ECEN 5623

RT Resources

Be Boulder.

University of Colorado Boulder

Outline

- Real-Time Services
  - The RT Embedded System Resource Space
    - CPU
    - IO
    - Memory
  - Service Release Timeline
  - Intro to Timing diagrams
  - Intro to Traces
  - Safe Real-Time Resource Utilization Bounds
- High-Level Design
  - Input Interfaces
  - Output Interfaces
  - Services
  - Real-Time Requirements (C_i, T_i, D_i for S_i)
  - Decomposition from Services to Functions and Vice Versa
  - Methodologies
    - Structured Analysis/Design
    - UML
    - SDL

View of RT CPU Resources

- CPU
  - ***How Fast?*** - Clock Rate
  - ***How Efficient***? - CPI = Clocks Per Instruction or IPC = Instructions Per Clock
    - Pipeline Stalls Due to Hazards – e.g. Read Data Dependency
    - Cache Misses, Write Buffer Stalls, Branch Mispredicts
  - ***How Complex?*** – C_i = Instruction Count on Service Longest Path
    - Ideally C_i Deterministic
    - WCET – Worst-Case Execution Time
      - Longest, Most Inefficiently Executed Path for Service
      - Not Directly Related to Response Time
        - IO Latency
        - Interference by Higher Priority Services and Interrupts
  - ***How Often?*** – T_i = Service Release Period

