

# What Makes an Embedded Application Tick?

# How Many Embedded Systems?

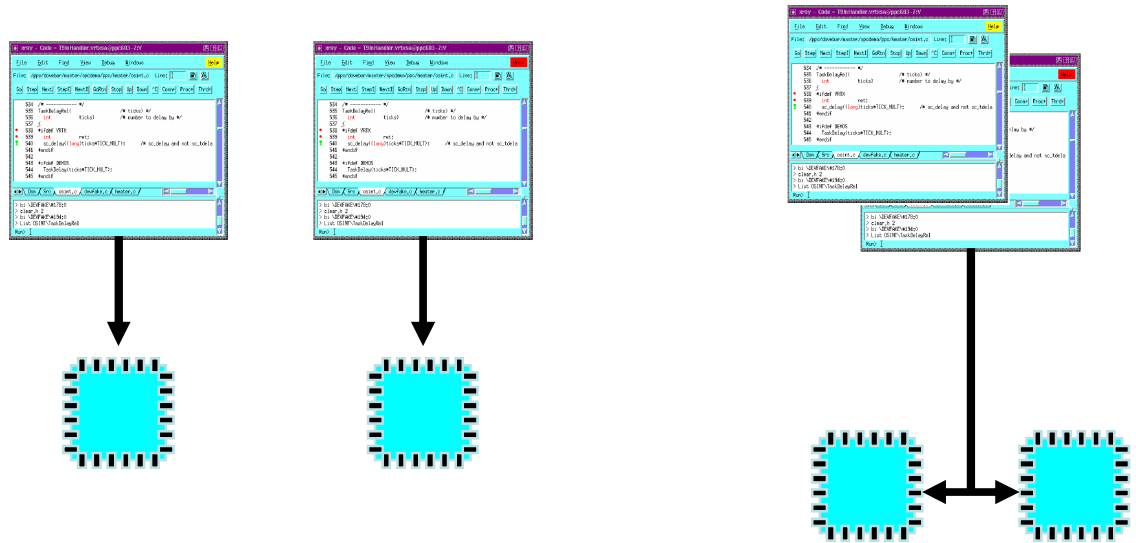
- In the average American household:  
around 40 microprocessors; not counting:
  - PCs, which contribute another 5-10 each
  - cars, which typically contain a few dozen
- Will rise 100X over next couple of decades
- Most people don't know what "embedded" means

# Development Challenges

- Multiple processors
- Limited memory
- User interface

# Multiple Processors

- Key challenge is debugging
- Need multi-core support



# Limited Memory

- May not be small, but probably not extendable
- Cost and power consumption issues
- Understand optimization
- C++ requires skill and the right tools

# User Interface

- Critically important
- Mainly implemented in software
- Ideal steps:
  - design the hardware
  - make some prototypes
  - implement the software [UI]
  - try the device with the UI and refine/re-implement as necessary

# UI Development

- Hardware not available
- Design may not even be complete
- Need to use prototyping/simulation technology to model on host computer

# Re-usable Software

- Used to be a “start from scratch” approach
- Now software is too big and too complex
- Nobody can have all the expertise
- Time to market pressure drives short development cycles
- Reuse widely accepted in hardware design – same needed in software

# Software Components

## Examples:

- Real-time operating system
- File system
- USB
- Graphics
- Networking

# Real Time Operating Systems

- 200 products on the market
- Still common to implement in-house
- Need to understand selection criteria

# RTOS Selection Factors

- Hard Real Time
- Royalty Free
- Support
- Tools
- Ease of Use
- Networking
- Broad CPU Support

# RTOS Standards

- Many proprietary
- Some standards available:
  - OSEK/VDX [automotive/transportation]
  - $\mu$ ITRON [Japan]
  - POSIX [migration from UNIX host]

