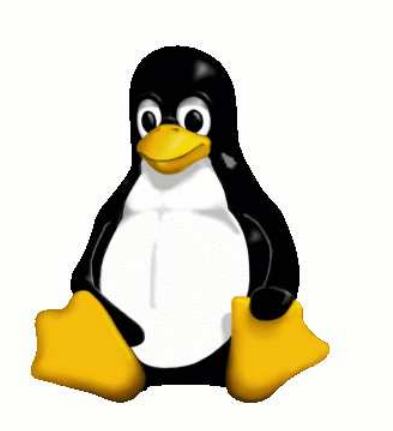
The Future of Embedded Linux

C3Expo, June 2005

Stephen A. Edwards

Columbia University



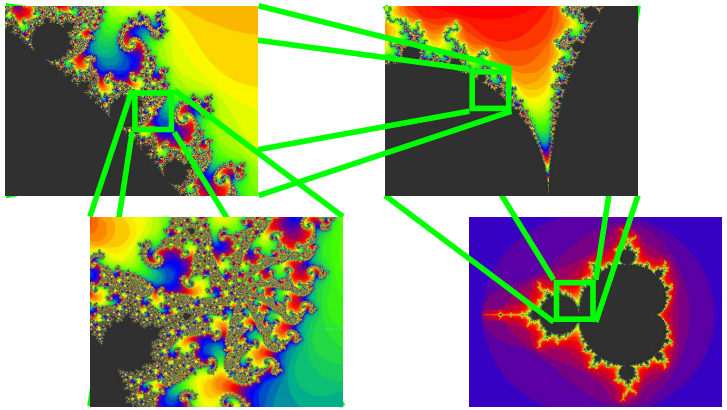
Developing Embedded Systems

From The Three Stooges' *A-Plumbing We Will Go*, Columbia Pictures, 1940

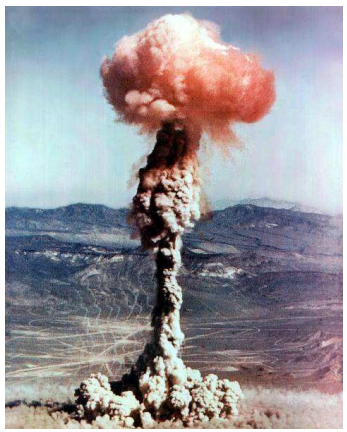
In a word, plumbing.

Embedded System Challenges

Real-time



Complexity



Power

Cost

System Complexity Growing!

Size of Typical Embedded System

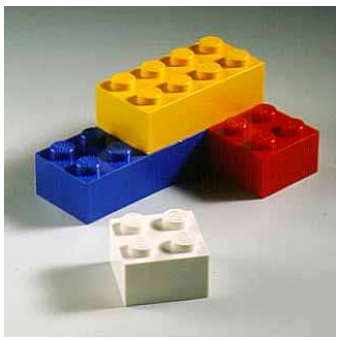
1985	13 kLOC	
1989	21 kLOC	↓ 44 % per year
1998	1 MLOC	
2000	2 MLOC	
2008	16 MLOC	≈ Windows NT 4.0
2010	32 MLOC	≈ Windows 2000

Source: “ESP: A 10-Year Retrospective,” Embedded Systems Programming, November 1998

How do you write this much code?

Simple: you don't.

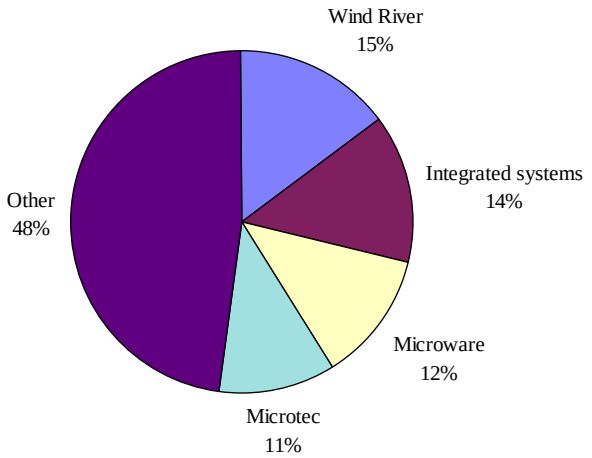
Instead, you grab it and integrate it.



Less like Legos

More like manufacturing cars

Historical Industry Trends



Embedded OS market was fragmented

Didn't matter: amount of code was small and always custom-tailored

1995 data, DMG report.

Wind River acquired ISI in 2000

Today, interoperability is everything.