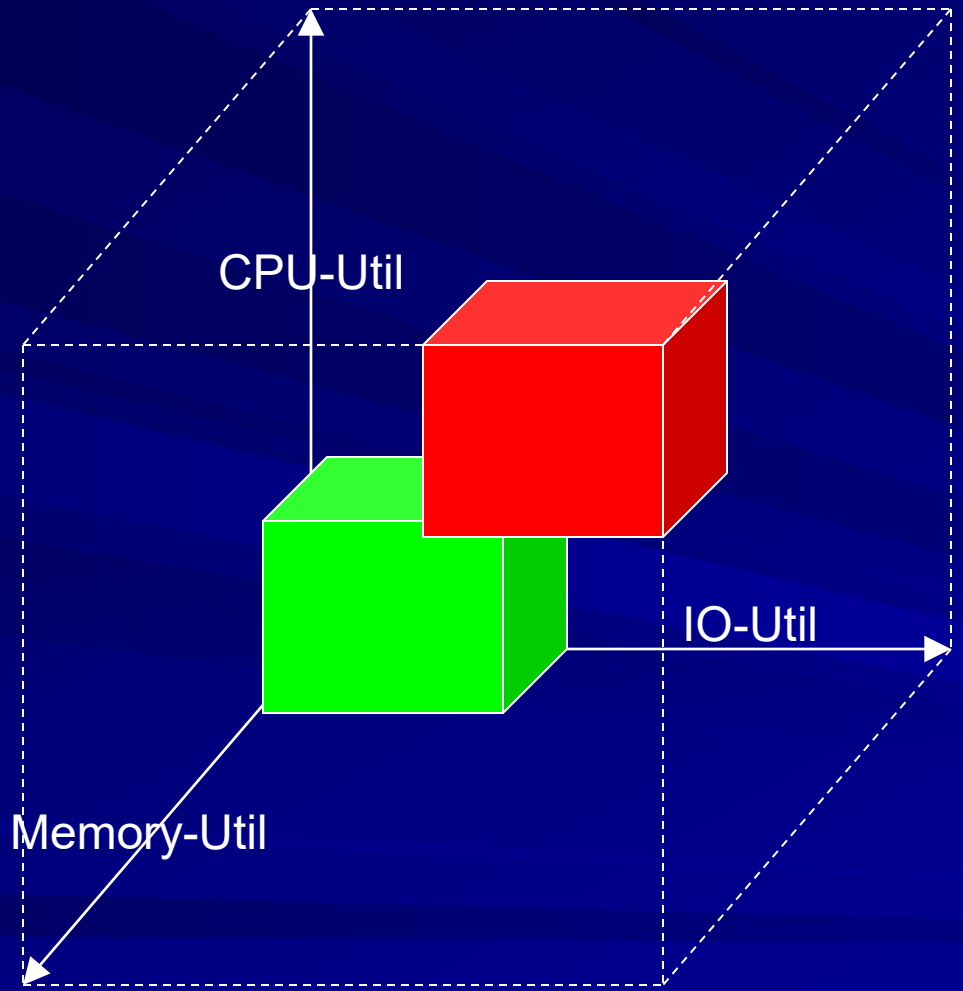
View of RT IO Resources

- IO
  - *Latency?*
    - *Arbitration Latency for Shared IO Interfaces – e.g. Bus*
    - *Read Latency*
      - *Time For Data Transit From Device to CPU Core Register, TCM or L1 Cache*
      - *Register, Tightly-Coupled-Memory, and L1 Cache Considered Zero Wait-State Single Cycle Access*
      - *Bus Interface Read Requests and Completions*
        - *Split Transaction*
        - *Delay*
    - *Write Latency*
      - *Time for Data Transit From CPU Core to Device*
      - *Posted Writes Prevent CPU Stalls*
      - *Posted Writes Require Bus Interface Queue*
  - *Bandwidth?*
    - *Average Bytes or Words Per Unit Time*
    - *Says Nothing About Latency*
  - *Queue Depth?*
    - *Write Buffer Stalls*
    - *Read Buffer – Most Often Stalled by Need for Data to Process*
  - *CPU Coupling?*
    - *DMA*
    - *Programmed IO*
    - *Cycle Stealing*

View of RT Memory Resources

- Memory Hierarchy From Least to Most Latency
  - Level-1 Cache
    - Single Cycle Access
    - Typically Harvard Architecture – Separate Data and Instruction Cache
    - Locked for Use as TCM, Unlocked for Set-Associative or Direct Mapped Cache
  - Level-2 Cache or TCM
    - Few or No Wait-States (e.g. 2 Cycle Access)
    - Typically Unified
    - Locked for Use as TCM, Unlocked to Back L1
  - MMR – Memory Mapped Registers
  - Main Memory – SRAM, SDRAM, DDR
    - Processor Bus Interface and Controller
    - Multi-Cycle Access Latency On-Chip
    - Many-Cycle Access Latency Off-Chip
  - MMIO – Memory Mapped IO Devices
  - Non-Volatile Memory
    - Processor Bus Interface and Controller
    - Slowest Read/Write Access, Most Often Off-Chip
    - Requires Algorithm for Block Erase
      - Interrupt Upon Completion
      - Poll for Completion
- Total Capacity for Code, Data, Stack, Heap
- Allocation of Data, Code, Stack, Heap to Physical Hierarchy
  - Most Frequently Accessed In Lowest Latency Segment
  - Moving Blocks or Cache Lines/Sets
  - Prefetches

