IMA: The Good, The Bad, and The Ugly

Peter H Feiler

phf@sei.cmu.edu

November 8, 2007

Outline

The Good

The Bad

The Ugly

Conclusion

Integrated Modular Avionics

Partitioned system architecture (ARINC 653)

- Componentized system
- Migration to common compute platform
- Integration of embedded software systems

Network architecture (ARINC 429, 629)

- Globally synchronous
- Globally asynchronous Locally Synchronous (GALS)

The Good

Partitioned system architecture

- Flexibility through configurability of componentized system & migration of legacy components
- Reduced cost through shared compute platform & increased utilization

Space & time partitioning

- Impact of run-away threads contained to single partition
- Partition-specific scheduling policies facilitate integration of subsystems
- Protected address spaces provide fault isolation barrier for safety-critical subsystems

Inter-partition communication semantics

- Directional port communication facilitates partition distribution
- Phase-delay semantics maintain determinism despite concurrency and partition reordering

Outline

The Good

The Bad

The Ugly

Conclusion

Late Discovery of System Problems

System integration problems

- System instability and failures
- Implicit and mismatched assumptions
- Shared computing resources
- Complexity of component interaction
  - Functional
  - Extra-functional

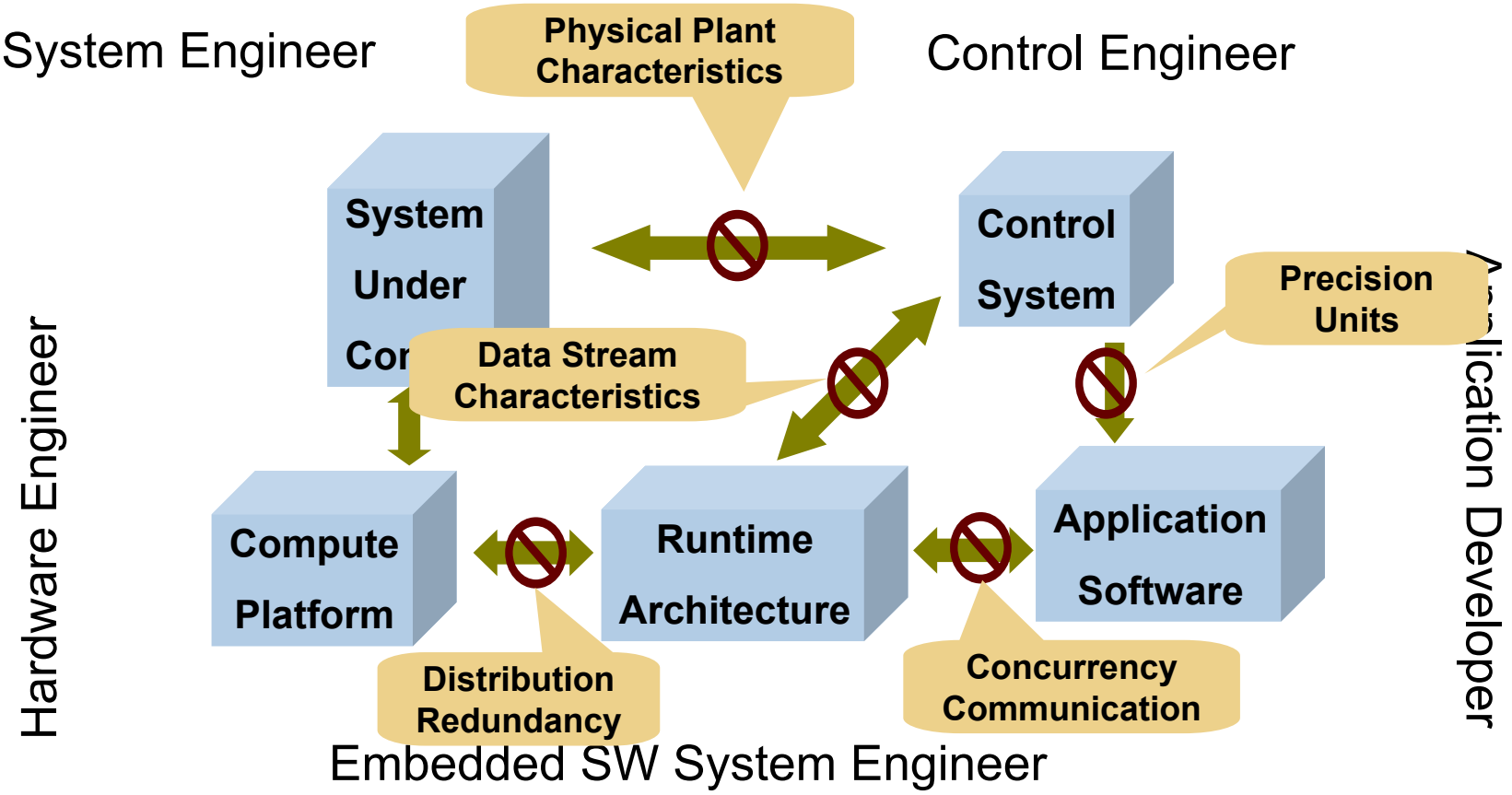
Current practice

- Build components first
- Then integrate and test

Way forward

- Analyze system models early and often
- Evolve components and integrated system

Mismatched Assumptions



Partition Assumptions

Partitions cannot affect other partitions in terms of resource use

- Unmanaged resource sharing across partitions
  - Partitions on different processors utilize shared hardware
- Unmanaged partition initiated tasks
  - DMA transfer continues on partition switch
  - Same memory accessed by DMA & instruction fetch

Partition cannot affect OS services

- Unmanaged DMS transfer may slow cache swap during partition switch

Partition Assumptions

Scheduling analysis is partition insensitive

- Task set on processor of prorated speed
- Pre-period deadline may not be met due to late window slot allocation

Fault tolerance through redundancy

- Partition virtualizes processors
- Partition binding must be considered

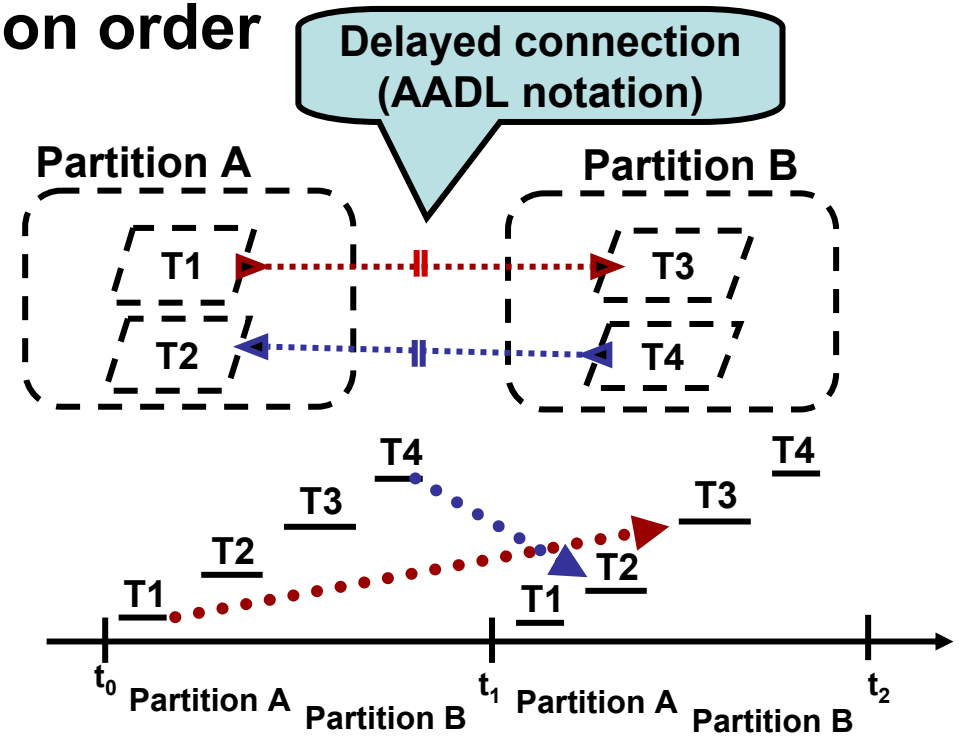
Inter-partition communication is always phase delayed

