

# Presentation

---

## Topic 4: Real-Time Simulation

# Outline

---

- ◆ Real-Time Simulation Terminology
- ◆ Purpose of Real-Time Simulation
- ◆ Revisit The Control Design Process
- ◆ Real-Time Simulations
- ◆ Real-Time Simulation Hardware
- ◆ Software Tools
- ◆ Summary

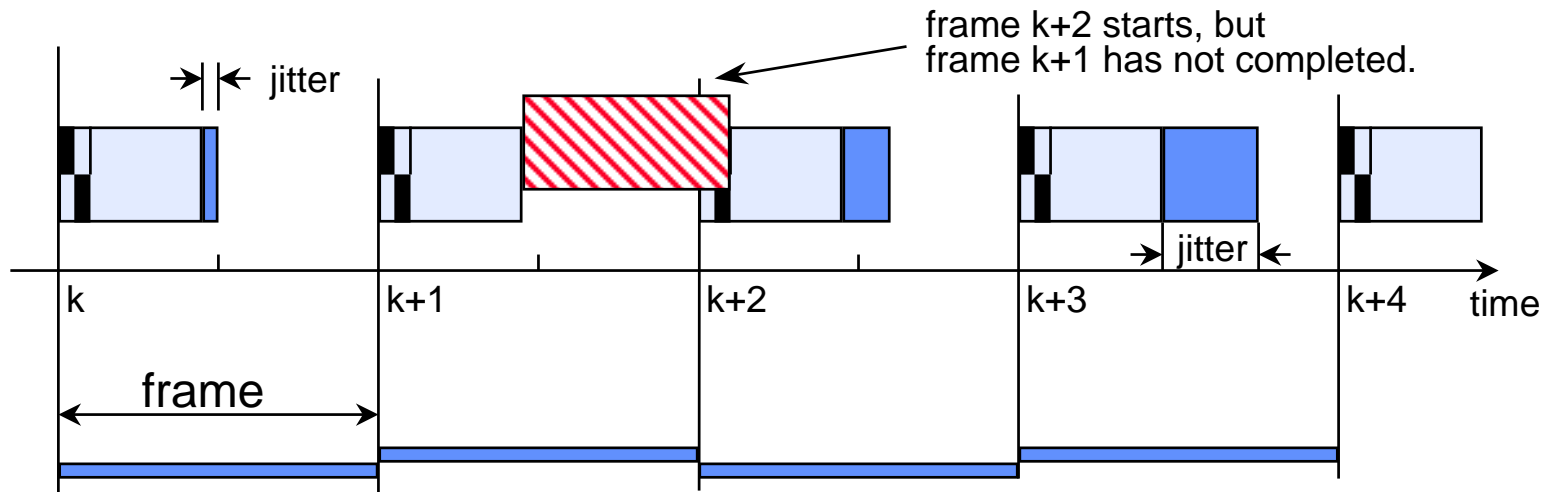
# Simulation Terminology

---

- ◆ Hard Real-Time: missed timing is a fault
- ◆ Soft Real-Time: missed timing is an annoyance
- ◆ Piecewise Linear Model - scheduled linear models
- ◆ Continuous - requires numerical integration
- ◆ Discrete - integration implicit. Already transformed.
- ◆ frame-time - window of allowable execution time, typically the minimum simulation time period
- ◆ overrun - execution beyond the end of the allowed time.
- ◆ jitter - task computational time variability

## Simulation Terminology, (continued)

- ◆ frame-time - window of allowable execution time.
- ◆ overrun - execution beyond the end of the allowed frame.
  - You can catch overruns when the interrupt service routine is already active when it is started.



frametime = interrupt time interval,  $T$

# Purpose of Real-Time Simulation

---

## ◆ Safety

- Evaluate control design prior to testing on real hardware
- Provide controllable platform for the analysis of final design
  - » Can walk/step through code.
  - » Can run in “scaled” real-time

