Advanced System Software
（先端システムソフトウェア）

#1 (2018/9/27)

CSC.T431, 2018-3Q
Mon/Thu 9:00-10:30, W832

Instructor: Takuo Watanabe（渡部卓雄）
Department of Computer Science

e-mail: takuo∅c.titech.ac.jp
https://titech-aos.github.io
ext: 3690, office: W8E-805

Agenda

- Class Introduction
- Programming Project
- Basics of Embedded Systems

- Course Material (slides, code, etc.)
  - Lecture Slides
    - OCW-i
  - Sample Code & Tutorials
    - https://titech-aos.github.io

About This Class

Overview

This course presents an overview of time as it relates engineering complex software systems including embedded systems. In this course basic concepts, terminology, and issues of embedded/real-time systems are examined.

Objective

The course aims to ensure that students have not only acquired the knowledge and technique required for embedded/real-time system development, but also gain ability to formally model/verify simple real-time systems.

Goal / Keywords

By the end of the course you should be able to

(1) define what it means to be a real-time system,
(2) design and develop simple embedded/real-time programs on RTOS,
(3) discuss timing, scheduling and related properties, and
(4) formally model and verify simple real-time systems.

Keywords

Embedded Systems, Real-Time Systems, Real-Time Operating Systems, Scheduling, Schedulability Analysis, Verification

Prerequisites

- Undergraduate Level Knowledge
  - Programming
    - C/C++ Programming
      - Processes, Threads
      - System APIs
  - Operating Systems
    - Basic OS Structures, Processes/Threads, Memory Management, File Systems, I/O etc.
  - Theoretical CS
    - Finite Automata
    - Propositional & Predicate Logic
- PC for running software tools
  - Mac, Linux, Windows

Schedule

1	Sep. 27	Course Introduction and Overview, Overview of Embedded Systems
2-3	Oct. 1, 4	Programming Embedded Systems
4-7	Oct. 11, 15, 18, 22	RTOS (Real-Time Operating Systems)
8-10	Oct. 25, 29, Nov. 1	Real-Time Systems
	Nov. 5, 8	No Class
11-13	Nov. 12, 15, (19)	Advanced Topics
14-15	Dec. 6	Project Demo

Workload and Grading

- Lectures
- Assignment 1: 50%
  - Report on Real-Time Systems
    - Schedulability
    - Specification / Verification
- Assignment 2: 50%
  - Team Programming Project
    - Develop a program using RTOS
    - Demonstrate the developed program

Agenda

- Class Introduction
- Programming Project
- Basics of Embedded Systems

Programming Project

- Develop a program running on M5Stack
  - Team Project
  - Evaluation: code and demo

- About M5Stack
  - M5Stack is an open-source modular toolkit for IoT devices.
  - An M5Stack module is an enclosed device that consists of an ESP32 SoC and several peripheral components, including three buttons, an LCD, and a speaker.
  - http://www.m5stack.com

Project Demo

- Date, Time & Place
  - December 6 (Thu) 15:05-18:20
  - W8E, 10F faculty meeting room
- Prerequisite
  - Use M5Stack
  - Use RTOS features in FreeRTOS
  - Use at least one peripheral in the module
  - Please let me know If you want to use extra peripheral devices other than on-board components

Preparation

- By the next lecture (10/1), each student who plans to enroll in this course should:
  - officially enroll the course
  - join the slack workspace "titech-aos"
    - send me an e-mail ASAP containing:
      - Name, Student Number, Department and Lab.
  - form or join a development team
    - # members in a team should not exceed 5
    - # of teams ≤ 20 (# of development kits)
  - setup your PC
    - Install a development environment

Agenda

- Class Introduction
- Programming Project
- Basics of Embedded Systems

Embedded Systems

- An embedded system is a computing device that is part of a complete system often including hardware and mechanical parts.
  - The presence of the device is often not obvious to the user of the entire system.
- An embedded system is usually designed to perform one or a few dedicated functions; e.g., controlling and/or monitoring the system.
- An embedded system is usually a (real-time) reactive system.