- Communication timing is sensitive to application level send/receive
- Application level legacy communication may impose additional delay

Partition Order & Timing Semantics

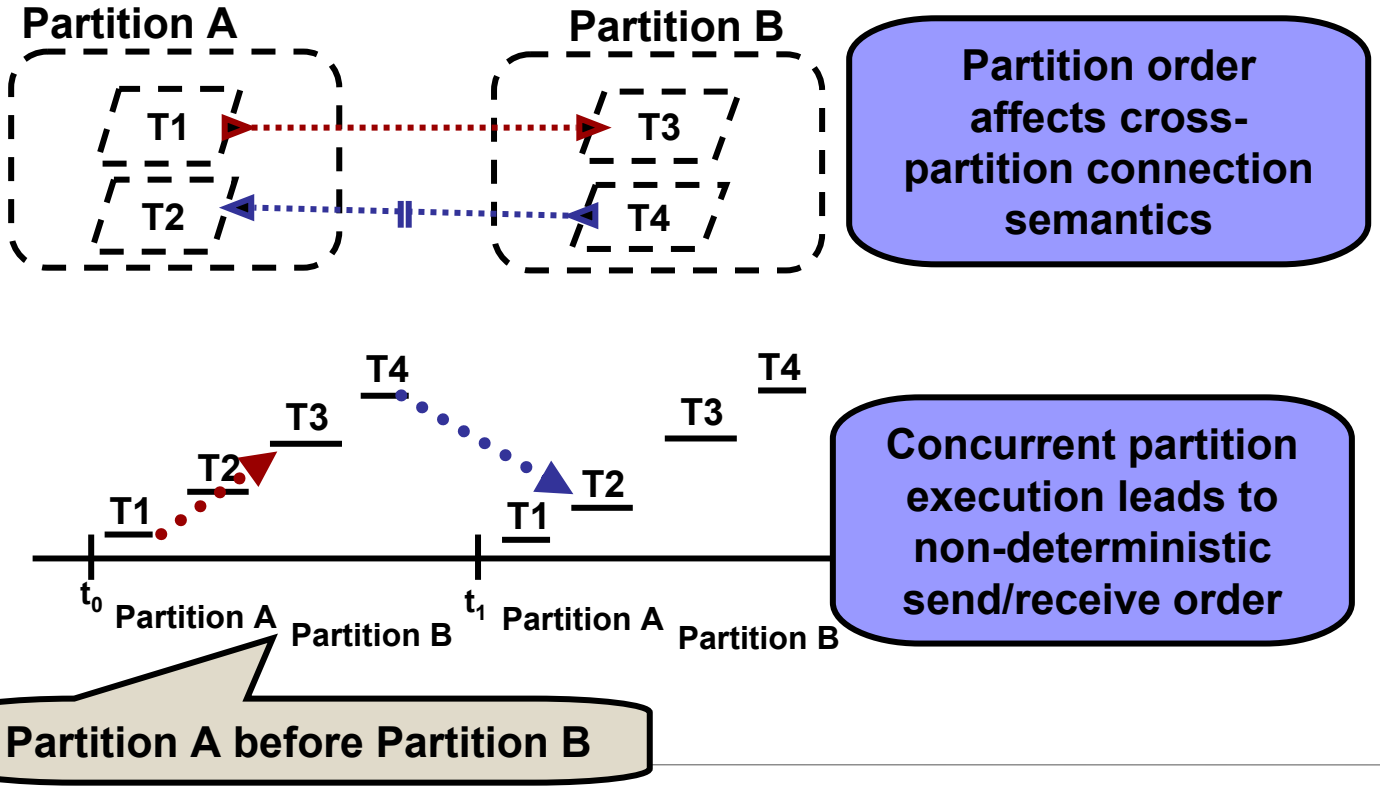
ARINC 653: enforced frame-delayed partition communication

