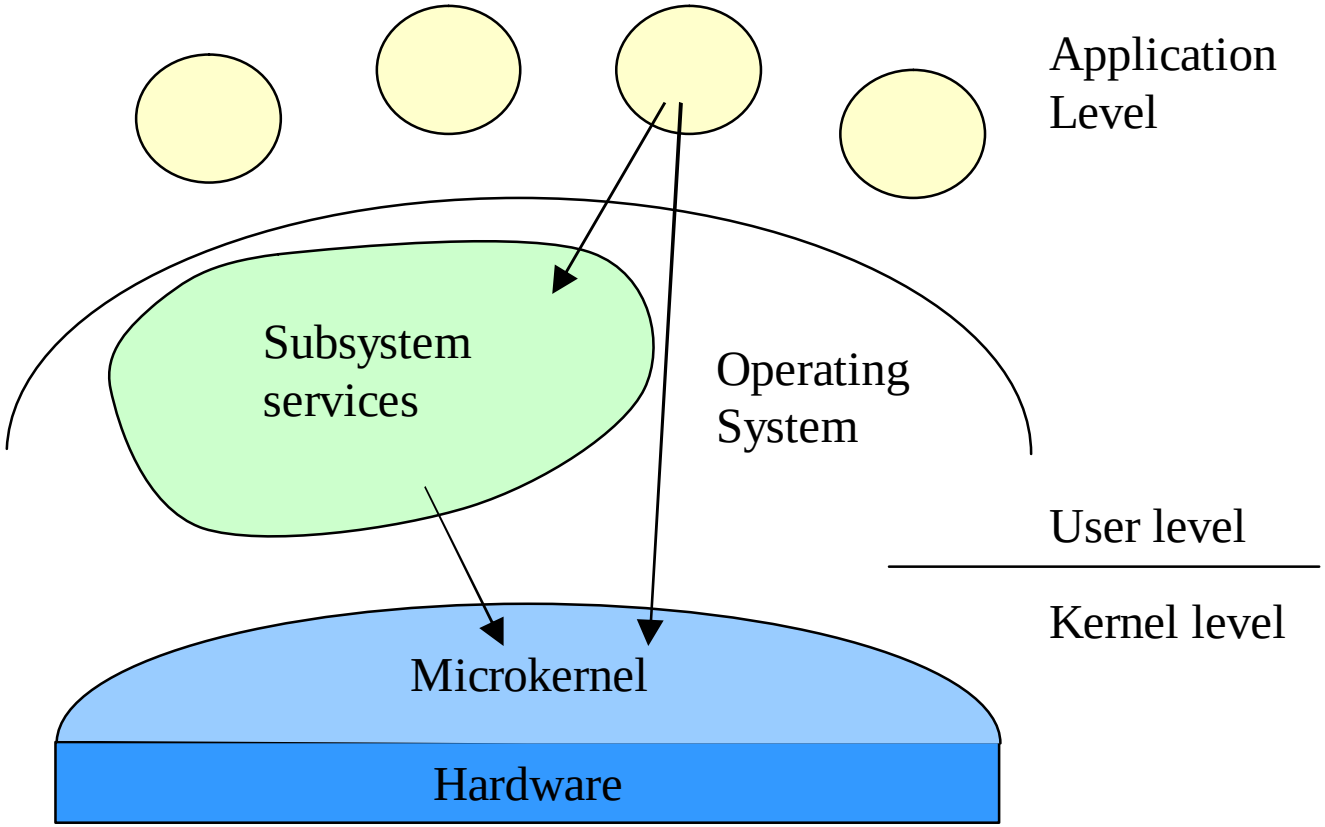
(can setup explicit shared memory model)