Examples of Embedded Systems

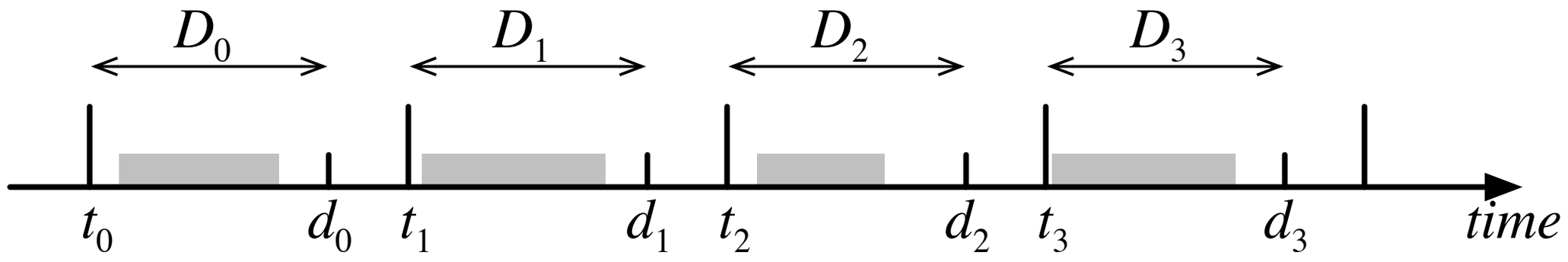
- Consumer digital devices
  - mobile phones, MP3 players, digital cameras, handheld game consoles, electronic musical instruments
- Home/Office appliances
  - TV sets, STBs, video game consoles, microwave ovens, dish washers
  - photocopiers, printers, video projectors
- Transportation
  - space shuttles, airplanes, trains, automobiles
- Public Infrastructure
  - nuclear plants, telecommunication
- Others
  - artificial satellites, weapons

Real-Time Systems

- A real-time system is a computer system that requires not only its computing results to be correct but also the results to be produced within specified *deadlines*.
- Results produced after the deadline has passed may be of no (or less) value, even if they are correct.

Deadlines

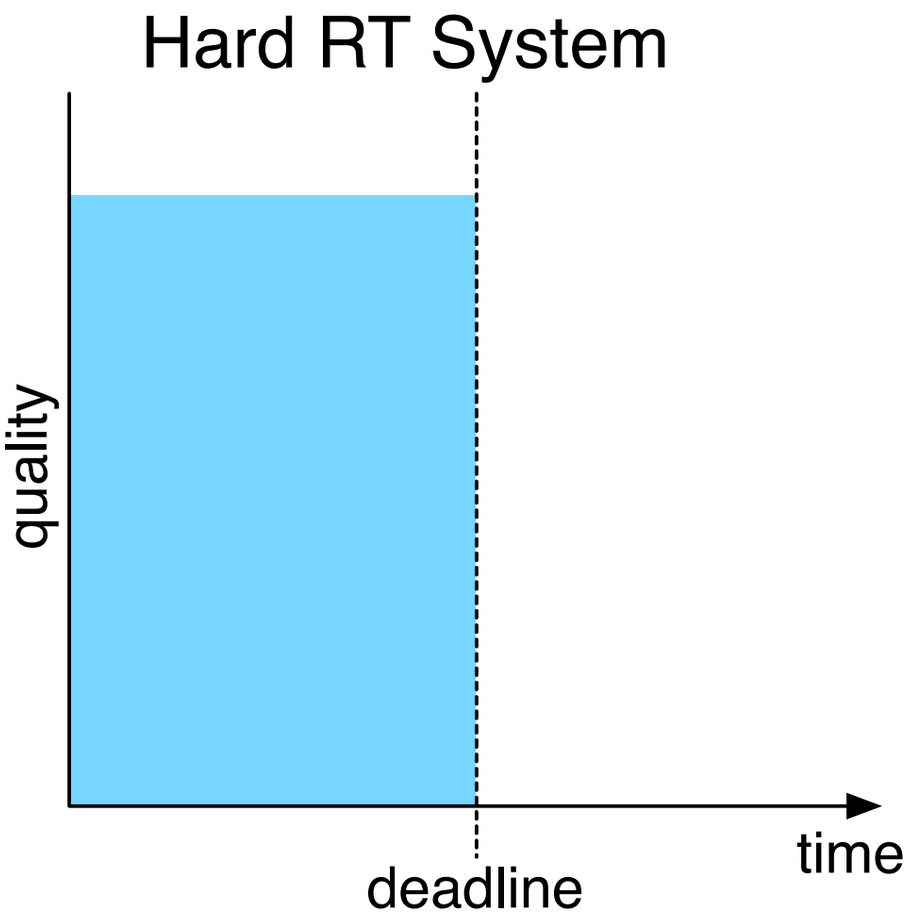
- *Relative* (D_i)
  - the amount of time in which the system needs to produce the result
- *Absolute* (d_i)
  - the precise point in time at which the system must produce the result