Timing semantics are insensitive to partition order

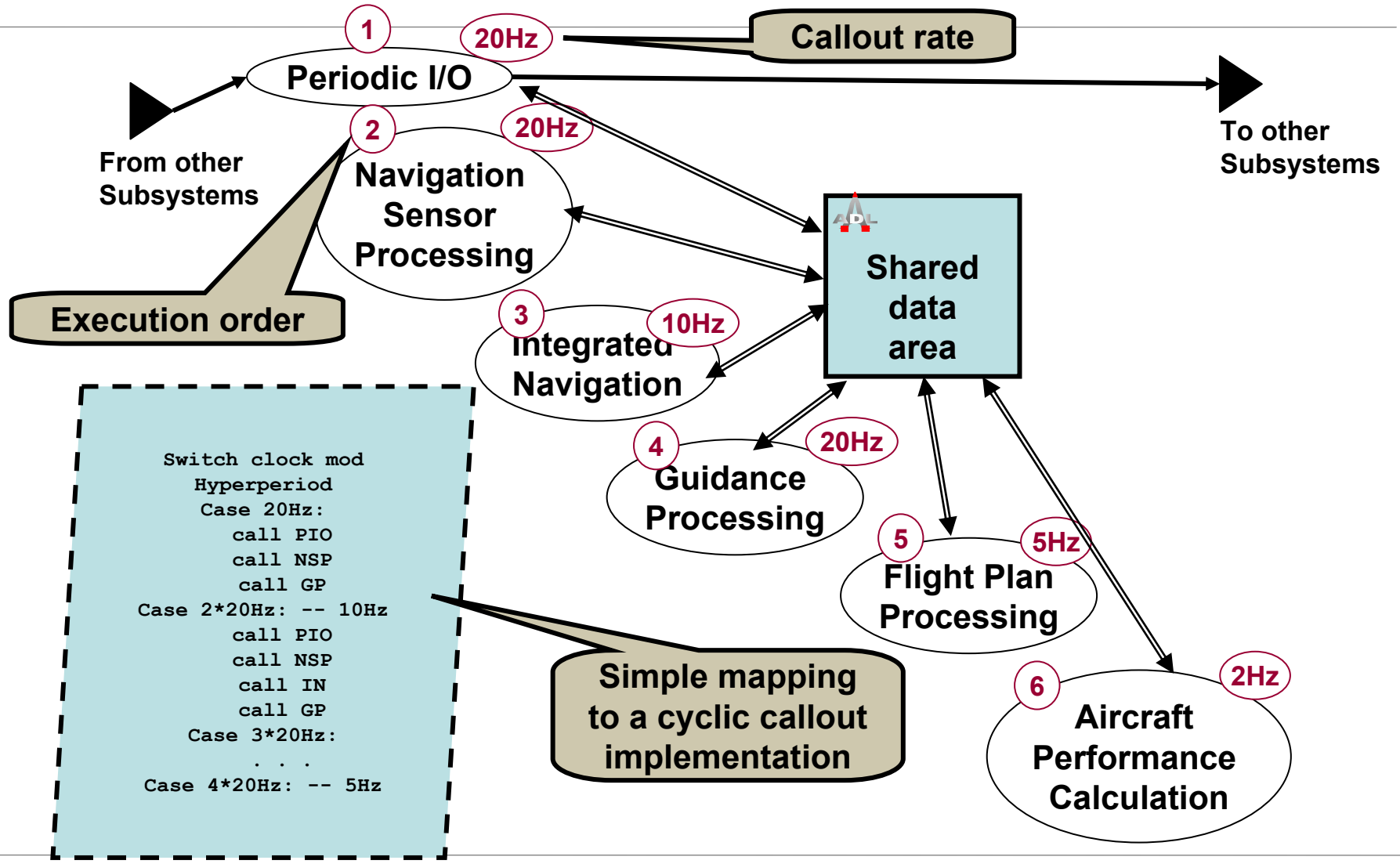


Application Level Send/Receive

- Message-based communication
- Transmission initiated by application send
- Sensitive to partition order & concurrency