## ◆ Save Money

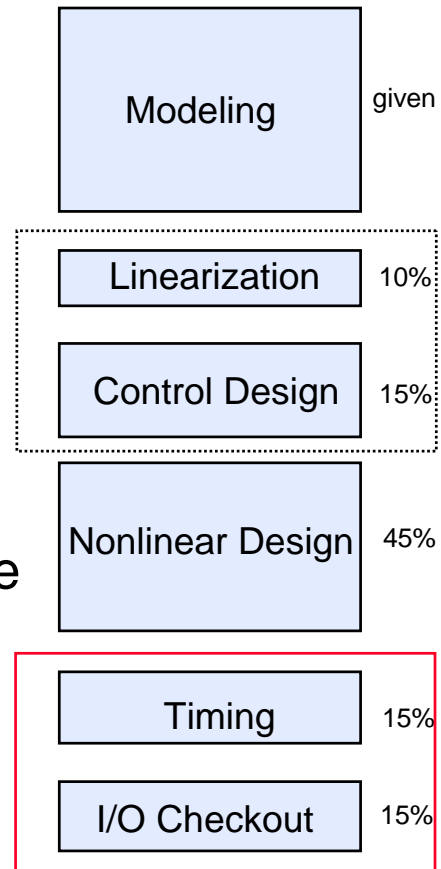
- Rapid Prototyping save time. Automates an iterative process.
- Avoid damage to real equipment due to design faults or errors

## ◆ Test Timing, discrete controller, and logic.

- Mode logic: STD = state transition diagrams

# Revisit The Control Design Process

- ◆ Model Creation & Validation
  - Simplified cycle deck plus dynamics
- ◆ Linearization
  - Generation of Design Point Linear Models
- ◆ Linear Control Design at design points
- ◆ Incorporation of Nonlinearities:
  - Operational and actuator limits, scheduling, mode logic, limit and integrator windup protection
- ◆ Real-time Simulation: check timing
- ◆ Real-time I/O Checkout: hard & software



percent time estimates assume you start with a validated dynamic model and experienced engineers working on all tasks

# Real-time Simulations Required

---

## ◆ Timing Analysis

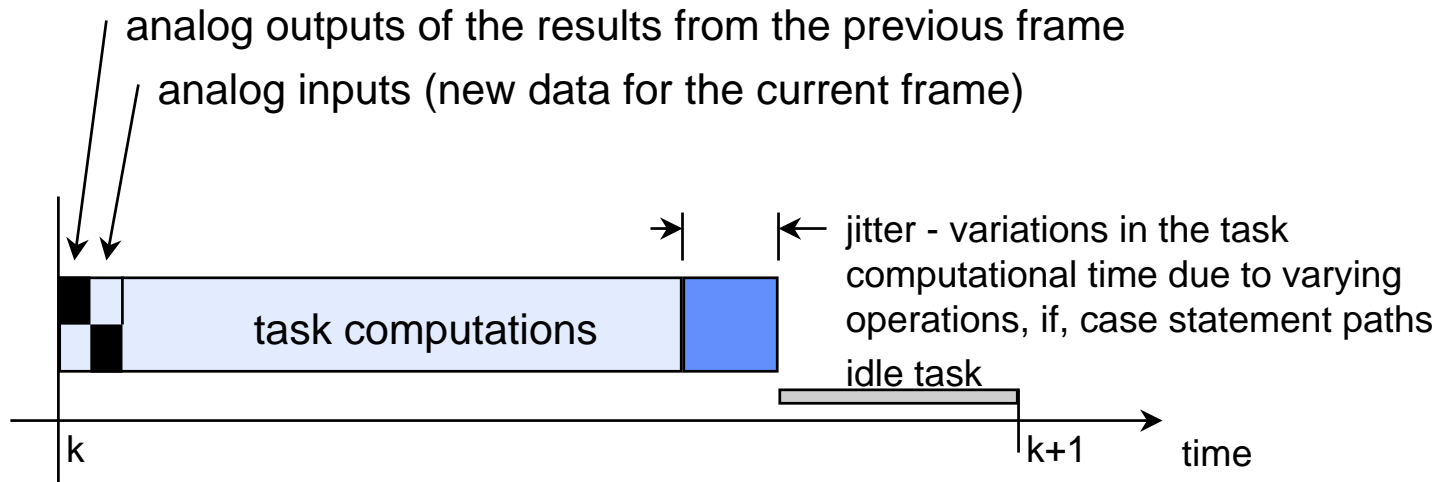
- Profiling to measure task execution
- Interaction of multiple tasks
  - » Main control tasks, Diagnostics tasks
- Interaction of task among multiple computers
  - » Synchronization

## ◆ Input/Output Checkout

- Checkout input/output hardware
  - » analog, digital, and frequency or counts
- “wet bench” (hydraulic) testing of actuators.