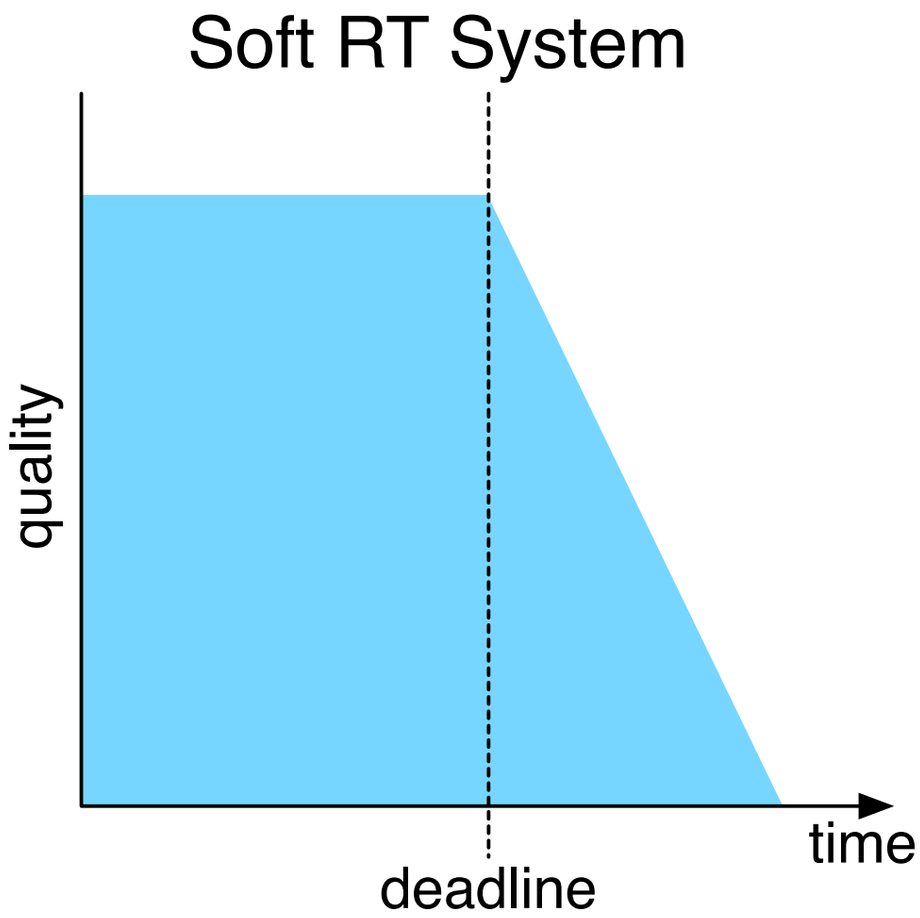


Taxonomy of Real-Time Systems

- Hard Real-Time Systems
  - A deadline miss results in an incorrect system.
- Firm Real-Time Systems
  - Infrequent deadline misses are tolerable, but degrade the quality of service (QoS).
  - Task instances that missed their deadline have no value.
- Soft Real-Time Systems
  - Deadline misses result only in a decreased QoS.
  - Task instances that missed their deadline have less value.