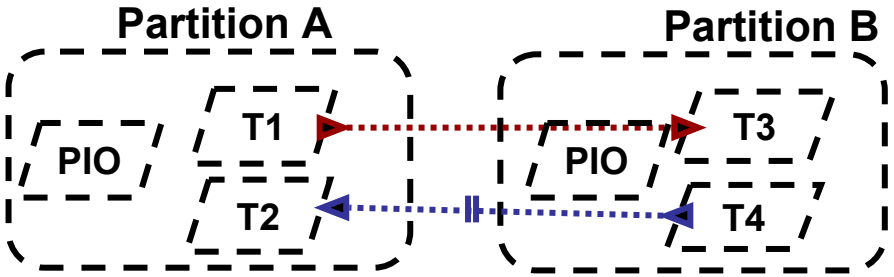


Legacy Phase-Delayed I/O

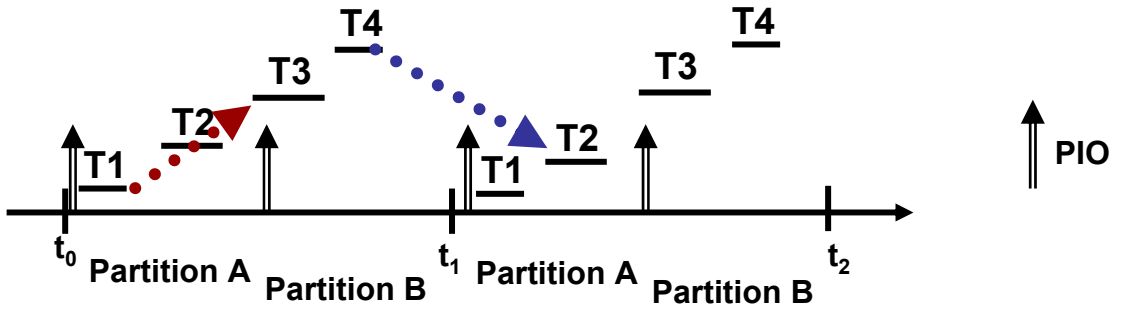


Partition Level Send/Receive

- Message-based communication
- Transmission handled by PIO
- Sensitive to partition order & concurrency