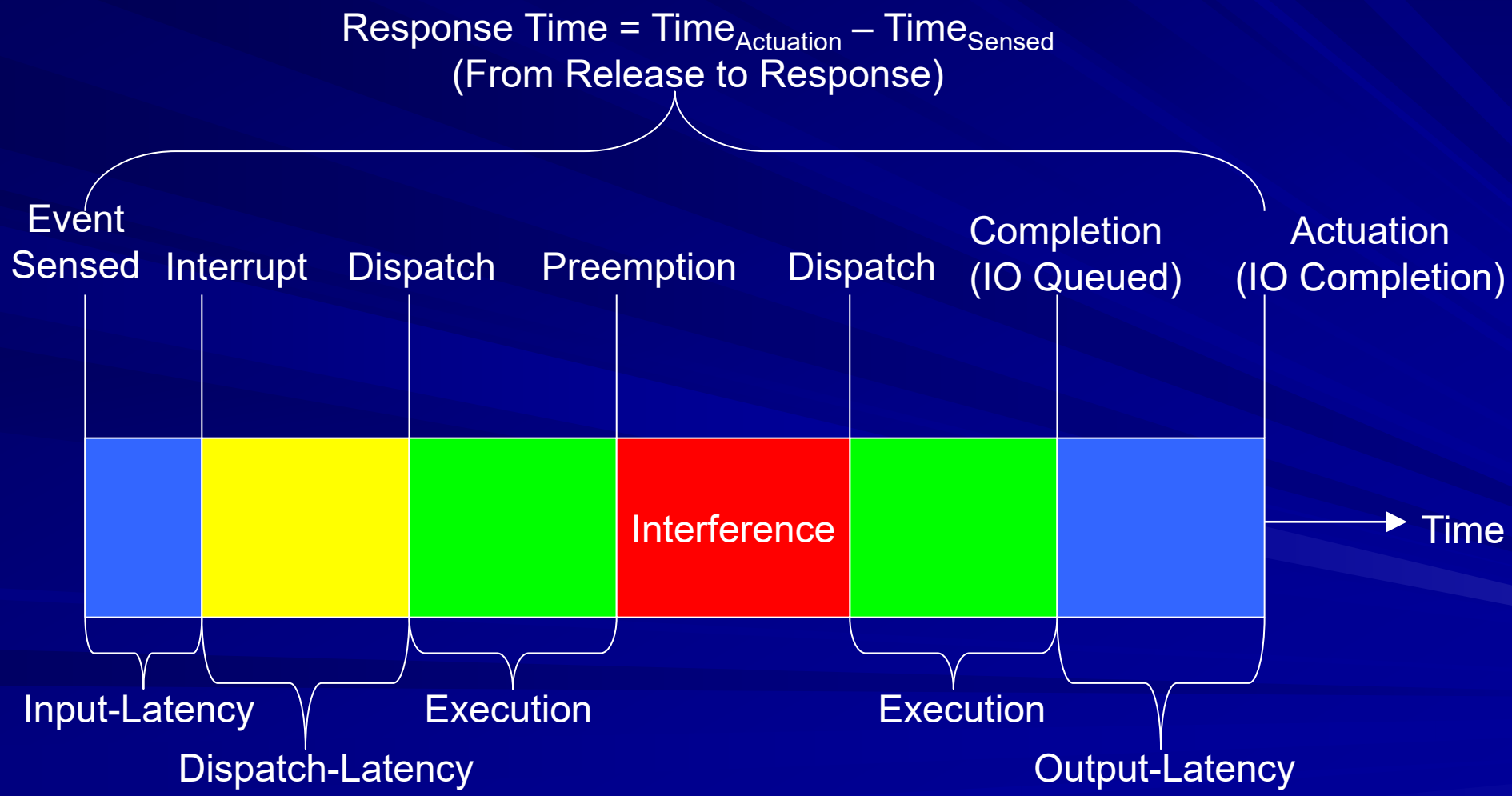
Conceptual View of RT Resources



- Three-Space View of Utilization Requirements
  - CPU Margin?
  - IO Latency (and Bandwidth) Margin?
  - Memory Capacity (and Latency) Margin?
- Upper Right Front Corner – Low-Margin
- Origin – High-Margin
- Most RT Systems will be CPU, IO, or Memory Bound

A Service Release and Response

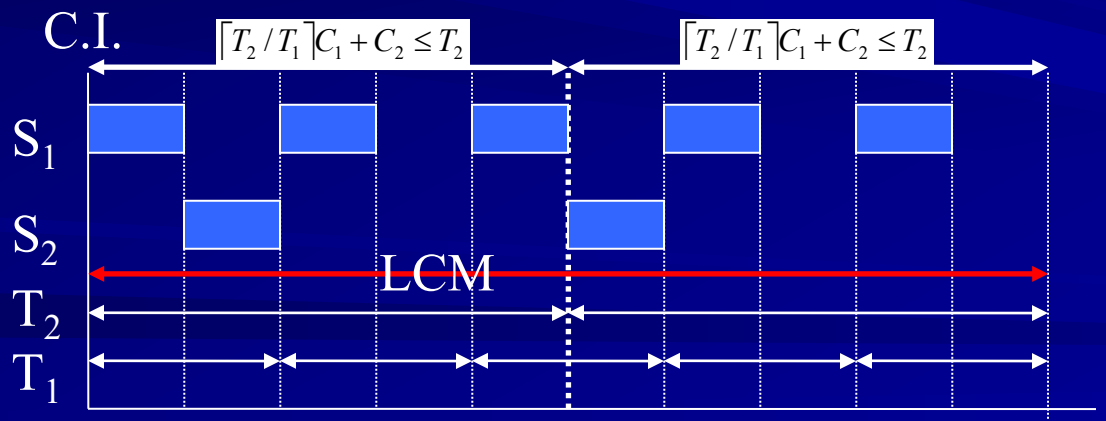
- C_i WCET
- Input/Output Latency
- Interference Time



