CS 5372
Specification and Design of Real-Time Systems

Dr. Salamah Salamah
CCS 3.0608
isalamah@utep.edu
915-747-6671

1

Real-Time - Introduction

- Today's Outline
  - Class intro (Syllabus, rules, and so on)
  - Overview of and introduction to real-time software systems
  - Host/Target Connection and Setup Lecture & Demo

Slides adapted from A. Kornecki, Embry Riddle Aeronautical University

2

Introductions

- Tell us
  - Your name
  - Level (PhD, master, undergrad)
  - What are you looking for in the course
  - Something else (anything)

3

Course Syllabus

4

Real Time – Historical Perspective

- **Replacement** of analog processing section of control systems by their digital equivalent
- **Progress** and miniaturization of computing hardware (minicomputers, microprocessors, microcontrollers)
- **Application** of computers in military and aerospace domains
- **Research** in operating systems, scheduling, concurrency, exception handling, reliability, and safety
- **Proliferation** of computers in all areas of science, technology, and everyday life

5

Modern computer applications (characteristics): In groups

- Have predictable and guaranteed timing behavior - they fail if the timing constraints are not met **(real-time)**
- Interact with and control the environment **(safety critical)**
- Operate continuously as a part of a larger system **(embedded)**
- We use the term real time software - designed to respond to and control the dynamic environment of a real system
- Another term: **cyber-physical systems**

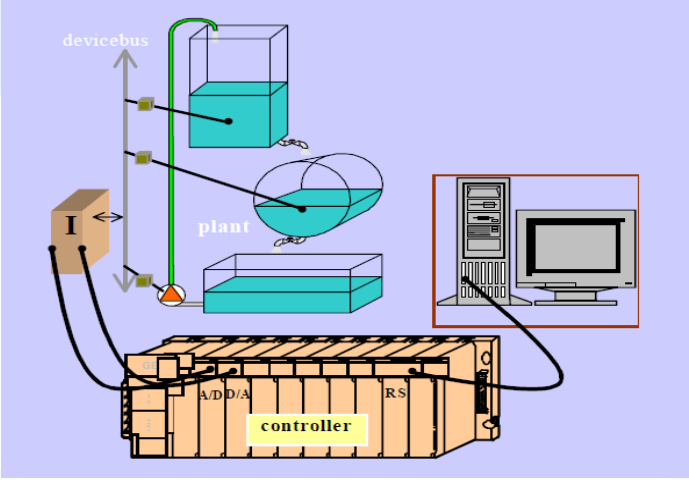
6

Application Domains

- **process control** (chemical industry, food processing)
- **robotics** (manufacturing, automated control)
- **avionics** (flight management, GPS)
- **aerospace** (jet engine control, fly-by-wire)
- **military** (weapon management, encryption)
- **data collection** (acquisition, signal processing)
- **communication** (fax machines, digital phones)
- **appliances** (microwave dishwasher thermostats)
- **automotive** (engine/cruise control, anti-lock brakes)
- **computer peripherals** (printers, terminals, modems)

7

Typical Industrial Control



8

Real Time System Layers/Viewpoints

- **Specification and Design**
  - theory, methodologies, CASE Tools, notations, process
- **Programming**
  - code, synchronization, communication, threads, development support (programming languages, compilers), run-time support (real-time kernels)
- **Operating System**
  - scheduling, thread management, memory management, interrupt handling, input/output, application programming interface (API), priority inheritance avoidance
- **Hardware**
  - Processor, bus, peripheral devices, memory, cache

9

Real-Time System Platforms

- Simple micro-controller chip *(car engine, copier, microwave)*

- Single controller board (industrial robot)

- Complex controller boards on a bus *(space data processing, aircraft avionics)*

10

Real-Time System Platforms (cont.)

