

Chips with power of whole computer systems now in many applications:

- Mobile Phones
- PDA's
- Smart Cards
- On-board controllers of HW

Characterisation of those systems:

- Fewer resources available: memory, storage space
- Often real-time applications necessary (on-board controllers)

Not so much of a problem in general: OS's designed for this case

Only issue: potentially missing MMU
⇒ virtual memory and protection of processes against each other not implementable
Also paging not available

Real-time Operating Systems

Have two different kinds of real-time

1.) **Hard real-time**: completion required within a guaranteed amount of time cannot be met by normal time-sharing systems; needs dedicated HW and adaptations to SW

2.) **soft real-time**: critical processes receive priority. Requires

- **pre-emptive priority scheduling** (plus sufficient resources to avoid starvation)
- **short dispatch latency** (time between arrival of process and start of execution)
Problem: context switch normally only after syscall-completion or when I/O takes place
way out: make **kernel pre-emptible** (e.g., Solaris 2)
- **Priority inversion**: increase priority of process if resources required by high-priority process