Threads: implicit shared memory

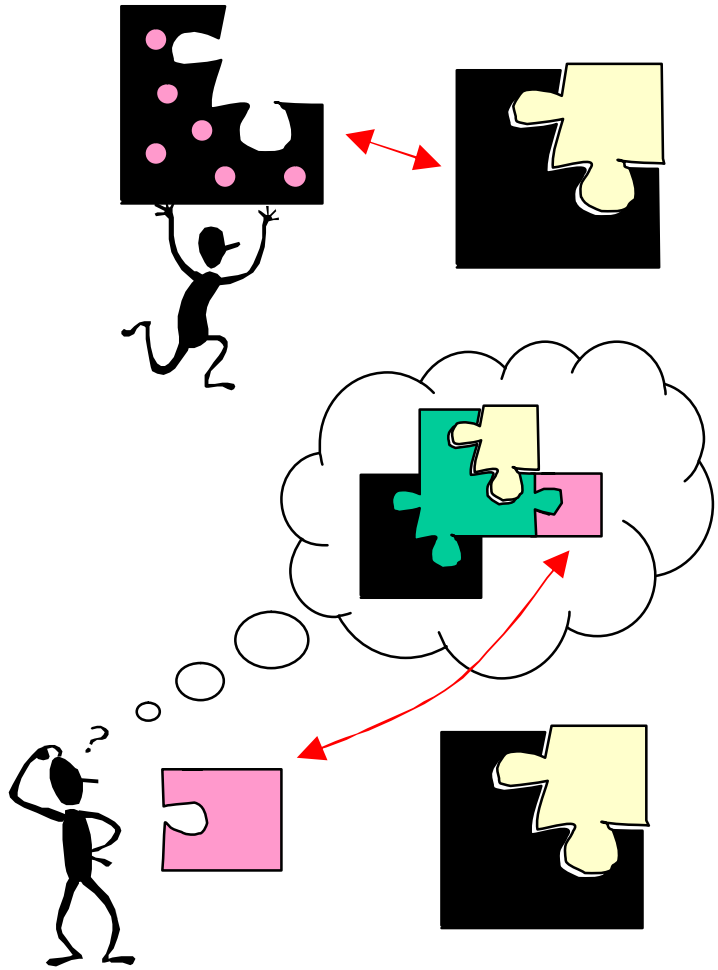
Microkernel technology

- Thin platform to subsystem and applications support
- Management of tasks, threads , memory, inter process communication



Flexible Composition

- Improve performance and efficiency through specialization
- Enable seamless evolution of services through extensibility
- Do it dynamically (DLL and .so)
- Loaders: new capabilities



UNIX emulation

- Implement UNIX abstractions on top of microkernel
- Needed for microkernel project success
- Servers
  - Multiple servers (distributed, maintain global status)
  - Single server (overhead, multithreaded)
- Libraries (UNIX interface)
  - System calls can be redirected as messages
- Binary emulation
  - Trap
  - Address space

Extensible OS

- Coarse granularity
  - Mach “in-kernel tasks”
- Linux Modules: device drivers, filesystems, network protocols, etc
- Fine granularity
  - Application level code (upcalls, grafts)
  - Kernel level component (streams)
  - Application specific kernel code (SPIN)
- Protection, verification, correctness

Virtual Machines

- Very old concept
  - partitioning the machine (IBM, 1960)
- Full system emulation
  - Simics (Virtutech) , Bochs
- Virtual workstation, data center
  - VMware flavors
- Java VM
  - platform independent programming language
- Application specific
  - Cross-over Office (Codeweavers)

New Hardware Interfaces

- Fast System Calls: SYSENTER (Pentium), EPC (Itanium)
- Fast address space and context switch
- User Level Drivers
- Speculation recovery
- Multiple size and protection in page tables
- 64 bit efficient execution
- Dynamic optimization/translation

Linux strikes back (1)

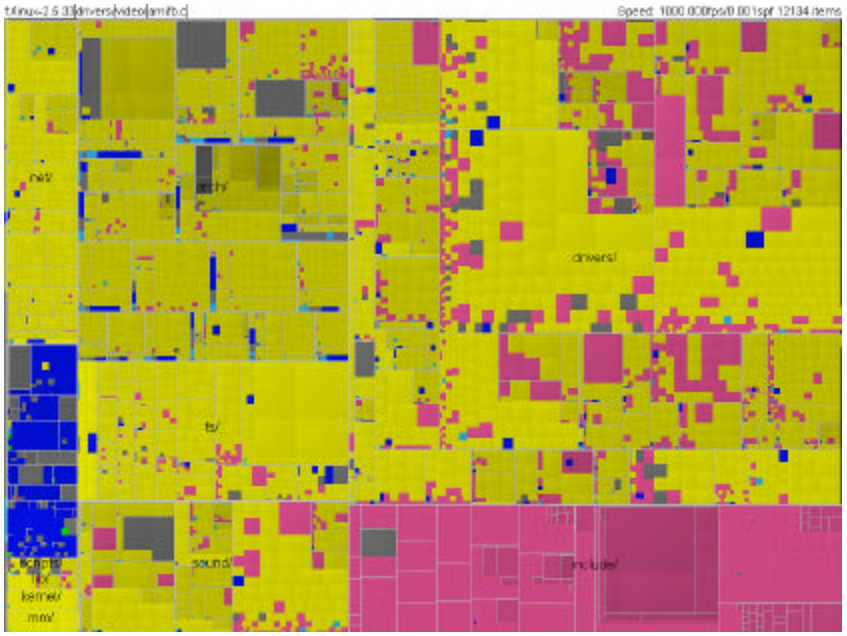
- Linux for supercomputers
  - Large address spaces
  - Large file systems, journaling
  - Multithreading, hyperthreading
  - Clusters, NUMA
  - AMD Opteron 64bits, Itanium IA64
- Linux for large data centers
  - Virtual machines, sandboxing, checkpointing
  - Projects: UML, KML

