ECEN 5623

RM applications

Application of RM Theory

- Policy to Assign Priorities in RTOS for Service Set
- Feasibility Test (On-Line, Off-Line, By Mode)
  - Off-Line for Fixed Set of Services
    - Transition from Off-Line Boot to On-Line RT Services
    - Service Set Can't Change – No Dynamic Admission
  - On-Line Admission of Services
    - Feasibility Test Must be Run On-Line to Re-Test Feasibility of New Services Before They Go On-Line
    - Feasibility Test Itself is a Service that Must Be Admitted by Off-Line Test
- Most RT Systems Have Static Services Sets
- Or, Modes with Different Service Sets
  - E.g. Space Telescope in Observing Mode, Ready Mode, Safe Mode
- Can Apply RM Feasibility Testing to Each Mode

Application of RM Theory

- Policy to Assign Priorities in RTOS for Service Set
- Feasibility Test (On-Line, Off-Line, By Mode)
  - Off-Line for Fixed Set of Services
    - Transition from Off-Line Boot to On-Line RT Services
    - Service Set Can't Change – No Dynamic Admission
  - On-Line Admission of Services
    - Feasibility Test Must be Run On-Line to Re-Test Feasibility of New Services Before They Go On-Line
    - Feasibility Test Itself is a Service that Must Be Admitted by Off-Line Test
- Most RT Systems Have Static Services Sets
- Or, Modes with Different Service Sets
  - E.g. Space Telescope in Observing Mode, Ready Mode, Safe Mode
- Can Apply RM Feasibility Testing to Each Mode

RM Theory Pessimism

- Pessimistic Assumptions Require Resource Margin for Safety
  - C = WCET
    - Longest Path for Code
    - Path is Executed Least Efficiently
    - WCET Is Worse than 3-Sigma Deviation From Norm (Rarely Happens)
  - T = Worst Case Inter-arrival Rate (Period Transform Required)
  - Importance Encoding Requires Period Transform
  - Critical Instant - Assumes that Service Requests May All Come At Once
  - RM LUB Only Sufficient
    - RM LUB is Based Upon Service Periods That Are Not Necessarily Harmonic, So Partial Interference is Over-Accounted When They Are
    - Feasible Service Sets May Be Rejected, Although Unsafe Will Never Be Admitted
    - Can Use Alternative Scheduling Point or Completion Test RM N&S Feasibility Tests

Pitfalls Applying RM Theory

- Problem: Period is not Constant
  - Service Releases have Period Jitter, are Sporadic, or A-periodic
  - ***Possible Solution***: Period Transform
    - Assume Service Period Based on Worst Case Inter-arrival Rate (Highest Known Frequency of Requests)
    - When Frequency is Lower than Worst Case, Margin Available for Slack Stealers
    - May not Handle Truly A-periodic Services (Worst Case Frequency Unknown)
- Problem: C is not Deterministic
  - Service Execution has Jitter due to Processor Architecture and Data Driven Algorithms
  - ***Possible Solution***: WCET
    - Assume Worst Case Execution Time
      - Longest Path Possible for Algorithms
      - Worst Efficiency Possible Executing that Path (e.g. all cache accesses miss)

Pitfalls Applying RM Theory

- Problem: RM Policy Does Not Encode Importance
  - The Longest Period Service May not Be the Service We Want to Fail in an Overload Situation
  - ***Possible Solution***: Period Transform
    - Assume High Importance Service with Long Period has a Shorter Period than it Actually Does to Artificially Increase Priority
      - Increases System CPU Resource Margin
      - Overload Failures Result in Less Important Services Missing Deadlines
- Problem: Deadline Does Not Equal Period
  - Deadline may be Less than Period when Response I/O Latency is Accounted for (e.g. if Service Response is Transported on a Network)
  - Deadline may be Greater than Period if Service Release Processing is Allowed to Overlap (e.g. 2 or more service requests within T)
  - ***Possible Solution***: Deadline Monotonic Policy and Feasibility Test

Deadline Monotonic Theory

- DM Policy: Higher Priority Given to Services with Shorter Deadlines
- DM Sufficient Feasibility: $$\forall i : 1 \le i \le n : \frac{C_i}{D_i} + \frac{I_i}{D_i} \le 1.0$$

For All Services from 1 to n, if the Deadline Interval is Long Enough To Contain The Service Execution Time Interval Plus All Interfering Execution Time Intervals, Then the Service is Feasible - If all Services are Feasible, Then the System is Feasible.

