Standby thread state in the Windows NT family*

Julian Kurz
Sven Krohlas - krohlas.de

*NT 4, Windows 2000, XP, Vista. The implementations differ and specific documentation is hardly available.
Thread states overview (simplified)

- **Waiting**
  - Affinity/priority changed
- **Deferred Ready**
  - Preemption
  - Affinity not OK
- **Ready**
  - Preemption
  - Affinity OK
- **Standby**
  - Selected by scheduler
  - Processor Idle/Preemption
- **Running**
  - Dispatching
- **Preemption**
  - No CPU available
Thread states (1)

- Per processor states:
  - Ready
  - Standby
  - Running

- Global states:
  - Deferred Ready
  - Waiting
Thread states (2)

- "Classic States":
  - "Ready": able to run
  - **Running**: current thread running on a processor
  - **Waiting**: blocked, waiting for an event

- **Ready** is split into:
  - **DeferredReady**: queued on any Processor
  - **Standby**: will be imminently start running
  - **Ready**: queued on target processor by priority

see [1]
Scheduling for another CPU

- High priority thread T1 on CPU A exits critical section
- Low priority thread T2 waits for CS
- CPU A can schedule T2 to another CPU B
  - A sets T2 as standby on CPU B
  - A sends interprocessor interrupt to B
  - B dispatches T2
- B does not need to decide again which thread to run next => Scalability
Locking

- local Ready queue will be used mainly, the global Deferred Ready only occasionally
- probability of one processor blocking the others on a SMP system when accessing global data structures is reduced
Literature

  - http://www.i.u-tokyo.ac.jp/edu/training/ss/lecture/new-documents/Lectures/03-ThreadScheduling/