# Real-Time Simulations: Timing

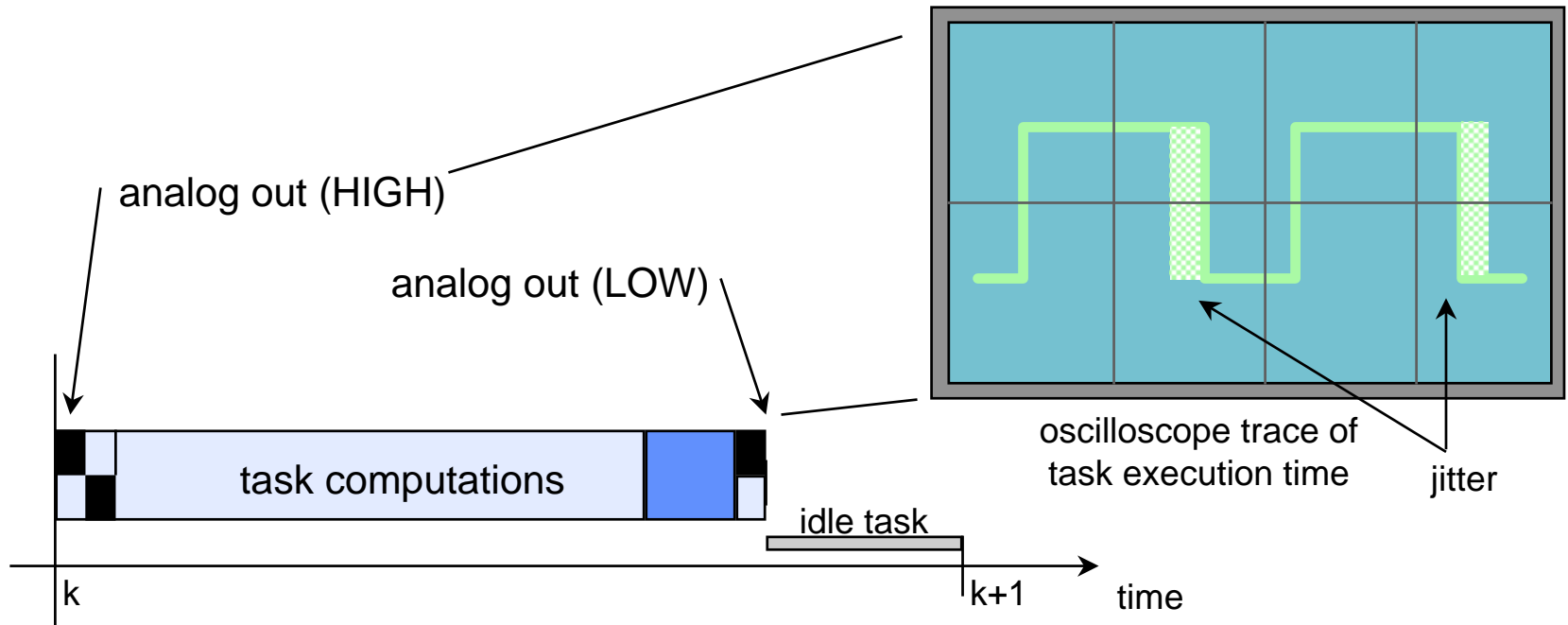
- ◆ Task is typically broken-up into the following pieces
  - Analog outputs, analog inputs, and computations
  - This sequence reduces the I/O jitter. Fixed sampling intervals.



If you write out the results of the computations at the end of the task, (analog output), the interval between outputs will vary due to "jitter". This will smear the frequency content of the signals.

# Real-Time Simulations: Timing

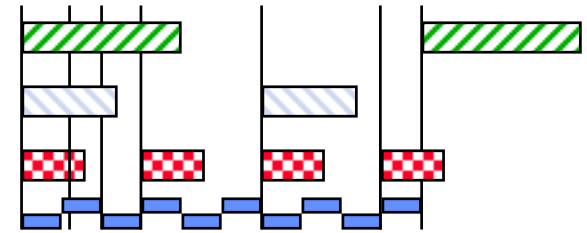
- ◆ Toggling an output at the beginning and the end of a task's computations can be used to profile the execution time.



