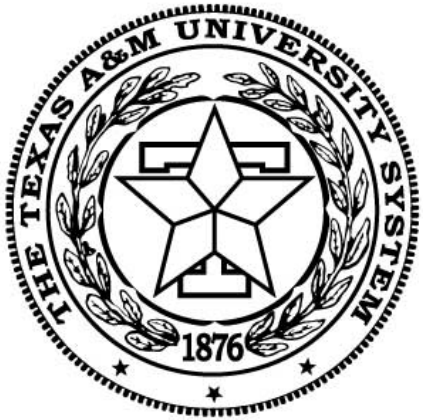
ECEN 449 – Microprocessor System Design



Processor Scheduling

Objectives of this Lecture Unit

- Processor Scheduling
  - Impact on getting things done in time

Processor Scheduling and tasks

- Processor is shared among many tasks
  - Time sharing
- Scheduling determines the order in which tasks are executed
- Sometimes, you may want tasks to finish in time
  - Every 20ms, want to check the temperature
  - When tasks take longer, may have important consequences
- How does normal processor scheduling work?
- Are there different ways to do this?

Processor scheduling

- Round Robin scheduling
- Run a task for a quantum of time, go to the next task
- Every task gets a fair share of the processor
- Every task runs on the processor within n*t
  - Number of tasks = n, quantum time = t
- Worst-case between two arrivals of the same task <= 2*n*t
  - Task arrives just after its turn is gone
- As number of tasks grows, waiting time for service grows
  - May not work if you want to check the temperature in 20ms

Processor Scheduling

- Weighted Round-robin
- Give larger weights to important tasks
  - Weight tells how many times the task can be scheduled in each round
  - Higher the weight, higher the service, lower the waiting time
- Time for a round still depends on the number of tasks
  - Possible to limit the waiting time for a task to predictable value by adjusting weights
  - Gets complicated when many tasks have “time deadlines”.

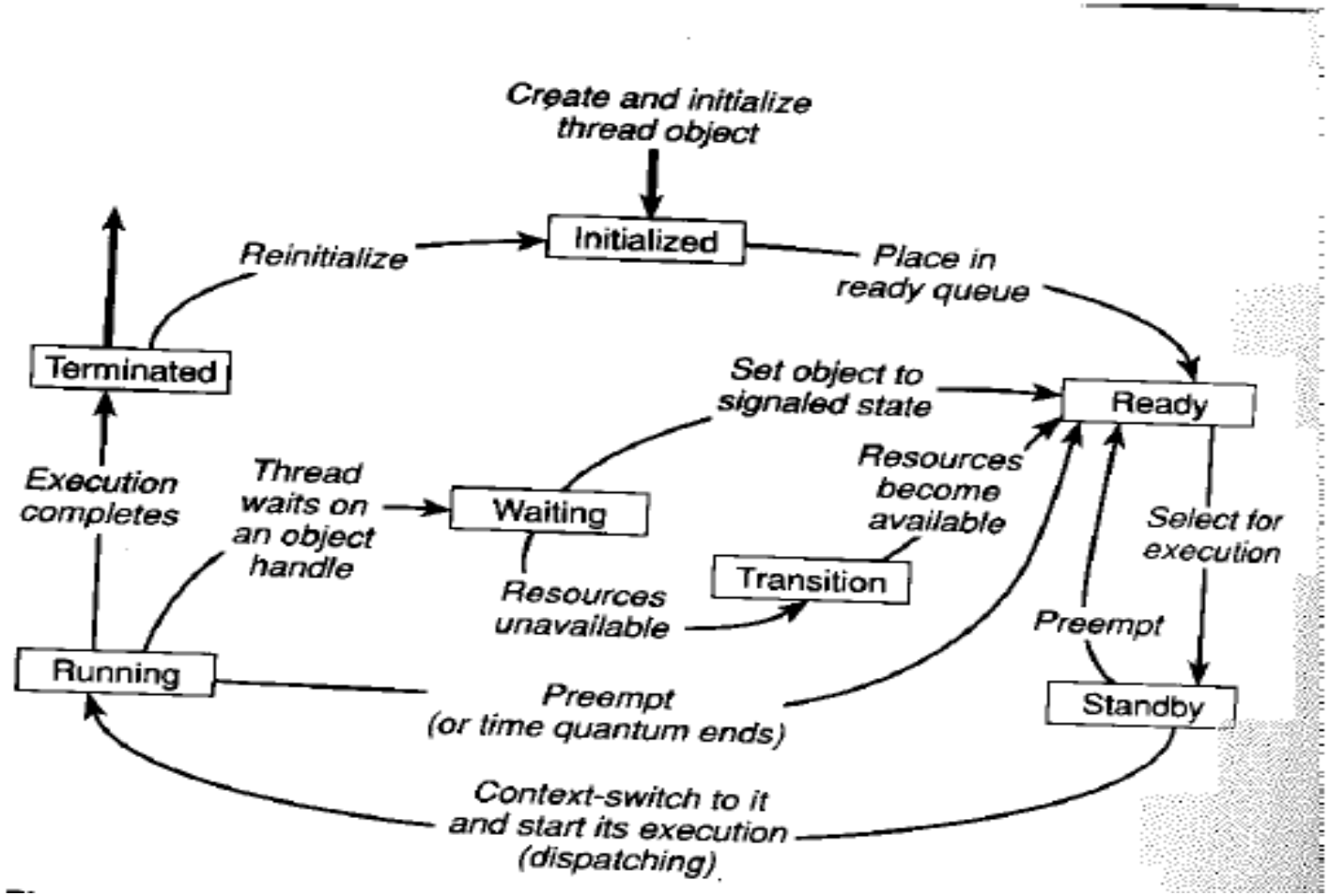
Priority Scheduling

- Assign jobs different priorities
- Higher priority tasks get scheduled earlier
  - Can get faster service
  - Can get larger share of service
- Non-preemptable
  - Run the currently scheduled task to completion
- Preemptable
  - Stop the current process and run the higher priority process
- Tasks at the same priority use round-robin
- Lower priority tasks could starve