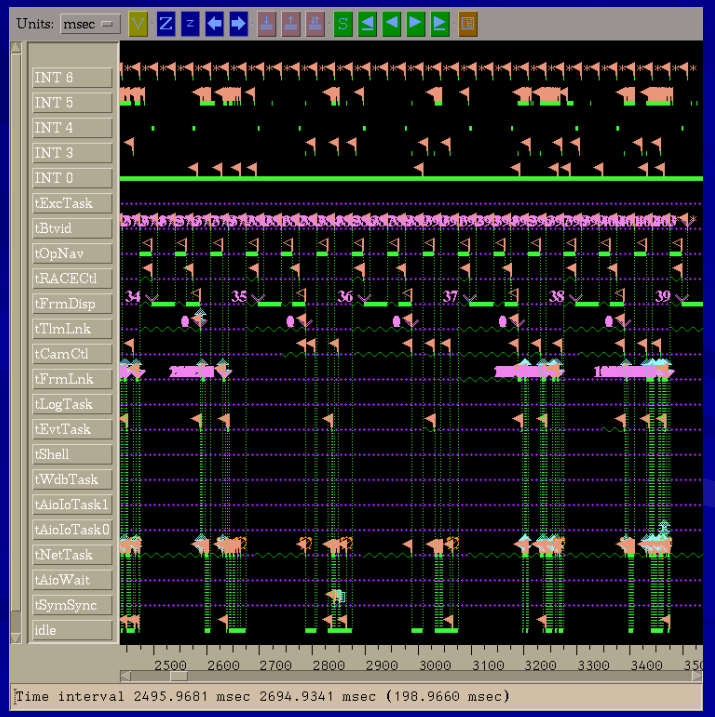
Real-Time Services

- Theoretical Timing Diagrams
  - Lehoczky, Shah, and Ding Theorem
  - Look at Timing Over LCM of All Periods
  - Simple for Small Number of Services
- Necessary and Sufficient Proof of Feasibility

Given: Services S1, S2 with periods T1 and T2 and C1 and C2, Assume T2 > T1
E.g. T1=2, T2=5, C1=1, C2= 1, then if prio(S1) > prio(S2), we can see that the service set is feasible by diagramming.



- Trace Tools for Real Timing
  - Software Trace Tools
    - WindView, Systemviewer(below)
    - Linux Trace Toolkit
    - Ftrace, Kernelshark, LTTng
    - Syslog with timestamps



Safe Resource Utilization Bounds

- CPU
  - How Much Margin to Guarantee Deadlines Will Be Met for a Set of Services?
    - Accounting for Interference
    - How Well Can CPU Be Scheduled?
  - Sufficient – RM LUB
  - Necessary and Sufficient – Lehoczky, Shah, Ding Theorem
  - If CPU Was the Only Resource Needed …
    - Priority Inversion
    - Efficiency
- IO
  - Is Bandwidth a Guarantee?
  - What About Latency?
- Memory
  - Allocation in Hierarchy By Latency Requirements
  - Total Capacity
  - Dynamic Memory Allocation?
  - Queue Depths
  - Producer/Consumer Rates – Matched?