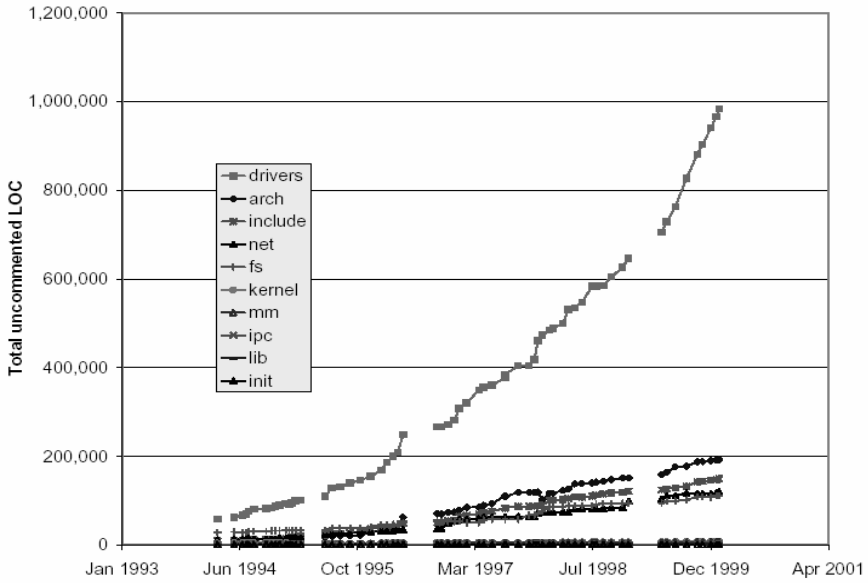
Linux strikes back (2)

- Linux for embedded, real time
  - Small size, better configuration
  - Preemptive kernel
  - No MMU machines
  - Suspend to disk
  - Dynamic modules
  - Human interface layer, wireless, multimedia
  - No keyboard, no display, no mouse
  - http://www.linuxdevices.com/