- Dedicated workstation *(simulation, graphics, medical device)*

- Distributed workstations *(air traffic control)*

- Networked supercomputers *(military, nuclear physics)*

11

So what exactly does Real-Time mean? In groups

- Logical/functional correctness
- Timing correctness
- Which is more important?
- Hard vs soft real-time (later in the lecture)

12

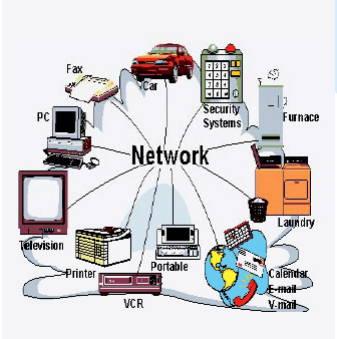
What is a Real-Time System ?

- RT system is a computer system that must satisfy **explicit response time constraints** (*the system correctness depends on the time at which the results are produced*)
- Dynamic systems are described by their state (or mode of operation)
- The system state changes in response to input event (interrupt, signal)
- Change of state requires time …
- Popular misconception:
  - *"real-time systems is a very fast system":*

13

Embedded System ?
In Groups

- a dedicated computer system designed for specific purpose
  - A **dedicated** real-time computer system providing control and computation as a part of a complete system
  - A computer **built into** a system and not seen by the user as a computer
  - An **execute-only** target environment running correct, reliable and safe software
  - A system developed using **host-target** paradigm



14

Embedded Systems

- Embedded systems features:
  - majority are reliable and predictable
  - some must operate in real-time
  - some are safety critical
  - … many are the combination of above
- Usually an invisible part of other systems or devices (systems within systems):
  - NASA Mars Path Finder
  - Lockheed Martin's Missile guidance system
  - Several modern automobiles
  - Household items
  - Network environment
  - Games

15

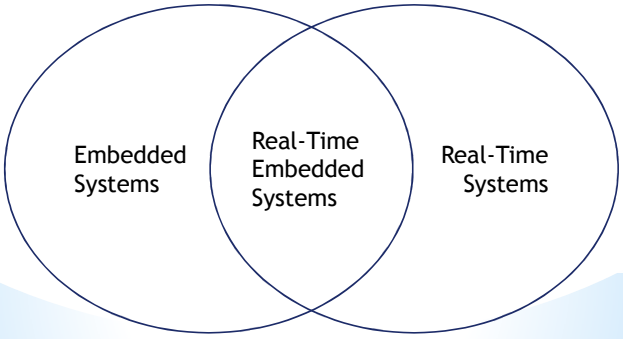
Embedded System Design Consideration

- Expected Performance:
  - Response time
  - Throughput
  - Reliability
  - Predictability
  - Safety
  - Faults/Failures
- Required Interface:
  - Types of signals
  - Range of signals
  - Frequency of signals
  - Communication protocols
  - Connectors standards
  - Sensors
  - Switches
  - Displays
- Environmental Factors:
  - Physical: weight, size, life span, maintainability
  - Electrical: power supply, electrical interface
  - Operational conditions: humidity, temperature, vibration

16

Real-Time Embedded Systems

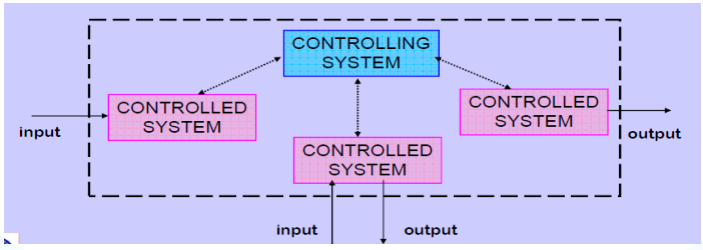
- Embedded systems that respond to events in timely fashion
- Knowledge of the hardware is a must
- Usually cross-platform development (Cross compilers)



17

Structure of Real-Time Embedded Systems

- A controlling system and at least one controlled system
- Interaction between controlling and controlled systems:
  - Periodic – predefined intervals
  - Aperiodic – no defined intervals
  - Combination of above



18

Examples (1):

- Weapon defense system protecting a naval destroyer by shooting the incoming missiles
  - Command-and-Decision System (controlling/controlled system ?)
  - Radar System (controlling/controlled system?)
  - Weapon Firing Control System (controlling/controlled system?)