High-Level Design

- Input Interfaces – HW Design and Characterization
- Output Interfaces – HW Design and Characterization
- Services – What Are They?
- Real-Time Requirements (C_i, T_i, D_i for S_i)
  - Identification of Services
  - Frequency of Service Requests
  - Execution/Processing Complexity
  - Deadline for Response
- Decomposition from Services to Functions and Vice Versa
- System-Level Methodologies
  - Structured Analysis/Design
  - UML
  - SDL

Rate Monotonic Understanding

- Real-Time Correctness requires [choose best]:
  A. Servicing requests for processing as fast as possible
  B. Servicing requests as fairly as possible
  C. Functionally correct output produced prior to a deadline
  D. Resource margins greater than 30%
  E. Periodic processing with minimal jitter

- Task/thread interference is [choose best]:
  A. When a thread/task is ready but lacks shared memory
  B. Stalls in the microprocessor pipeline during execution
  C. Cache misses that reduce execution efficiency
  D. Time during which the thread/task is preempted by another with higher priority
  E. None of the above

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

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

Real-Time Correctness requires [choose best]:

A Servicing requests for processing as fast as possible

B Servicing requests as fairly as possible

C Functionally correct output produced prior to a deadline

D Resource margins greater than 30%

E Periodic processing with minimal jitter

Powered by Poll Everywhere

Start the presentation to see live content. For screen share software, share the entire screen. Get help at **pollev.com/app**

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

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

Task/thread interference is [choose best]:

A When a thread/task is ready but lacks shared memory

B Stalls in the microprocessor pipeline during execution

C Cache misses that reduce execution efficiency

D Time during which the thread/task is preempted by another with higher priority

E None of the above

Powered by Poll Everywhere

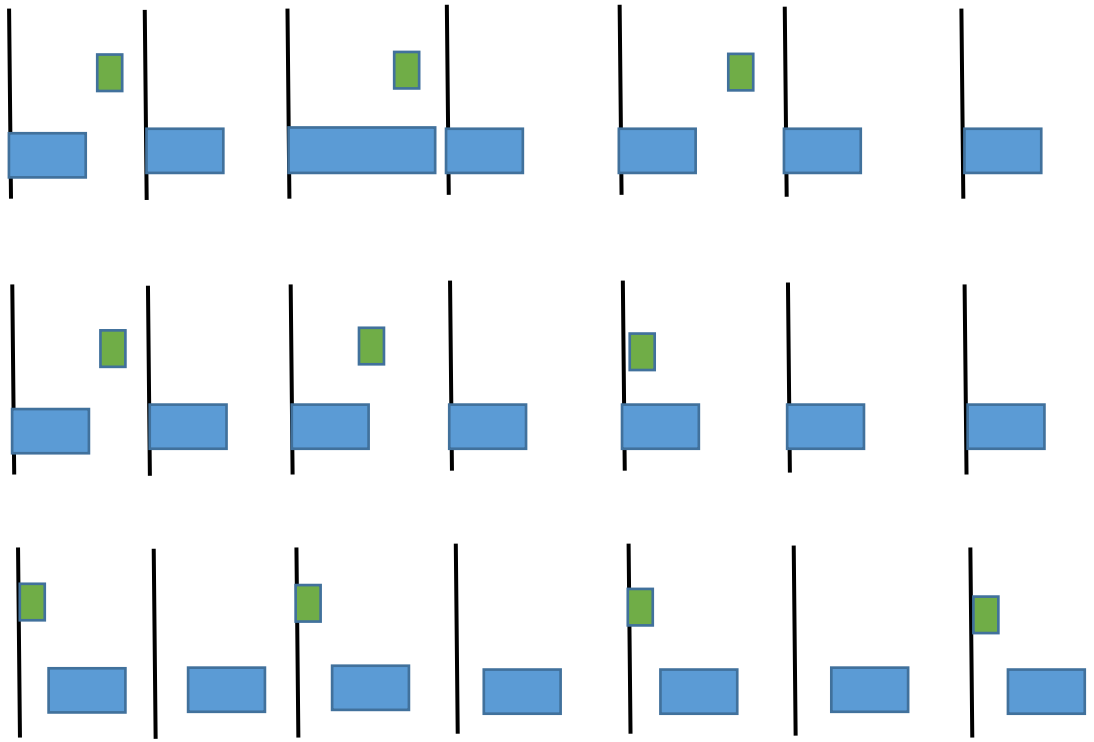
Start the presentation to see live content. For screen share software, share the entire screen. Get help at **pollev.com/app**