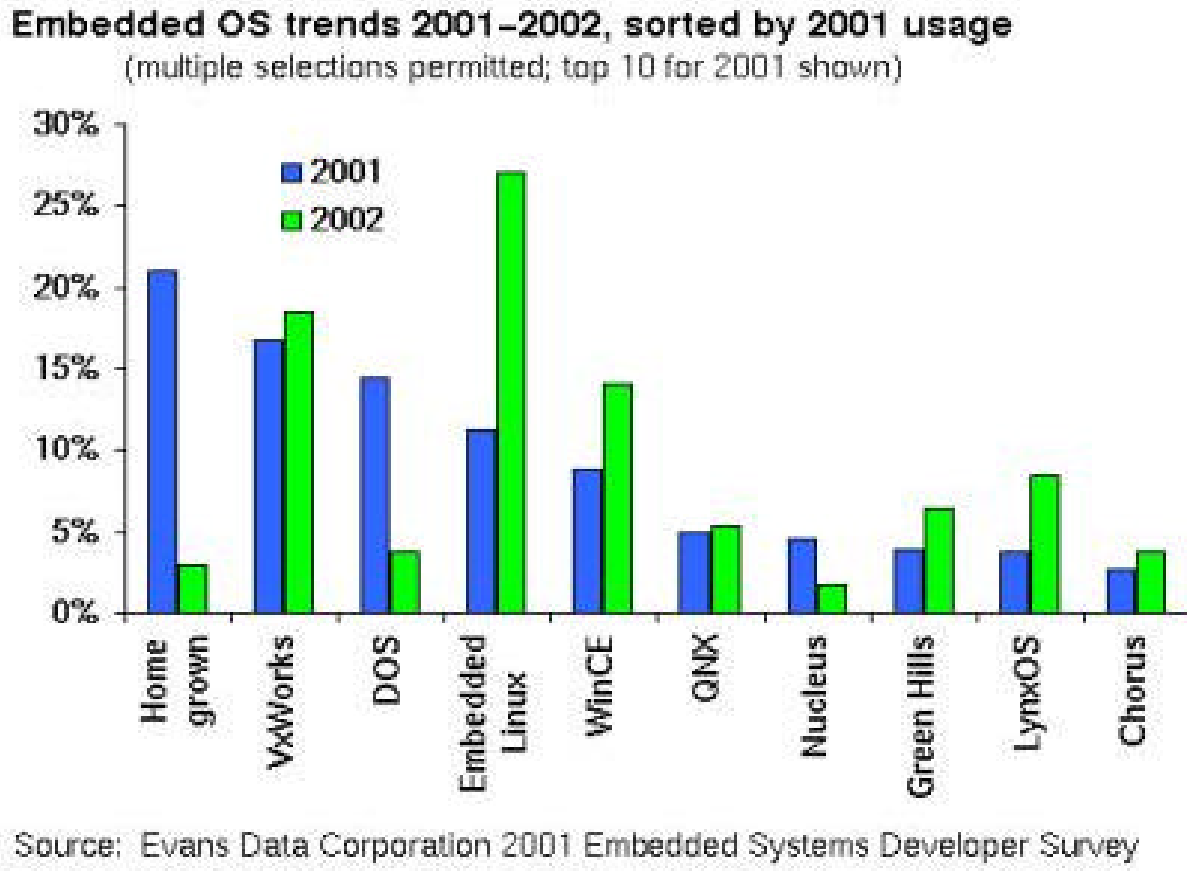
Linux source code size evolution

- Kernel source is continually growing
  - New drivers (exponentially), new architectures
  - New features and policies in the core kernel and existing drivers
- Need new tools to manage configurations
- Customization should be done automatically