Partition order affects Periodic IO



Outline

The Good

The Bad

The Ugly

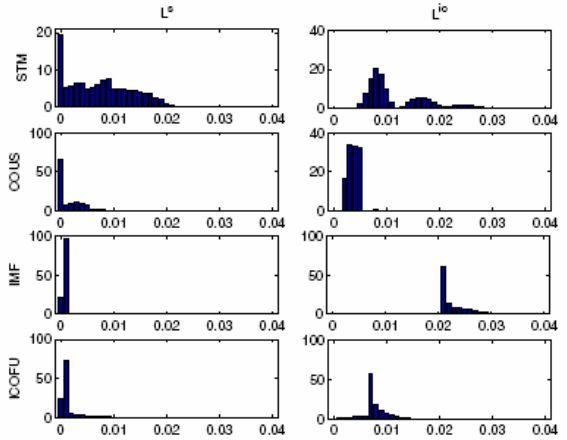
Conclusion

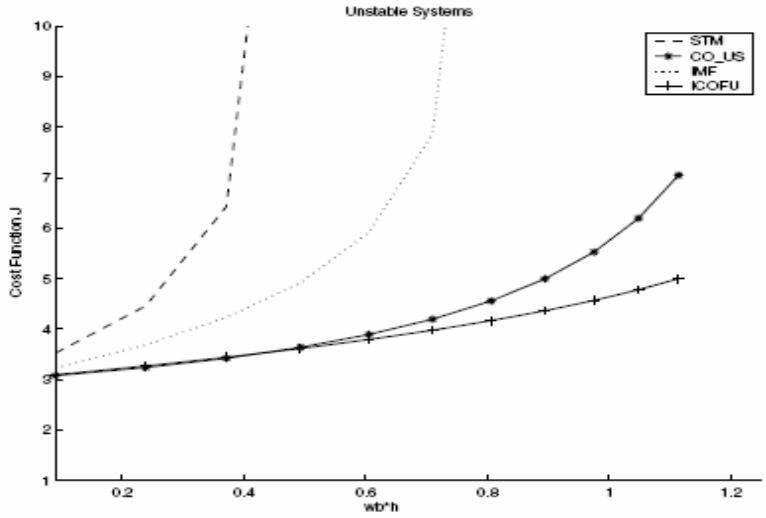
Impact of Sampling Latency Jitter

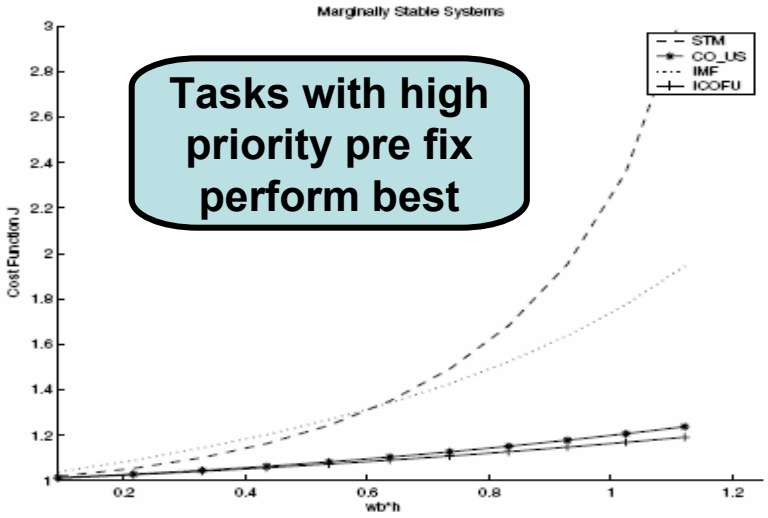
Impact of Scheduler Choice on Controller Stability

- A. Cervin, Lund U., CCACSD 2006

Root cause: sampling jitter due execution time jitter and non-deterministic communication