19

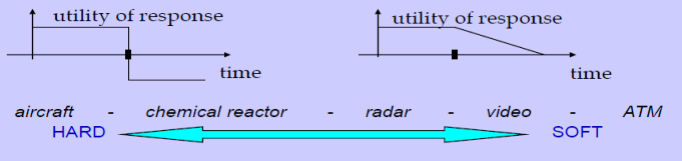
Characteristics of RT systems

- Must produce correct computational result
  - Functional or logical correctness
- Must conclude and produce result within a predefined period
  - Timing correctness
- Each component of a RT system MUST have a deterministic behavior

20

Hard vs. Soft Real-Time

- HARD RT - does specify the time at which each task must finish *(e.g. actuation of flight surfaces in a flight control system)*
- SOFT RT - does specify level of urgency *(e.g. missing data in a data acquisition system)*
- "what differentiates hard RT systems and soft RT systems are the degree of tolerance for missed deadline, usefulness of computed results after missed deadlines, and severity of the penalty incurred for failing to meet deadlines"
  - Hard RT – penalty is catastrophic
  - Soft RT – penalty is a degraded performance
- The issue is not the speed but the impact of missing the deadline



21

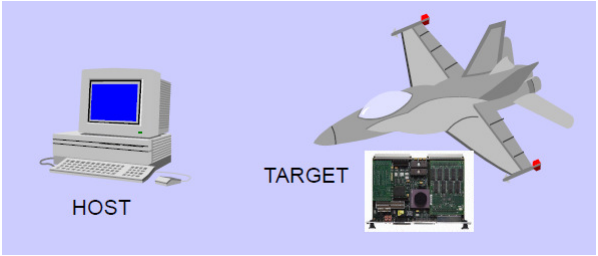
Nature Of Real-Time Programming

- In Real Time programs the input **events** arrive **asynchronously** from the outside world requiring timely **response**
- The program is able to **interrupt** current thread of execution and instead execute some pre-defined code to respond to the input
- The executed code, in turn, may **activate** some other programs that were **waiting** for that input
- The system is able to **resume** its previous activity
- The system fails if the **timing constraints** are not met
- Dynamic **tasks** for various events, entities, and functions
- Tasks are activated **periodically** or **sporadically**
- Task **synchronization** and **interaction is required**
- Timely **completion** of each task before the **deadline is required**
- System shall **not hang/crash**
- Guarantee of timing behavior makes the system **predictable**
- The system must be **reliable** (not fail) and **safe** (not hurt or cause any loss)

22

The Development Environment

- **Native** environment: development on the same platform as the resulting executable
- **Host/target** environment: development on the host with the executable downloaded to target



23

Patriot Missile

http://www.ima.umn.edu/~arnold/disasters/patriot.html

- On February 25, 1991, during the Gulf War, an American Patriot Missile battery in Dharan, Saudi Arabia, failed to track and intercept an incoming Iraqi Scud missile

- The Patriot battery had been up with no reset over 100 hours

24

Patriot Missile – What you will learn in this class

- The prediction of where the Scud will appear next is a function of the velocity and the time of the last radar detection
- Velocity is a real number (e.g., 3750.2563…miles per hour)
- Time has been kept by the system's internal clock in tenths of seconds, expressed as an integer and thus it must be multiplied by 1/10 to get time in seconds
- 24 bits error is 0.00000005960464477539 06
- After 100 hours of operations without reset the resulting time **drift error** was enough to fail the intercept - due to the magnified chopping error of 24-bit calculations

25

Mars Mission

- On July 4, 1997, the Mars Pathfinder with Sojourner rover landed on the surface of Mars and rather than executing tasks started resetting computer
- Software with over 150,000 lines of code was written by seven JPL engineers over three years using Wind River Systems tools - VxWorks being the first commercial RTOS to operate on another planet