Embedded OS trends

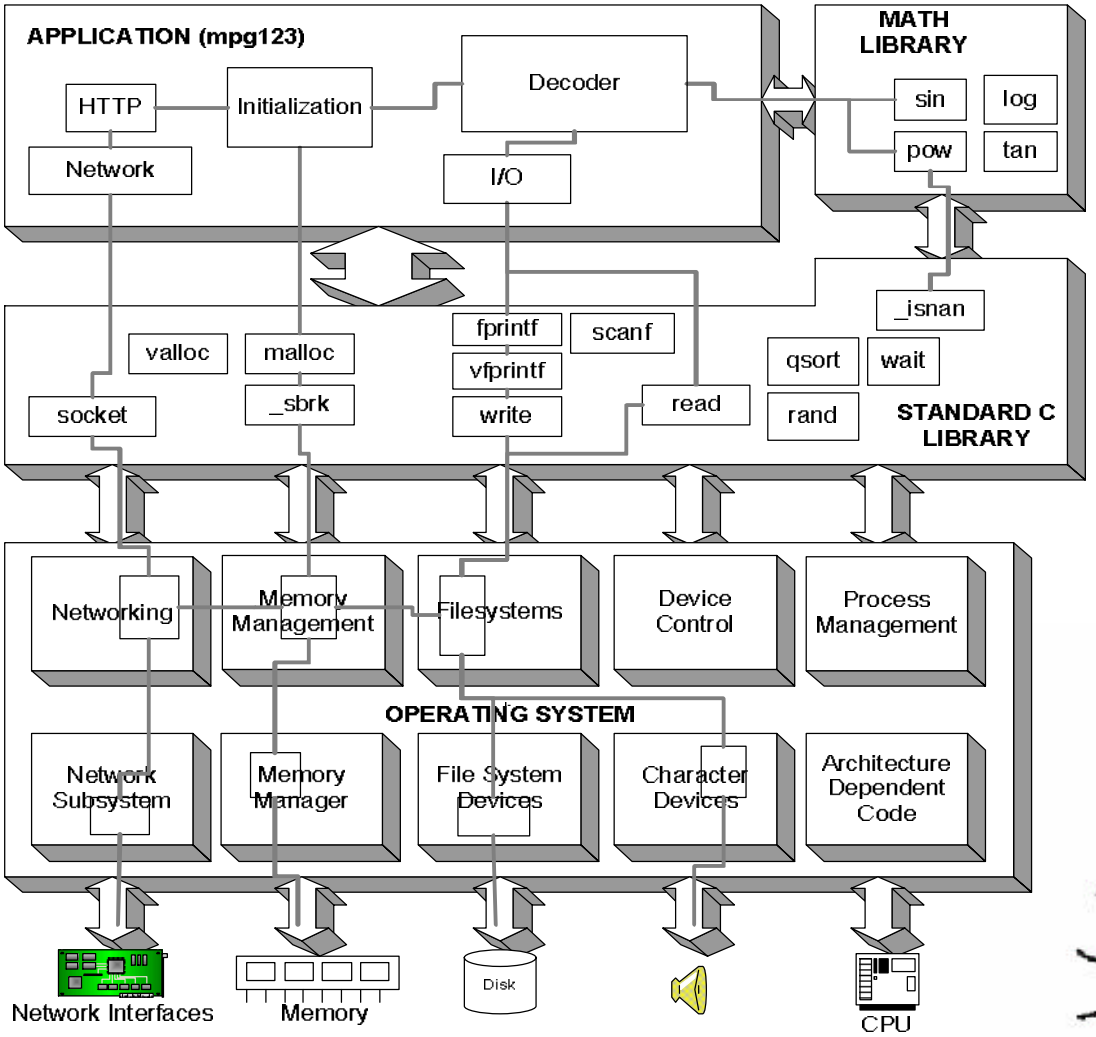


Appliances evolution

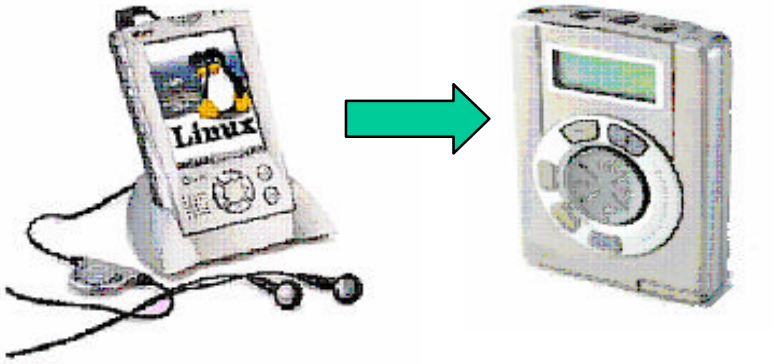
	2003	2004	2005	2006	2007	2008
Total Handset Shipments	482.3	523.8	567	610.3	652.7	687.1
Full-Feature Handset Shipments	10.9	30.4	69.2	131.6	221.8	291.8
Percentage Full-Feature Handsets	2.3%	5.8%	12.2%	21.6%	34.0%	42.5%

Source: Zelos Group, Inc.

MP3 / Linux Usage Characterization

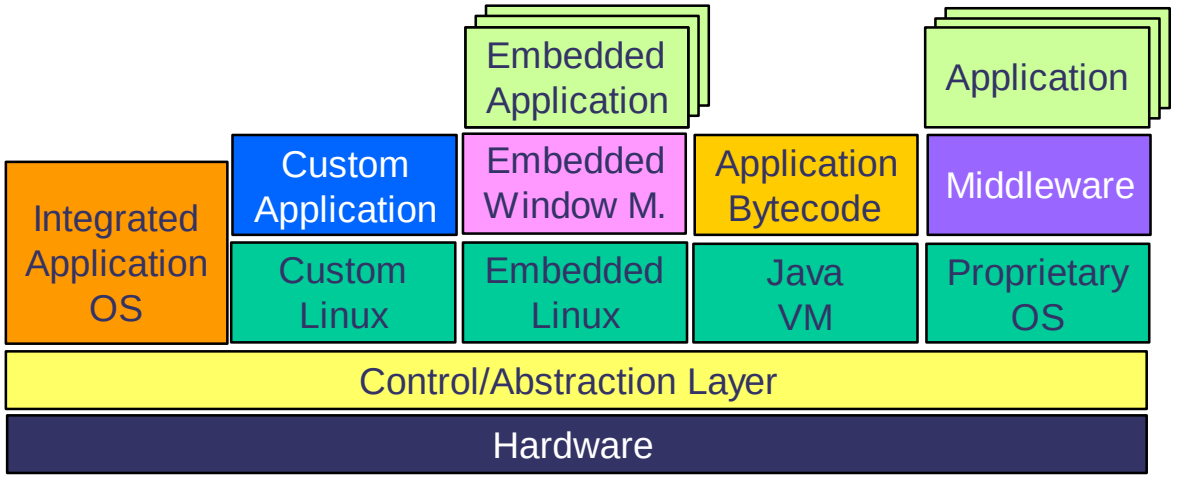


- Customization based on hard/soft specifications
- Control flow profiling
- Multimedia and communication software slices