# Real-Time Simulations: Timing of Tasks

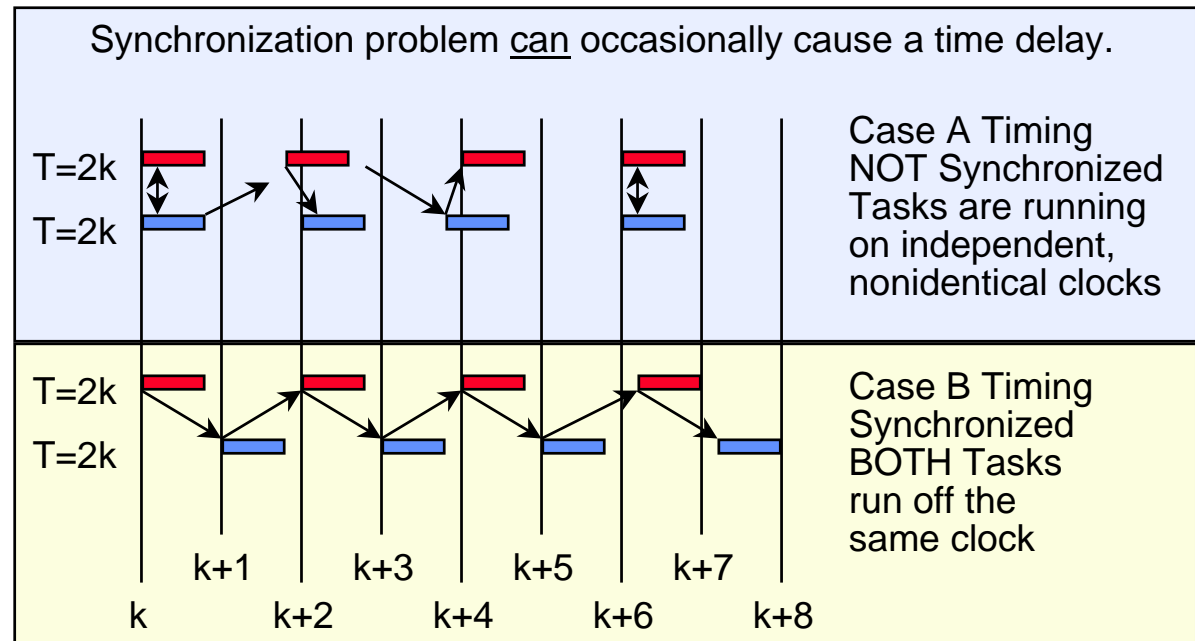
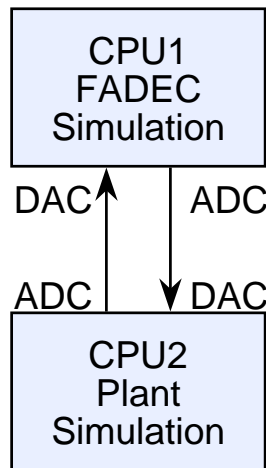
---

- ◆ Real-Time simulations of physical systems are the easiest real-time systems because all you really need is one periodic interrupt with a frametime that is the least common divisor of all the task frametimes.
  - Task 1 - frametime = 0.050 seconds
  - Task 2 - frametime = 0.030 seconds
  - Task 3 - frametime = 0.015 seconds
  - Least common denominator of 15,30,50 is 5.
  - So one interrupt service routine with a frame of 0.005 seconds can be used to manage and synchronize all the about tasks.
  - Have to manage the task priorities. If you only have two tasks, a “plant” and a “control” this becomes trivial.



# Real-Time Simulation: Multiple Computers

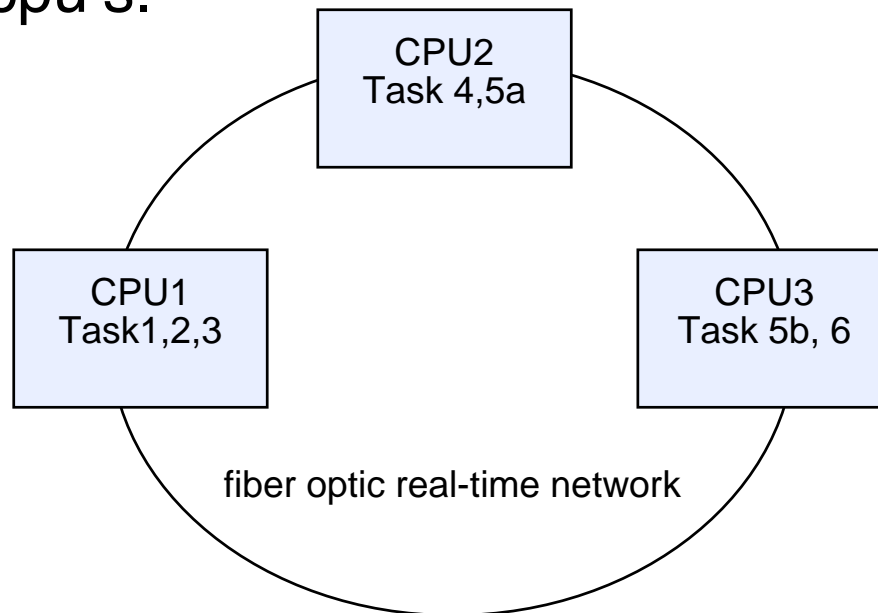
- ◆ Either to checkout hardware or for computational reasons, sometimes you want more than one computer.
  - Need to be careful with synchronization of timing.