Tasks with high priority pre fix perform best

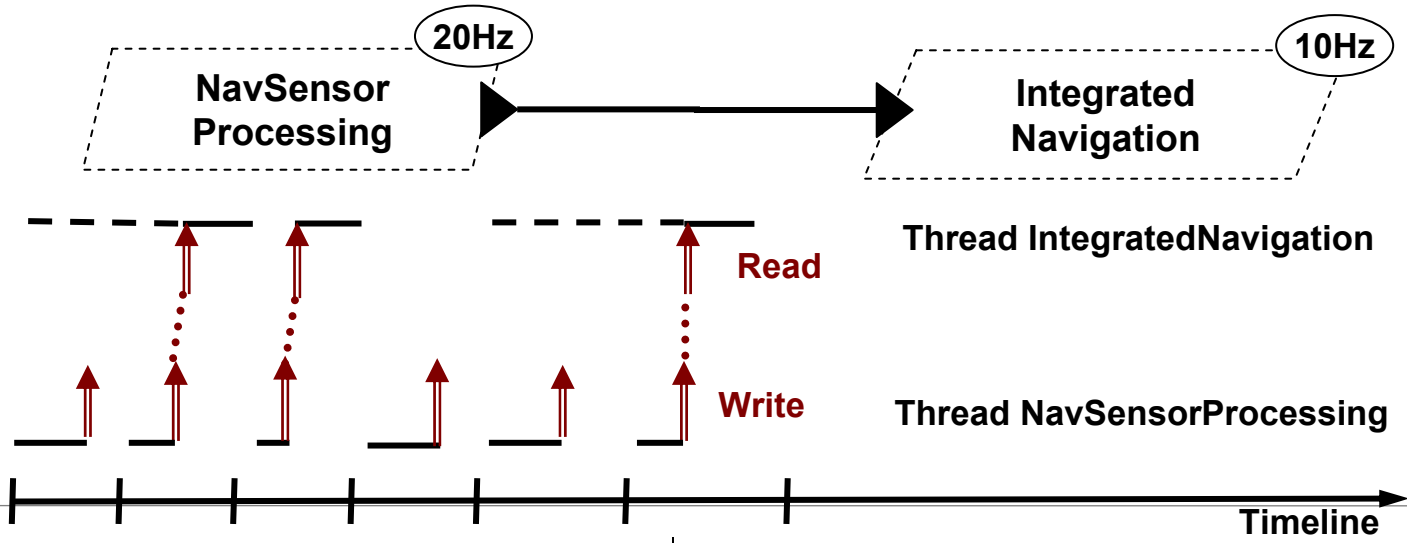
Frame-Level Latency Jitter

Variation in actual write & read time due to preemption or concurrency

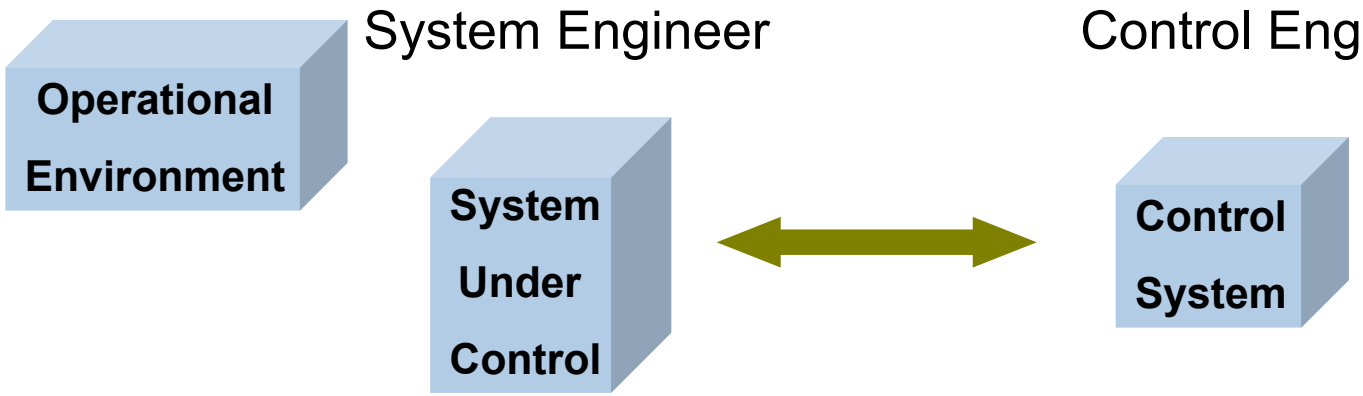
- Operation performed by application code
- Preemption of application threads

Example: Downsampling

- Desired sampling pattern 2X: n, n+2, n+4 (2,2,2,…)
- Worst-case sampling pattern: n, n+1, n+4 (1,3,…)



Latency Contributors



- Processing latency
- Sampling latency
- Physical signal latency
- Age vs. latency

Software-Based Latency Variation & Jitter Contributors

Preemptive thread scheduling & legacy shared variables

Concurrency due to multiple & multi-core processors

Resource contention

Protocol specific communication delay

Globally asynchronous systems

Rate group optimization within partition

Migration of partitions

Application redundancy & partition binding

Preemptive scheduling of partitions

Data-driven processing & cross-partition communication

Outline

The Good

The Bad

The Ugly

Conclusion

Conclusion

Predictability through quantitative analysis

- Requires an architecture modeling notation with well-defined semantics
- Requires the ability to leverage existing analysis capabilities

Prediction of runtime behavior

- Requires modeling notation for embedded software systems
- Requires ability to represent dynamics of runtime architecture

Embedded systems with different architectures

- Require extensible modeling notation for analysis specific annotations
- Require analysis frameworks that span engineering views

SAE AADL for embedded systems modeling & analysis

- As industry standard allows for leveraged industry investment
- Provides a transition platform for university and industrial research
- OMG MARTE AADL profile provides UML migration path