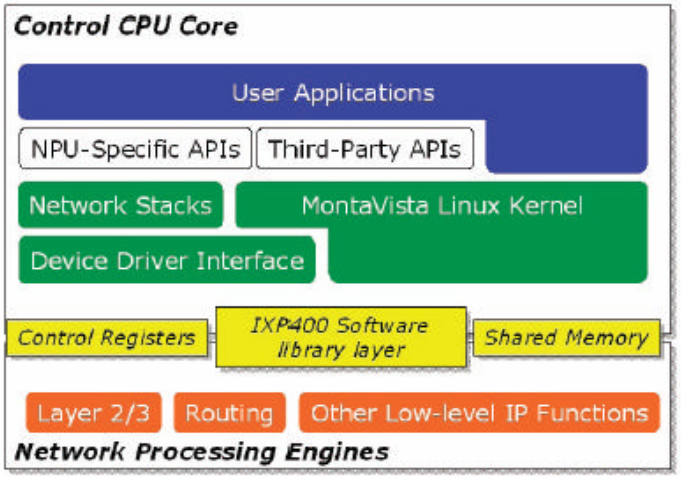
Embedded dynamic platform

- Thin control/abstraction software layer that loads and sequences executions of operating systems and virtual machines
- On demand downloading / swapping OS and applications
- Highly dynamic modification or complete change in the fly

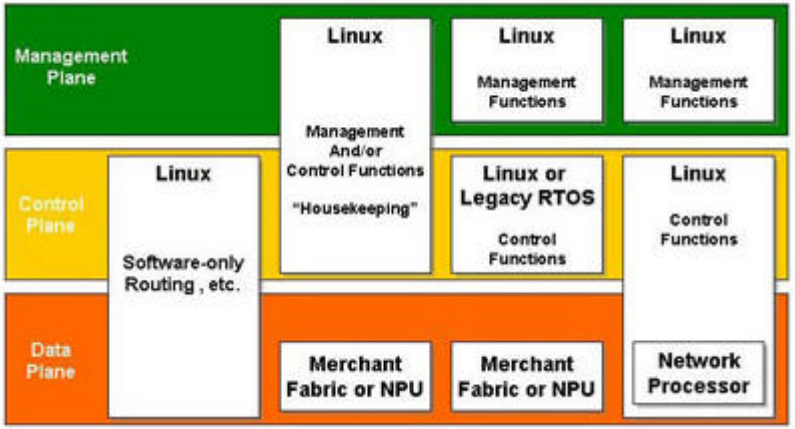


Linux in Network Processors

- Interface OS / low level hardware layers
- Network processors API or shared memory



- Linux configurations for a router



Power Management

Static Power Management	Dynamic Power Management
• Design time (off-line) • Hard+Soft	• Runtime (on-line) • Hard+Soft

• Low-Level approach (CPU-Level)
  • Target ✍ CPU
  • Cycle level
  • Instruction-Level
	• CPU-Level
	  • Dynamic voltage scaling

	• System-Level • Policies: Time-Out, Predictive, stochastic

• System-Level
  • Target ✍ System
  • State-Level models
  • Application & OS-Level
  • Complete system Level
	• Cluster System-Level

OS when Every Joule is Precious

- Allocating resources taking energy efficiency into account
- Fair allocation of battery resources rather than CPU resources
- Operating system Functionality:
  - Disk Scheduling: spin down policies
  - Security: adaptive cryptographic policy
  - CPU scheduling: voltage scaling, idle power modes
  - Application/OS: interaction for power management
  - Memory allocation: placement, switch energy modes
  - Resource Protection/Allocation: fair distribution, critical resources
  - Communication: adaptive network polling, routing, servers

Conclusions

- Microkernel and extensibility concepts influence current and future OS's
- Standard interfaces are more and more needed for rapid software development
- Tools for mptimization, compile-time and run-time specialization
- Power management is crucial
- Current trend: Linux everywhere