Consolidating: down to only a few players

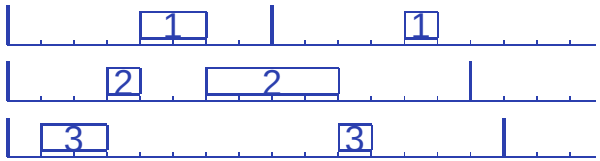
Wind River VxWorks

Microsoft Windows CE

Embedded Linux

A Technical Challenge

Most embedded operating systems are “real-time”



Fixed-priority preemptive scheduling is standard

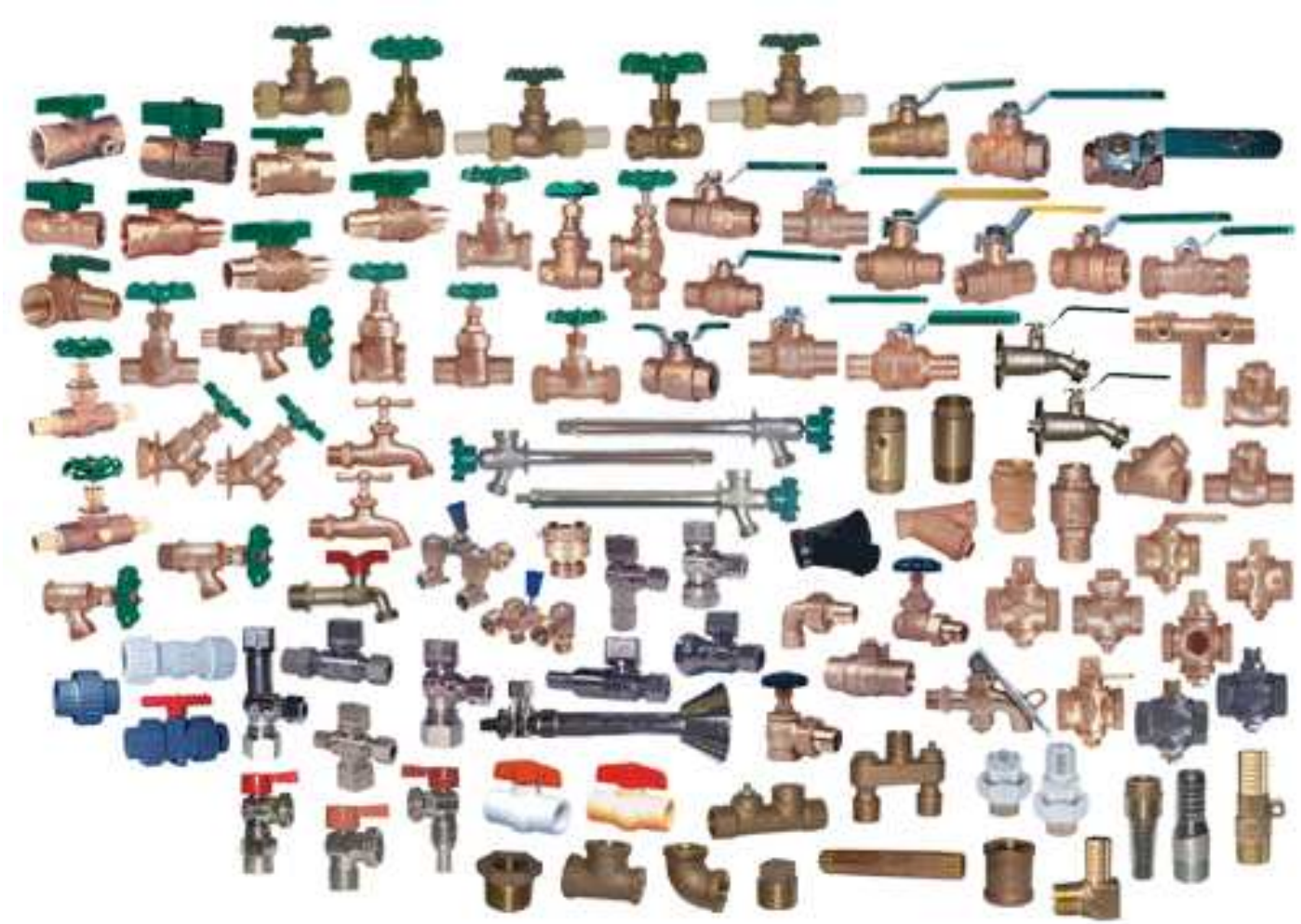
Traditional operating system schedulers strive for fairness

The Linux Kernel isn’t well-suited to real-time applications!

Solutions

- RTLinux, Adeos: Run the Linux kernel as a task on an RTOS
- RTAI: run real-time tasks as threads in an uninterruptable Linux process
- KURT, Linux/RT: patch the kernel to add real-time scheduling policies

Embedded Linux



A nice solution to the plumbing problem