$$I_i = \sum_{j=1}^{i-1} \left\lceil \frac{D_i}{T_j} \right\rceil C_j$$

$$\left\lceil \frac{D_i}{T_j} \right\rceil$$

Worst Case Number of Releases of S_j Over Deadline Interval for S_i

Interference to Service S_i is due to Preemption By All Higher Priority Services S_1 to S_{i-1} and the total Interference Time is the Number of Releases of S_j Over The Deadline Interval D_i for S_i Multiplied by S_j Execution Time C_j Summed for All S_j which Have Greater Priority Than S_i.

DM Improved Feasibility Test

- DM Sufficient Feasibility Pessimistic
  - Assumes that All Releases of S_j Over D_i Fully Interfere with S_i
  - Last Release of S_j Over D_i May Only Partially Interfere with S_i
- Improved Interference Estimation:

$$I_i = \sum_{j=1}^{i-1} \left[\left[\left\lfloor \frac{D_i - D_j}{T_j} \right\rfloor + 1 \right] C_j + \left[\left\lceil \frac{D_i}{T_j} \right\rceil - \left[\left\lfloor \frac{D_i - D_j}{T_j} \right\rfloor + 1 \right] \right] \times Min\left[C_j, D_i - \left\lfloor \frac{D_i}{T_j} \right\rfloor T_j \right] \right]$$

Full Interference Time

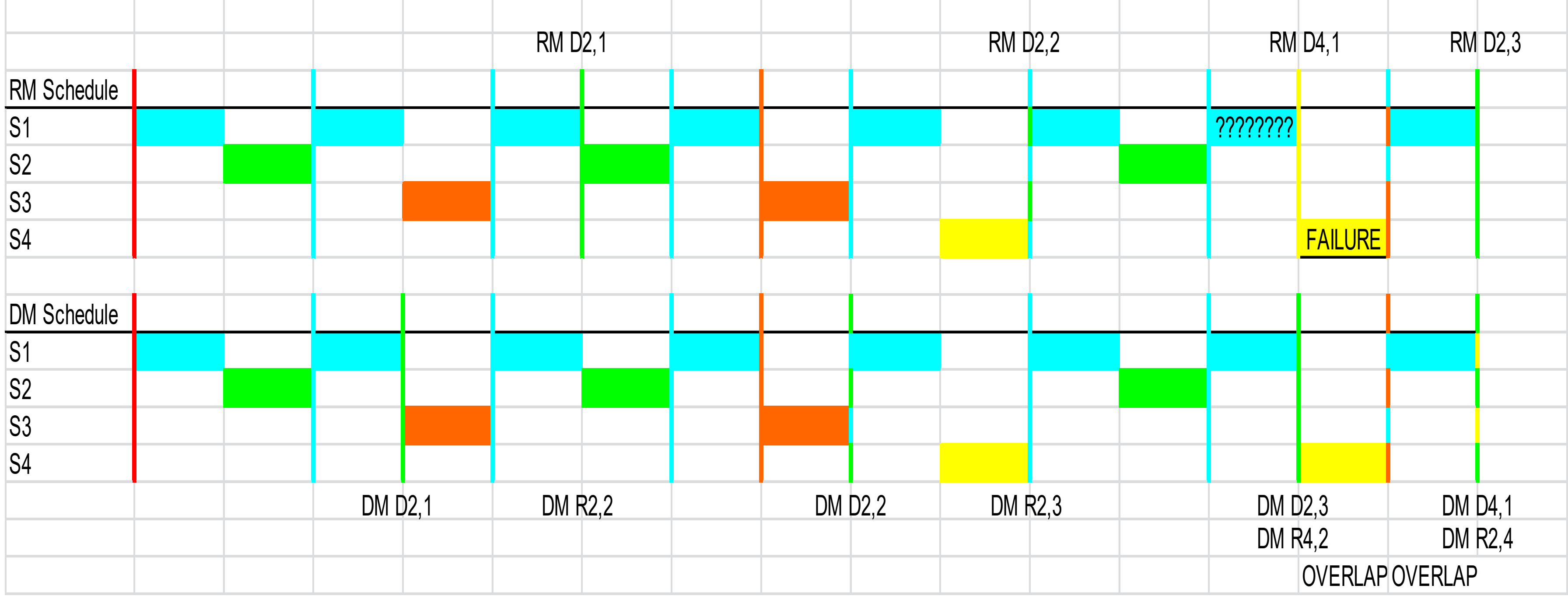
=0 if No Partial Interference
OR =1 if Partial Interference

