16.070 Introduction to Computers and Programming

May 2 **Recitation 12** **Spring 2002**

Topics:

- Project Q&A
- Real-time overview
- Multitasking

Final Project:

Any Questions?

Real-time overview

What is a real-time system?

- Correct data – computation must be correct or the system fails
- On-time data – data must be available at certain deadlines or the system fails.
- Continuous operation – tasks are run repeatedly. Contrast this to the simple programs you wrote at the beginning of the term that did one set of calculations and then exited.
- Can you think of a computer system that is not at least soft real-time?

What is an embedded system?

- Integral part of hardware – the hardware won't work without the software
- Specific to the hardware – the software is written to do a specific task on a specific machine. You can't take the software in a microwave and have it run on a calculator.

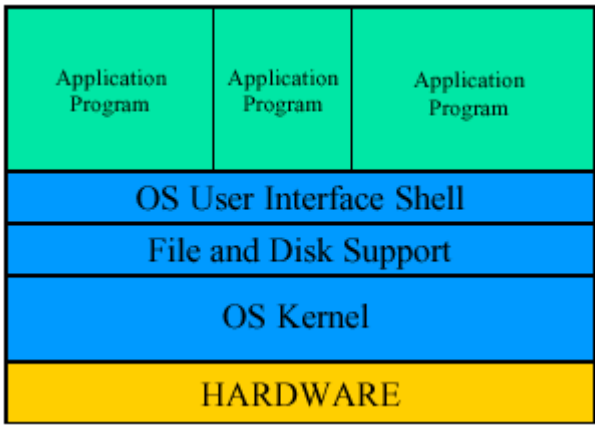


Figure 1 - Layers of a Real-time System

Hardware: The part of a computer system that can be kicked – M. Barr

- CPU
- Memory
- I/O to Sensor and Actuators
- Testing tools

Operating System Kernel – most basic part of an operating system

- Task Scheduler
- Task Dispatcher
- Intertask Communication

Kernel Design Strategies

Two examples…

Polled Loop

- Single Task
- Checks sensor/interrupt/input repeatedly until event occurs
- Pros
  - Simple
  - Response time easy to determine
- Cons
  - Simple
  - Wastes CPU time
  - Can fail if several events happen close together (burst)
- Example: Processor for an antenna “listens” for a contact. Starts data recorder when contact received.

Interrupt Driven

- Hardware signal changes the value in an “interrupt register”
- Interrupt register checked during each Fetch-Execute cycle
- If an interrupt has occurred, switch to the interrupt-handling task.
- More than one interrupt can be used - Priority
- Pros
  - CPU can do other things while waiting for an interrupt

    - Can handle more than one task
- Cons
    - More complex
    - Timing
    - Priority Conflicts
- Example: Data processing system on a satellite
    - Incoming data sets an interrupt that starts Data Collection task.
    - Data Collection fires an interrupt every 10 memory blocks to start Data Processing task.
    - Data Processing task fires an interrupt when data is ready to be sent to Earth.
    - Interrupt servicing task sends processed data to Earth.

Multitasking

Context Switching - What happens when you need to change tasks.

- Save program counter
- Save registers
- Load new program counter
- Load registers

Cyclic Executive

- Run a fixed number of tasks in a fixed sequence

Round-Robin

- Tasks executed in the order they arrive
- Each task is assigned a fixed time slice
- Context switched at the end of the time slice
- AND context switched when a task finishes
- Example 1

Preemptive Priority

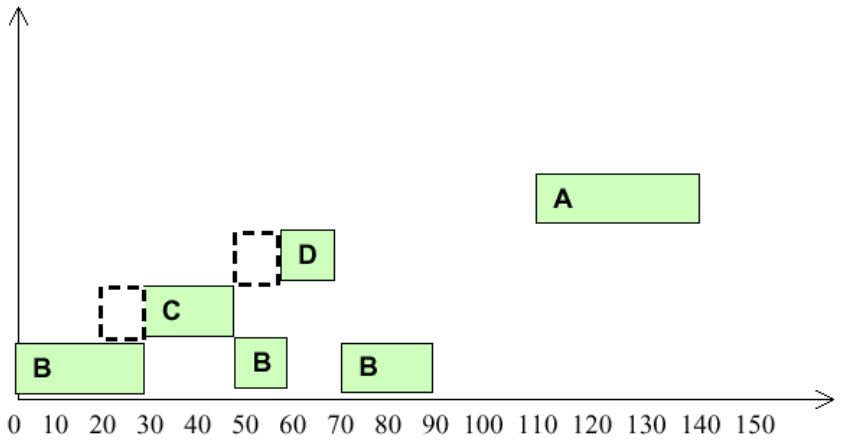
- Tasks executed in order or priority
- Higher priority tasks interrupt lower priority tasks
- Context switched when a higher priority task arrives
- AND context switched when a task finishes
- Example 2

Example 1: Round Robin

Imagine a round-robin system with 4 tasks. Context is switched every 30 ms **and** when a task ends. The tasks are cycled through in the order of their appearance. Using the information provided, fill in the table.

Task	Arrives at t= (msec)	Time Needed (msec)
Task A	**110**	**40**
Task B	**0**	**60**
Task C	**20**	**20**
Task D	**50**	**10**

Time (msec)	Task Running	Task(s) Pending	Explanation
0	B		B arrives
10	B		
20	B	C	C arrives
30	C	B	Context switch, C runs
40			
50			
60			
70			
80			
90			
100			
110			
120			
130			
140	A		A finishes



Example 2: Preemptive Priority

Now take the same set of tasks, but assign them the following priorities. Context is switched when a higher priority task arrives, and when a task finishes. Using the information provided, Fill in the table below and draw a timing diagram.

Task	Arrives at t= (msec)	Time Needed (msec)	Priority
Task A	**110**	**40**	**1**
Task B	**0**	**60**	**10**
Task C	**20**	**20**	**5**
Task D	**50**	**10**	**20**

Time (msec)	Task Running	Task(s) Pending	Explanation
0	B		B arrives
10	B		
20			
30			
40			
50			
60			
70			
80			
90			
100			
110			
120			
130			
140	A		A finishes