# File System

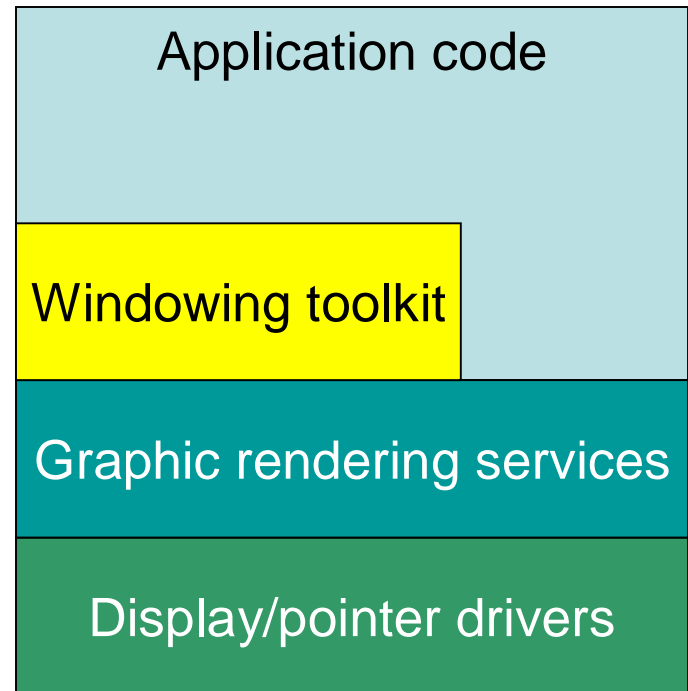
- Persistent storage
- Magnetic, optical or NVRAM [flash]
- Standards-based approach best
  - interoperability issues
  - data transfer
- MS-DOS the easiest standard to adopt

# USB

- Implementation is very complex
  - hence ease of use
- Smart part is software not hardware
- Support already done for host computer
- Needed for embedded devices
- USB On-The-Go becoming available

# Graphics

- LCD panel may have 2 functions
  - graphic output
  - user interface
- Doing graphics seems easy, but can quickly become complex
  - simplified with graphics library
- GUI is typically another library on top of regular graphics



# Networking

- At least a third of embedded system are connected
- May be wired or wireless
- TCP/IP is quite straightforward to program
  - additional applications and protocols are challenging