Minimum of S_j Execution Time
OR S_j Partial Execution Time

- Still Not N&S Feasibility Test
  - Over-Accounts for Partial Interferences (They Don't All Just Sum)
  - Some Partial Interferences Over-lap in Time and Preempt Each Other

Example #6
(DM Ok, RM Fails)

Example 6	T1	2	C1	1	U1	0.5	LCM =	70	For DM	D1	2	
	T2	5	C2	1	U2	0.2				D2	3	EARLIER
	T3	7	C3	1	U3	0.142857				D3	7	
	T4	13	C4	2	U4	0.153846	Utot =	0.996703		D4	15	LATER



RM Necessary and Sufficient Feasibility

$$U = \sum_{i=1}^{m} (Ci/Ti) \leq m(2^{\frac{1}{m}} - 1)$$

- RM LUB is Only Sufficient
  - Order N
  - Sufficient, but Not Necessary and Sufficient
  - A Sufficient Test Will Never Incorrectly Pass an Infeasible Service Set, But Will Fail Some Feasible Ones
  - An N&S Test is Exact – Will Not Pass Infeasible Sets Nor Will it Fail a Feasible Set
- N&S Feasibility Tests (Order N^3) for N Services
  - Based Upon Lehoczky, Shah, Ding Theorem
    - If Services Can Be Scheduled over Longest Period, Then Set is Feasible
    - Best To Look over LCM (Least Common Multiple) of All Periods
  - N&S Test Concept: Iterate Over Subsets of Services for Each Longest Period for All Services
  - Encodes Process of Hand Drawing a Service Timing Diagram With Preemption According to RM Policy
    - Scheduling Point (Lehoczky, Shah, and Ding)
    - Completion Test
  - See RTECS Text for Details on Algorithms

Rate Monotonic or EDF?

- Liu & Layland Rate Monotonic - Fixed Priority Policy
  - Feasibility Test - Sufficient RM LUB
  - Most Hard RT Systems Have Static Services Sets

$$U = \sum_{i=1}^{m} (Ci / Ti) \leq m(2^{\frac{1}{m}} - 1), U \lim m \to \infty = \ln(2) \approx 0.69$$

- EDF (Deadline Driven) – Dynamic Priority Policy
  - Sufficient Feasibility Test Based on Hyper-Period (Product of All Periods)

$$\forall tasks \in 1 \ldots m, T_{hyperperiod} = 0 \ldots (T_1 T_2 \ldots T_{m-1} T_m)$$

$$(T_1 T_2 \ldots T_{m-1} T_m) = \frac{(T_1 T_2 \ldots T_{m-1} T_m)}{T_1} C_1 + \ldots + \frac{(T_1 T_2 \ldots T_{m-1} T_m)}{T_m} C_m$$

