Notes in ECEN 5623

Zachary Vogel

January 21, 2016

Bureaucracy

Dev kit checkout today, start homework, read chapters 1 and 2.

Lecture

CPU Scheduling Overview
3 fundamental resources are processor, i/o, and memory.

Fair vs Unfair scheduling.
need a programmable interval timer interrupt for fair scheduling.
fair scheduling doesn't provide a predictable response.

other option is to prempt and run to completion.

Flynn's Taxonomy

Scheduling Taxonomy tree looks cool.

elements of scheduling class.
scheduling policies: cooperative, preemptive, non-preemptive
sheduling feasibility: can your scheduler work given available resources.
tunability?: if actual differs from expected what can you do?

simple scheduling prediction=pessimistic

WCET = Worst Case Execution Time

Typical RTOS Scheduling Policies:
Single processor, fixed priority Preemptive or dynamic preemptive.
Rate monotonic feasibility test (Least Upper Bound (LUB))
other test: scheduling point algorithm, complpetion test algorithm (over least common multiple (LCM) periods)

task is an implemenation of a service
task control block (TCB)
dispatcher running
dispatch forces a context switch

RTOS local preemptive
Fixed priority RTOS rate monotonic
advantages: deterministic latency, well established, predictable overload failure, can get 100% theoretically,

RM LUB 70%, knowon latency/throughput tuning methods
disadvantages: RM LUB is pessimistic, requires preemption overhead (interrupts, context switch), can be complex for global-multi processor architectures (load balanacing, asymmetric service allocation, message passing overhead)

local preemptive with dynamic priority
advantages: 100 utilization and high throughput
disadvantages: hard, Least latency first(LLF) is easier thand easiest deadline first(EDF), catastrophic failure can occur

compared to linux/unix style RR/time slice
advantages: best effor fair, everbody gets a slice
disadvantages: do we care about fairness? huge overhead on quantum preemption and context switch

Why use an RTOS?
common scheduling framework for:
fixed priority preemptive (Rate monotonic, deadline monotonic), dynamic priority preemptive (EDF, LLF), Non-preemptive cooperative (multi-frequency executive, dataflow)
what else: filesystems, memory, context switching, communication, ectetera

Real time service types:
types: hard real time, soft RT, Best effort, isochronal hard real time (digital feedback control systems), isochronal soft real-time (continuous media, vidio, audio)
Isochronal definition here: the deadline can't be early or late has to be right on time.

real time service types in terms of utility
Utility curve, value/harm of response over time.

Issues beyond policy and feasibility
Latency control: do you allow deadline over-runs, shutdown when deadline is missed?
Dealing with Non-determinism: Do we always have to assume worst case (WCET, Shortest period(worst case release frequency)), when have we tested enough to verify worst case assumptions?
overhead: lump it into worst case? ignore it?