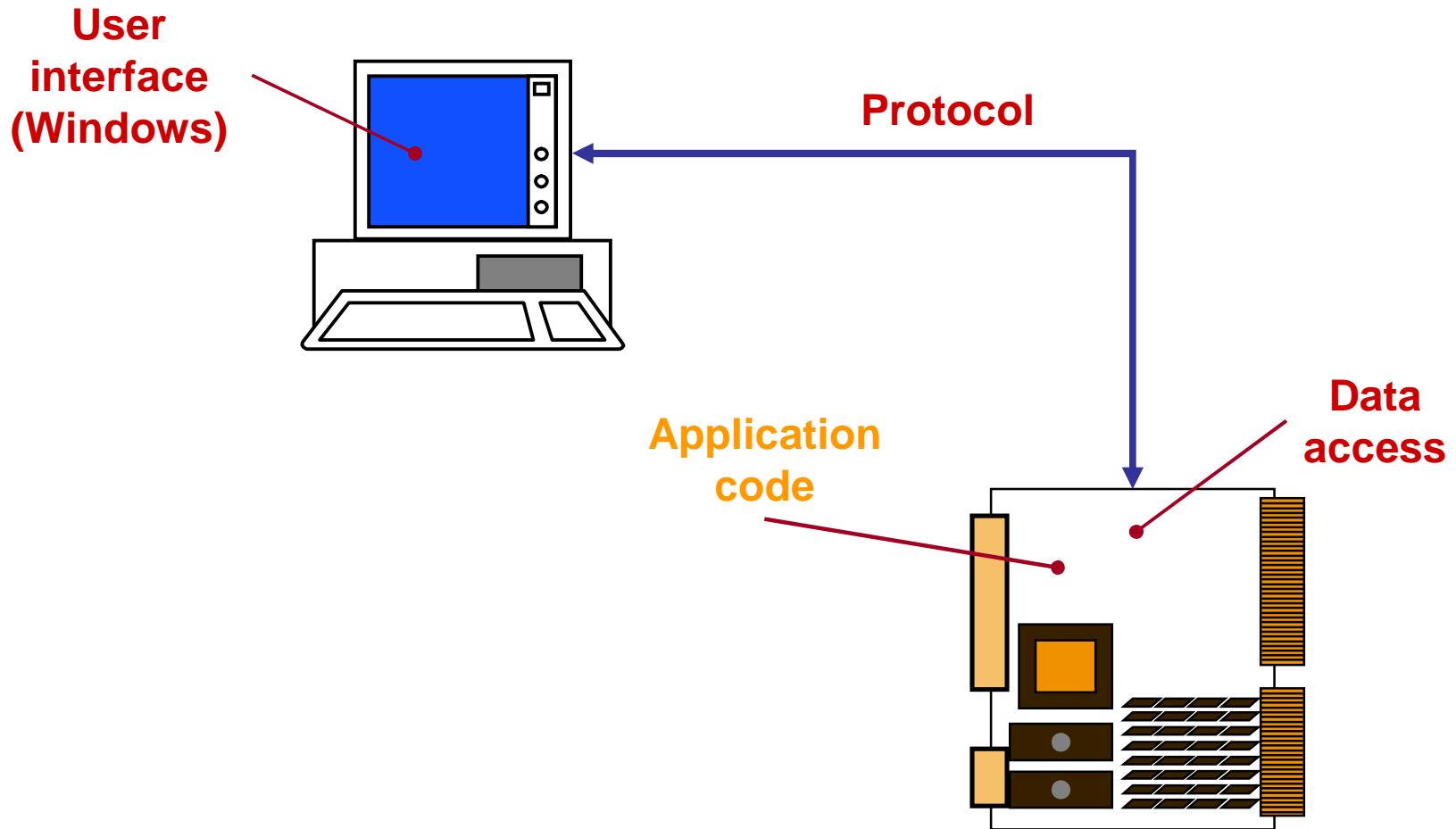
# IPv6

- IPv6 is the next version of the Internet Protocol
  - Rules that define communication over networks
  - Current version is IPv4
- IPv6 solves a number of problems with IPv4
  - Virtually Unlimited Address Space
  - Robust Addressing Architecture
  - Lower Maintenance Costs
  - Increased Flexibility

# IPv6 Addressing

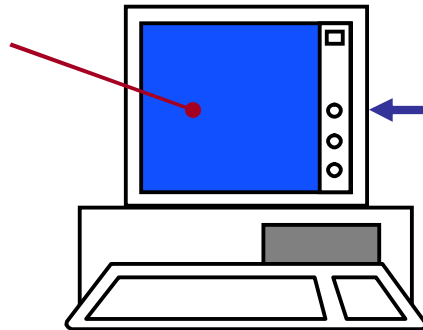
- Standard Format
  - 3ffe:2900:0102:0001:0000:0000:0000:0002
- Leading Zeros removed
  - 3ffe:2900:102:1:0:0:0:2
- Double Colon Notation
  - 3ffe:2900:102:1::2

# Who Needs a Web Server?



# Web Server Solution

Any Web browser  
(any OS,  
any platform)



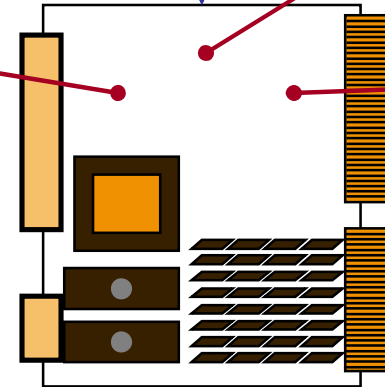
HTTP



Application code

WebServ

Simple data access



# SNMP vs Web server

- SNMP
  - Complex security features built in
  - MIBs can be added easily
  - Browser format fixed
  - Browsers cost money
- WebServ
  - SSL optional
  - HTML and coding straightforward
  - Interface can be customized to the application
  - Browsers free