# Real-Time Simulation: Multiple Computers

---

- ◆ Sometimes you need a network of computers.
  - Example: Integration of propulsion into flight control
    - » Replicated, Shared Memory Network (Systran Corp., Bit3,..etc)
- ◆ Split task between cpu's.
  - Task 5a, 5b



Use specialized processors for specific tasks (Signal Processing, CFD, FEA)

# Real-Time Simulation: I/O Checkout

---

- ◆ Address Variable Scaling and Scaling Routines.
  - example: +/- 10 volt range to 0-350 psig pressure
- ◆ Introduction of Hardware
  - analog, digital, frequency (counter).
    - » pressures, temperatures, rotor speeds
  - device drivers, interrupt service routine.
    - » communication between real-time code and hardware devices
  - “wet bench” (hydraulic) testing of actuators.
    - » fuel flow actuator
    - » compressor stator vanes
    - » nozzle actuators

# Real-Time Simulation: Hardware

---

- ◆ Low Cost (less than \$5000 U.S.)
  - PC/DOS extender (Watcom), PC/Linux (do it yourself)
  - Simulink Tools: Xanalog Realoop, Quanser's Wincon for 95
- ◆ Midrange (\$5000 U.S. - \$25000 U.S. )
  - MathWorks & DSPace (complete plug and play package)
  - Windows NT solutions (RadiSys, VenturCom, Imagination)
  - LP Elektronik GmbH, LP-RTWin Toolkit, (Germany)
- ◆ Expensive (\$25000 U.S. on up)
  - ISI: AC100 DSP solution (complete plug and play package)
  - Real-time Unix: Applied Dynamics International, Harris NightHawk, Concurrent Computer Corporation

# Real-Time Simulation: Hardware

---

## ◆ Issues

- If you have to use FORTRAN, a DSP solution will not work. DSP solutions only support “C”.
- The expensive “package” solutions are single vendor systems. They are typically not open systems. You have to buy from them and use their hardware and their software drivers. It is difficult to introduce new hardware to these systems. The advantages of these systems is that they are automated with little user intervention required in the process.
- The PC solution is certainly the cheapest hardware, but Windows 95/NT are not real-time operating systems.

# Real-Time Simulation: Windows NT

---

- ◆ NT is not a real-time operating system. It is not deterministic and should not be used for “HARD” real-time, hardware-in-the-loop applications.
  - SMI has used NT for “soft” real time with a 25ms framerate.
  - There are 3rd party solutions to modify NT for real-time.
  - [www.radisys.com](http://www.radisys.com), Radisys Corp. InTime
    - » based on old Intel iRMX operating system
  - [www.imagination.com](http://www.imagination.com), Imagination’s HyperKernel
    - » modification to NT Hardware Abstraction Layer (HAL)
  - [www.venturcom.com](http://www.venturcom.com), VenturCom, Realtime Extension
    - » modification to NT HAL and Kernel

# Software Tools

---

- ◆ The MathWorks real-time Workshop (\$10,000 U.S.)
  - Separate real-time and accelerator products. Accelerator generates code you can link back into Simulink.
  - Available device drivers appear as icon's in Simulink.
  - Supported PC compilers: Watcom V11 and MicroSoft V5
  - Generated code use the MathWorks Simulink simulation structure: Simstruc. You must use this structure or object to use this code. This is different than the code generated by ISI's AutoCode product. AutoCode generates C or Ada.
  - Built in Support for DOS, VxWorks, DSPace
  - Generated code for Discrete or Continuous Models
  - NOT ALL Simulink blocks can be converted to code.
    - no absolute time blocks. Signal Generators, Continuous Time Delay

# Summary

---

- ◆ Frame-Time and Overrun defined
- ◆ Real-Time Simulation primarily for Safety
  - Rapid Prototyping speeds iterations and saves money.
- ◆ The Control Design Process, revisited again
- ◆ Timing issues, computer-to-computer configurations and communication
- ◆ Real-Time Simulations Solutions
  - 95/NT is not a real-time operation system
- ◆ Software Tools:
  - The MathWorks RTW works but has its own quirks.