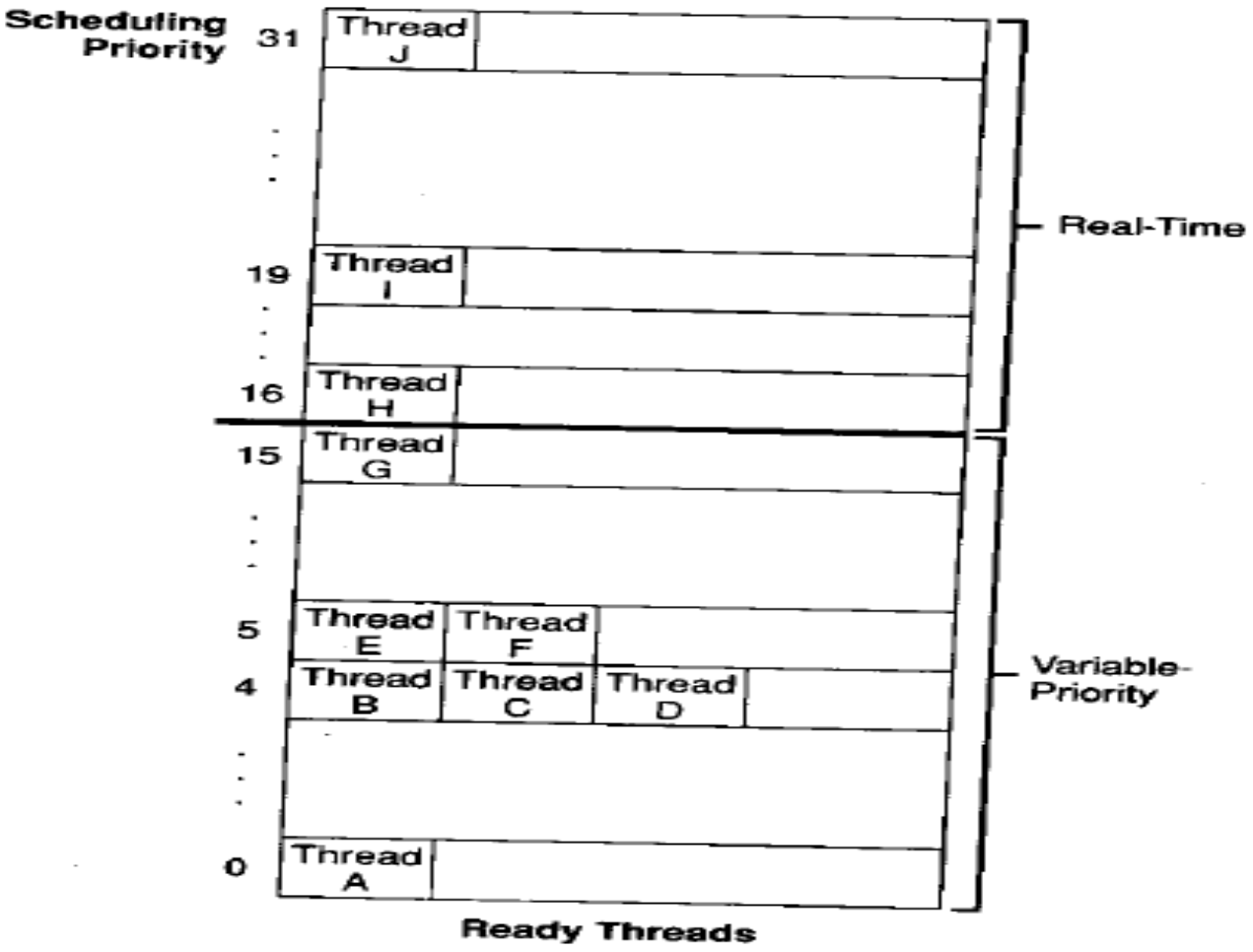
Priority Scheduling

- Priority Inversion can occur
- High priority task waiting on a lower priority task
  - Effectively, lower priority task should have higher priority
- Another medium priority task can run ahead of both of them
- Hard to predict completion schedules of tasks

Priority Scheduling –Windows NT



Priority Scheduling –Windows NT



Meeting Timing requirements

- Can be a challenge for various reasons
- Most O/Ss employ priority-driven scheduling
  - Hard to predict schedule –depends on dynamic nature of tasks
- Depends on timer granularity
  - Scheduler doesn’t run until next clock tick
  - Clock ticks can be as large as 10ms
- Depends on the number of processes
- Depends on interrupts
  - Generally have higher priority and can’t be preempted.
- Real-time O/Ss
  - Designed to reduce task latency

Real-time Scheduling

- When you want things done on time
- Earliest Deadline First Scheduling (EDF):
  - Schedule tasks based on their deadlines
  - Earlier the deadline, higher the priority
- EDF will find a schedule when schedule is feasible
- EDF assigns priorities dynamically
- Rate Monotonic scheduling
  - Useful when tasks arrive at regular rates
  - Assign higher priorities for higher rate tasks