$$\therefore \sum_{i=1}^{m} (Ci / Ti) \leq 1$$

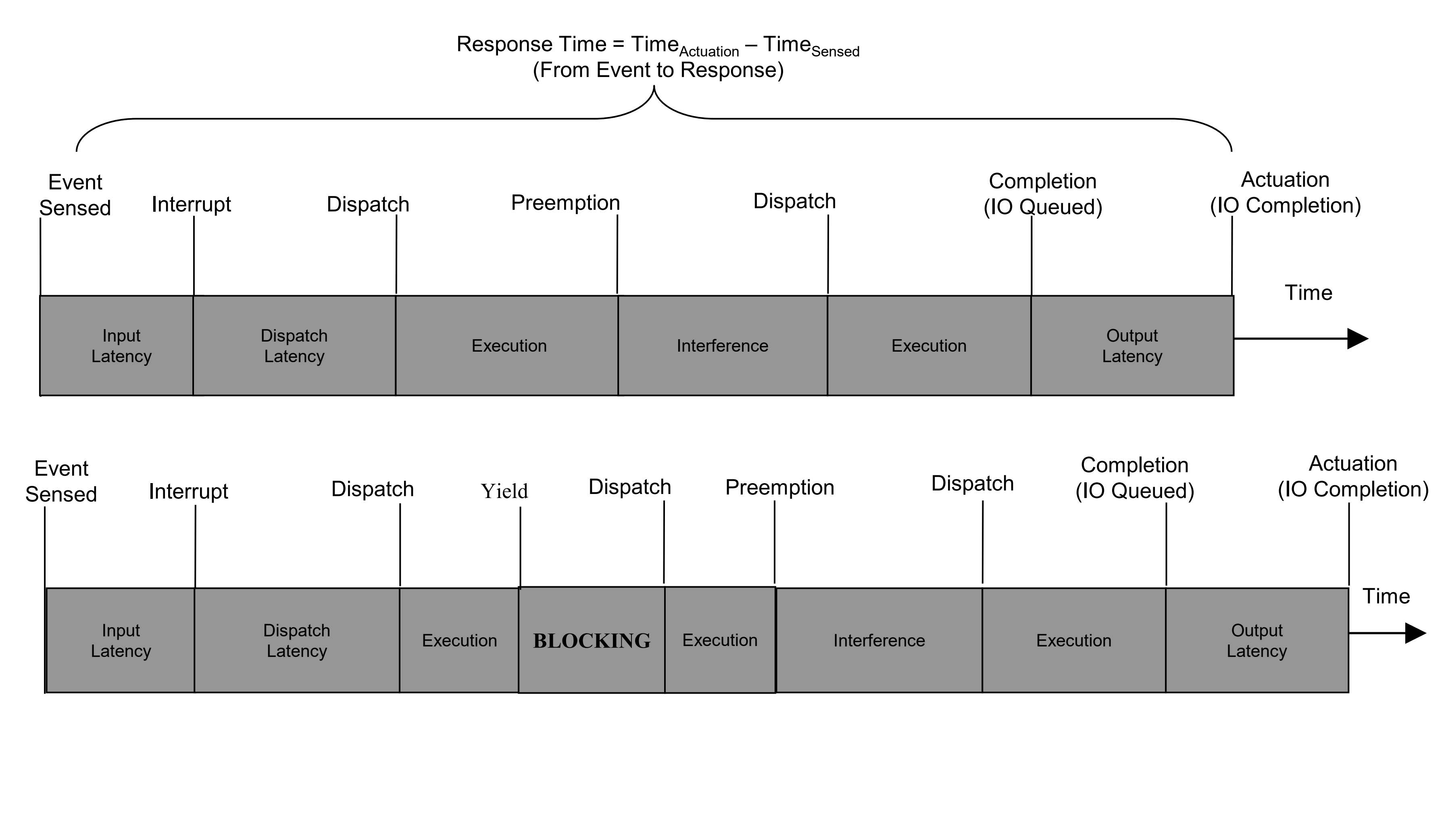
Notes on Dynamic Admission

- Off-Line for Fixed Set of Services
  - Transition from Off-Line to On-Line RT Services
  - Service Set Can't Change – No Dynamic Admission
- On-Line Admission of Services
  - Feasibility Test Must be Run On-Line to Re-Test Feasibility of New Services Before They Go On-Line
  - Feasibility Test Itself is a Service that Must Be Admitted by Off-Line Test or Run on Dedicated Hardware

Pitfalls Applying RM Theory

- Problem: Service Requires Resources In Addition to CPU
  - A Service may Block
  - Blocking on a Resource When by RM Policy the Service Should be Executing Causes Priority Inversion - I.e. a Service With Lower Priority Runs in Place of the Service Which is Expected to Run
  - ***Possible Solution***: Eliminate or Model Worst Case Blocking Time in Response Time (I.e. Inversion Duration is Bounded)
    - CPU is Yielded While Blocking, so Blocking Time Creates Margin, but Must Be Accounted for in Response Time
    - Blocking Could Be Eliminated by Polling in Some Circumstances if Additional Resource is Optional for Progress

Service Response Timeline
(With Intermediate Blocking)



What is I/O Latency Hiding?

A. Creating a separate thread for IO
B. Scheduling IO so that it overlaps
C. Dedicating a separate processor for IO
D. Allowing IO to access only one dedicated memory area

When poll is active, respond at **pollev.com/timscherr391**

Text **TIMSCHERR391** to **37607** once to join

What is I/O Latency Hiding?

A Creating a separate thread for IO

B Scheduling IO so that it overlaps

C Dedicating a separate processor for IO

D Allowing IO to access only one dedicated memory area