The Value of a Task



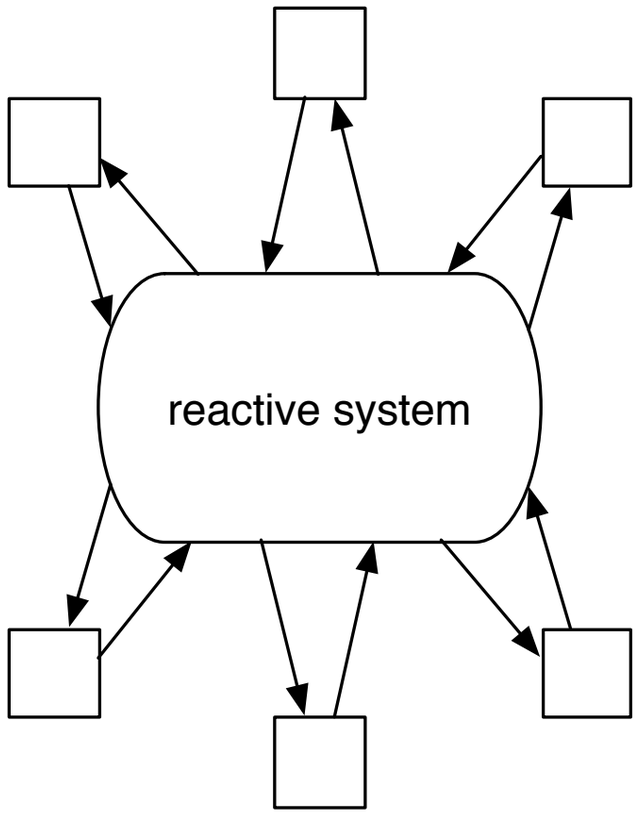


Reactive Systems

- A reactive system is a computing system that interacts with its environment.
  - A reactive system is not supposed to stop but should be continuously ready for interactions.
  - A real-time system is often a reactive system.

- cf. Transformational Systems
  - A transformational system accepts an input and then produces an output and stops.
  - Classical model of computation (function)

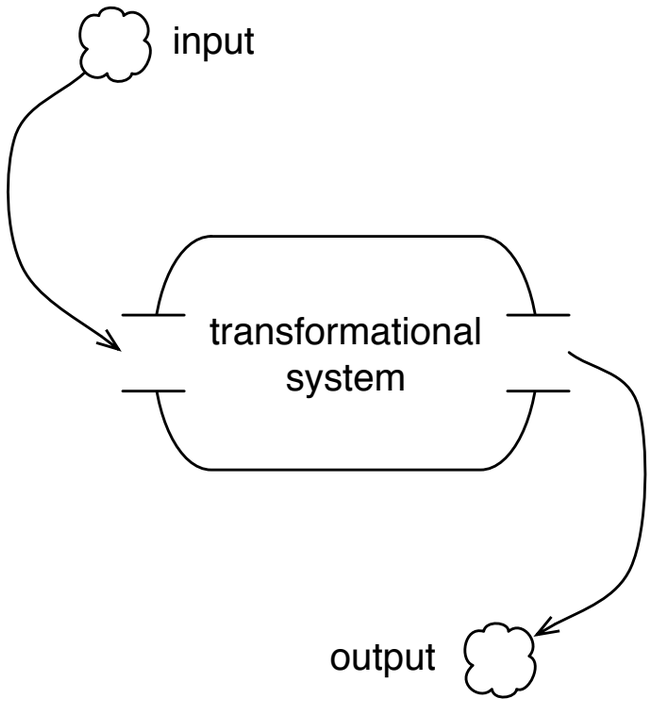
Reactive System



- A computing system that interacts with its environment
  - A reactive system is not supposed to stop but should be continuously ready for interactions.
  - A real-time system is often a reactive system.

- Cf. transformational system

Transformational System



- A system that accepts an input and then produces an output and stops
- Classical model of computation (function)

Programming Embedded Systems

- Blinking an LED (Arduino)

```
void setup() {
  pinMode(LED_BUILTIN, OUTPUT);
}

void loop() {
  digitalWrite(LED_BUILTIN, LOW);
  delay(1000);
  digitalWrite(LED_BUILTIN, LOW);
  delay(1000);
}
```

Handling Inputs using Polling

```
#include <M5Stack.h>

void setup() {
    M5.begin();
    M5.Lcd.fillScreen(TFT_BLACK);
    M5.Lcd.setTextFont(4);
    M5.Lcd.setTextColor(TFT_CYAN);
}

void loop() {
    if (M5.BtnA.wasPressed())
        M5.Lcd.printf("A");
    if (M5.BtnB.wasPressed())
        M5.Lcd.printf("B");
    if (M5.BtnC.wasPressed())
        M5.Lcd.printf("C");
    M5.update();
}
```

Summary

- Class Introduction
- Programming Project
- Basics of Embedded Systems
  - Basic Concepts
  - Real-Time Systems
    - Relative/Absolute Deadlines, Hard/Soft RT Systems
  - Reactive Systems
  - Simplest Programming Patterns