26

Mars Mission
What you will learn in this class

- Priority multitasking allows a higher-priority task preempt a lower-priority task
- It is theoretically possible that a mid-level-priority task can interfere with a higher-priority task causing the system to time out and reset
- This condition, called **priority inversion**, can be prevented by a *priority inheritance* protocol, which is a user-selectable feature of VxWorks
- Engineers at JPL sent commands to "flip" one bit and thus turn on the priority inheritance for the critical semaphore - after that, the system worked flawlessly

27

Categories of Real-Time System Requirements

- Functional
  - Data collection
  - Control
  - Interface
  - …
- Temporal
  - Deadlines
  - Jitter
  - Latency
  - …
- Dependability
  - Reliability
  - Safety
  - Maintainability
  - Availability
  - Security
  - …

28

Definitions: Event/Interrupt

- **EVENT**
  - an instantaneous *(time-point)* internal or external occurrence that causes the program to branch
- Event types:
  - **Synchronous/Periodic**
    - at predictable time points (e.g. clock driven)
  - **Asynchronous**
    - at unpredictable time points (e.g. due to external conditions)

29

Definitions: Event/Interrupt

- **INTERRUPT**
  - an externally or internally generated event to allow for the disruption of a processor's normal execution
  - handled by the microprocessor hardware interrupt interface
- **INTERRUPT LATENCY**
  - time required to recognize and start responding to an interrupt

30

Definitions: Timing/Determinism

- **RESPONSE TIME**: time interval between presentation of inputs *(stimuli)* and the appearance of the associated outputs *(response)*
- **DEADLINE**: a time point before which the task must complete the execution (*or a specific event must occur*)
- **DETERMINISTIC SYSTEM**
  - for each state and set of inputs a unique set of outputs can be determined
- **TEMPORAL DETERMINISM**:
  - the response time is known (or bounded) for each set of outputs
- **PREDICTABILITY**:
  - the property of meeting the temporal determinism criteria

31

Definitions:
Task, Schedulability, Process, Thread, Kernel

- **TASK**
  - a unit of a sequentially executing program designed to fulfill a specific system function
- **SCHEDULABILITY**
  - a property of a set of tasks ensuring that all tasks will meet their respective deadlines
- **PROCESS**
  - a virtual computing environment set up to run as a program (contains its own data, code, context, & resources)
- **THREAD**
  - a sequence of instructions executed within the context of a process
- **KERNEL**
  - a core component of run-time support (or operating system) residing continuously in main memory

32

Definitions:
Failure/Fault and Dependability

- **FAILURE**
  - when the system fails to perform its required function in the operational phase
- **FAULT**
  - a manifestation of error in system (may cause failure) from the system engineering perspective – the requirements are not met … but what about incomplete or faulty requirements?
- **DEPENDABILITY**
  - the property of the system that justifies reliance on its services (**Quality of Service – QoS**)
  - Dependability defined as an overarching concept including all other "-ities"

Source: "Dependability: Basic Concepts and Terminology", Edited by Laprie, J.-C., Springer Verlag, 1992, ISBN: 3-211-82296-8

33

Definitions:
Safety/Reliability

- **SAFETY**
  - a property of a system that it will provides hazard-free operation
- **HAZARD**
  - the capability of the system to harm the people, destroy the property or environment
- Are reliability and safety the same thing?
  - **RELIABILITY** - probability that the system will function correctly over a given period of time
  - Reliability and safety are NOT identical
  - **Reliability** is a **bottom-up** activity focusing on system **failures**
  - **Safety** is a **top-down** approach concentrating on system **hazards**

34

Conclusion

- Embedded real-time systems are here to stay:
  - Expanding application domain
  - Advances in technology
  - Globalization
  - Connectivity, internet
  - Smaller and smarter products
  - Expanding marketplace
- Practical experience with real-time system development is of great value to your future employers in many areas of industry
- Industry is in a desperate need for real time software developers (IAB)
- Real-time has place in computing curricula
- Enforcing good software engineering methodology and process is critical

35

Assignment for Next Week

- Read Chapters 1-4 of RTCES
- Install workbench 3.3 on your own laptop and bring to class.

36