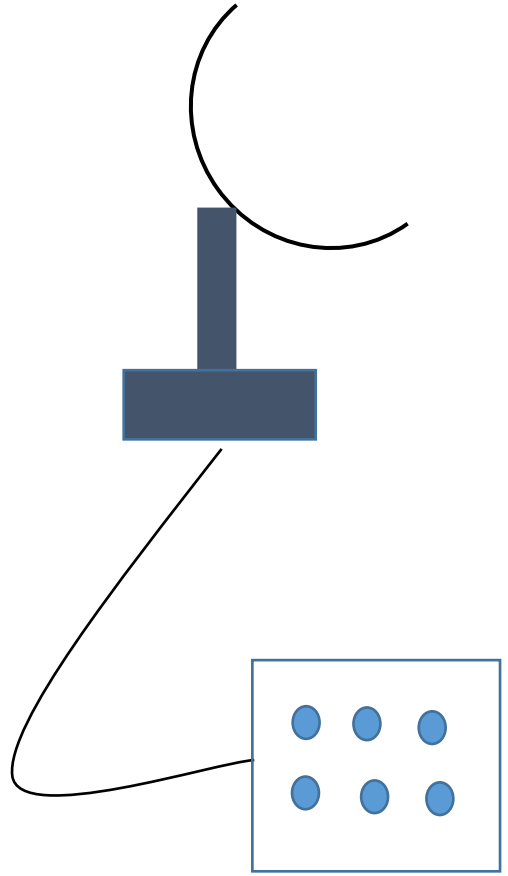
An Early RT Challenge

Antenna control system with position updates for inertial navigator

Navigation data every 20 ms

Antenna resolver sensors sampled every 10 ms





Space Transportation System – Shuttle Example (1989-1992)

- Ascent and Entry Guidance - PASS
- Architecture of the Space Shuttle Primary Avionics Software System – Gene D. Carlow [Canvas]
- Phases of Flight Divided into Major Modes [E.g. 601]
- Each Mode Has Real-Time Scheduling
  - Cyclic Executive with High, Medium and Low Frequency
  - Each Executive is a “Loop” with Interrupts to Preempt it
  - Services are Implemented on an Executive
    - E.g. Control and Sensors in HFE (Stability, Monitoring, Health & Safety)
    - E.g. Navigation in MFE (State Estimation from Multiple Sensors over Time)
    - E.g. Guidance in LFE (Long Term Targets)

DATA CHASER Shuttle Payload Example (1994-97)

- Rapid Development
  - 1 Year from Concept to Launch
  - Demonstration of AI Real-Time Automation for Payloads
  - Anytime Algorithms Used for On-Line Planning/Re-planning Services
- Motorola 68EC030
- Custom FPGA IO Boards for 3 Instruments
- Fairly Low Resource Utilization (Low Frame Rates)
  - RS232 9600 baud Command/Telemetry
  - RS422 1 Mbit / sec Downlink for Science Data
- Flown on STS-85
- Automation Demonstration Successful
- Science Return Minimal
- Developed by CU for JPL

SIRTF High Level Design Example (1997-2000)

- CPU Bound – 95%+ Loading for Sky-Scan Mosaic Images
  - Data Compression
  - Synchronization Between Space Telescope Slew and Imaging
- IO Bound – 95%+ VME Bus Loading for Sky-Scan
  - 3 Synchronized Detectors Operating Concurrently
  - FIFO interface Block Transfer
    - Write-through DMA prevented use of DMA
    - Read/Write Multiple Instruction on PowerPC (Programmed IO)
  - Data Compression and Grouping for Downlink
- Allocation of Functions/Service to:
  - 2 Hardware State Machines (Actel FPGA)
  - 23 VxWorks Tasks
- In Operation Now … (www.spitzer.caltech.edu)

Next Time …

- Cheddar
- Complete RM LUB Derivation
- Discuss Pitfalls
- Introduce Extensions to overcome Pitfalls