ILLINOIS INSTITUTE OF TECHNOLOGY

Transforming Lives. Inventing the Future.

www.iit.edu

Software Engineering
CS 487

Prof. Dennis Hood

Computer Science

Lesson Overview

- Embedded Systems
- Reading
  - Ch. 19 – Systems Engineering
  - Ch. 20 – Systems of Systems
  - Ch. 21 – Real-time Software Engineering
- Objectives
  - Analyze the requirements and design challenges of embedded software systems
  - Discuss approaches for architecting embedded solutions
  - Consider the factor of timing in embedded systems and examine approaches for testing proper system operation with respect to timing constraints

Week 11
Embedded Software

Case Study: Traffic Control System

- Priorities
  - Maintain safety / prevent accidents
  - Optimize flow of vehicles and pedestrians
- Partners
  - Lights and other display elements
  - Sensors: in-road, pedestrian buttons, motion sensors, cameras, other?
- Exceptions
  - Power outages
  - Human error
- Assumptions
  - People know the rules and will follow them
  - People have high SA regarding the signaling

Embedded Systems

- Characteristics
  - Software embedded in hardware
  - Minimal user interface / interaction
  - Real-time response to environmental change
  - A product of the environment
- Performance
  - Proper performance means a correct response within an acceptable time
  - Availability is expected
  - Safety and reliability issues are prominent in design decisions

Design Considerations

- Platform selection
  - Timing, power, environment, OS, etc.
- Stimuli response
  - Periodic vs. aperiodic stimuli
- Timing requirements
- Process architecture
  - Aggregate stimuli and response processes
- Algorithms to support computations
- Data architecture
- Process scheduling

Process Coordination

- Processes in a real-time system have to be coordinated and share information
- Process coordination mechanisms ensure mutual exclusion to shared resources
- When one process is modifying a shared resource, other processes should not be able to change that resource
- When designing the information exchange between processes, you have to take into account the fact that these processes may be running at different speeds

Real-time System Modelling

- The effect of a stimulus in a real-time system may trigger a transition from one state to another
- State models are therefore often used to describe embedded real-time systems
- UML state diagrams may be used to show states and state transitions in a real-time system

Real-time Design Patterns

- Observe and React
  - Sensors are monitored and status displayed
  - Sensor change initiates handler
- Environmental Control
  - Sensors monitor the environment and actuators adjust it
  - Sensor change initiates signals to actuators
- Process Pipeline
  - Data transformation is required before processing
  - Separate processors to handle concurrent transformations

Timing Analysis

- The correctness of a real-time system depends on both the correctness of the outputs as well as the time at which they are produced
- Timing can be difficult with a mixture of periodic and aperiodic stimuli and responses
- Key factors in analyzing timing requirements:
  - Deadlines – the time by which stimuli must be processed
  - Frequency – the number of times per second that a process must execute
  - Execution time – the time required to process a stimulus and produce a response (consider both average and worst-case)

Example: Real-time Operating Systems

- Real-time applications often require a more efficient and responsive real-time operating systems
- RTOS components:
  - A real-time clock for periodic events
  - An interrupt handler for aperiodic requests
  - A scheduler for examining and selecting processes for execution
  - A resource manager for allocating system resources such as memory
  - A dispatcher for starting the execution of processes