

Project Development & Software Design

Lecturer: Annie Guo

Lecture Overview

- Basic project development steps
- Some software design techniques
- Example

Project Steps

- A project development basically consists of a number of stages:
 - Project definition
 - Design
 - Hardware implementation and Software development
 - Test

Project Definition

- Includes:
 - Project requirement analysis
 - Development of a complete operational specification and coarse or macro-level block diagram
 - Possible project proposal
- Time spent: 10~15% of the total project time.

Design

- Determines
 - Hardware design
 - Hardware components required and their connections
 - Overall system structure
 - Hardware+software
 - Software design
 - Software data structures and flowchart
 - Hardware and software interaction
 - Basic tasks and their interactions
- Takes 40-50% project time

Software Design

- Consists of a number of steps:
 - Listing tasks to be performed
 - Prioritizing the tasks and scheduling tasks if required
 - often it is the case in the embedded system design
 - Designing interrupts for task scheduling if required
 - Creating a software flowchart
 - Key components and tasks
 - Writing code

Test

- The system developed in the project is tested in accordance with the operational specification developed in the project definition stage.

Example

- Lift Control Project
 - Refer to “Project Example” on the course website

Example (cont.)

- Project definition
 - It basically has been done (in the course of this kind)
 - Hardware resources: AVR lab board
 - Operational specification
 - But you have freedom to add more, especially for enhancement

Example (cont.)

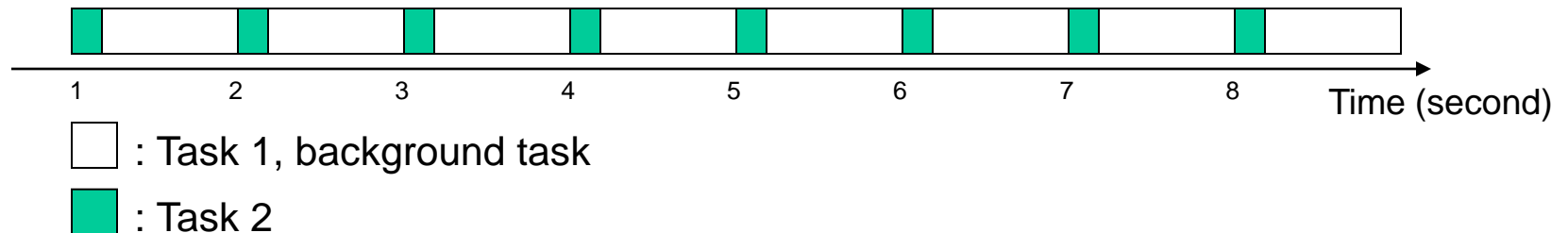
- Hardware design and implementation
 - The lab board is given, the basic hardware design and implementation has been done for you.
 - What you need to do is to design how to use the available hardware resources
 - As specified
 - Using LED and LCD to display the status of the lift operation
 - Inputs: Encoding input keys of the keyboard and push buttons to represent all possible input requests
 - Additionally
 - Using LCD and LED bars to display values for the debugging purpose

Example (cont.)

- Software design
 - How many tasks are there?
 - Depending on your design and how you like your code organized.
 - One basic function → one task
 - A group of related functions → one task
 - Example: two basic tasks (could be more)
 - Task 1: Getting the service requests
 - Task 2: Controlling the lift operations: moving up, moving down, stop

Example (cont.)

- Software design
 - Are there any task priority issues?
 - Yes, Task 2 has a higher priority as specified in the project description – real time operation
 - If there are requests, the lift going from one floor to the next floor takes **two seconds**; If there is a stop, during traveling, the stop takes **one second**.
 - a timing diagram for the task schedule

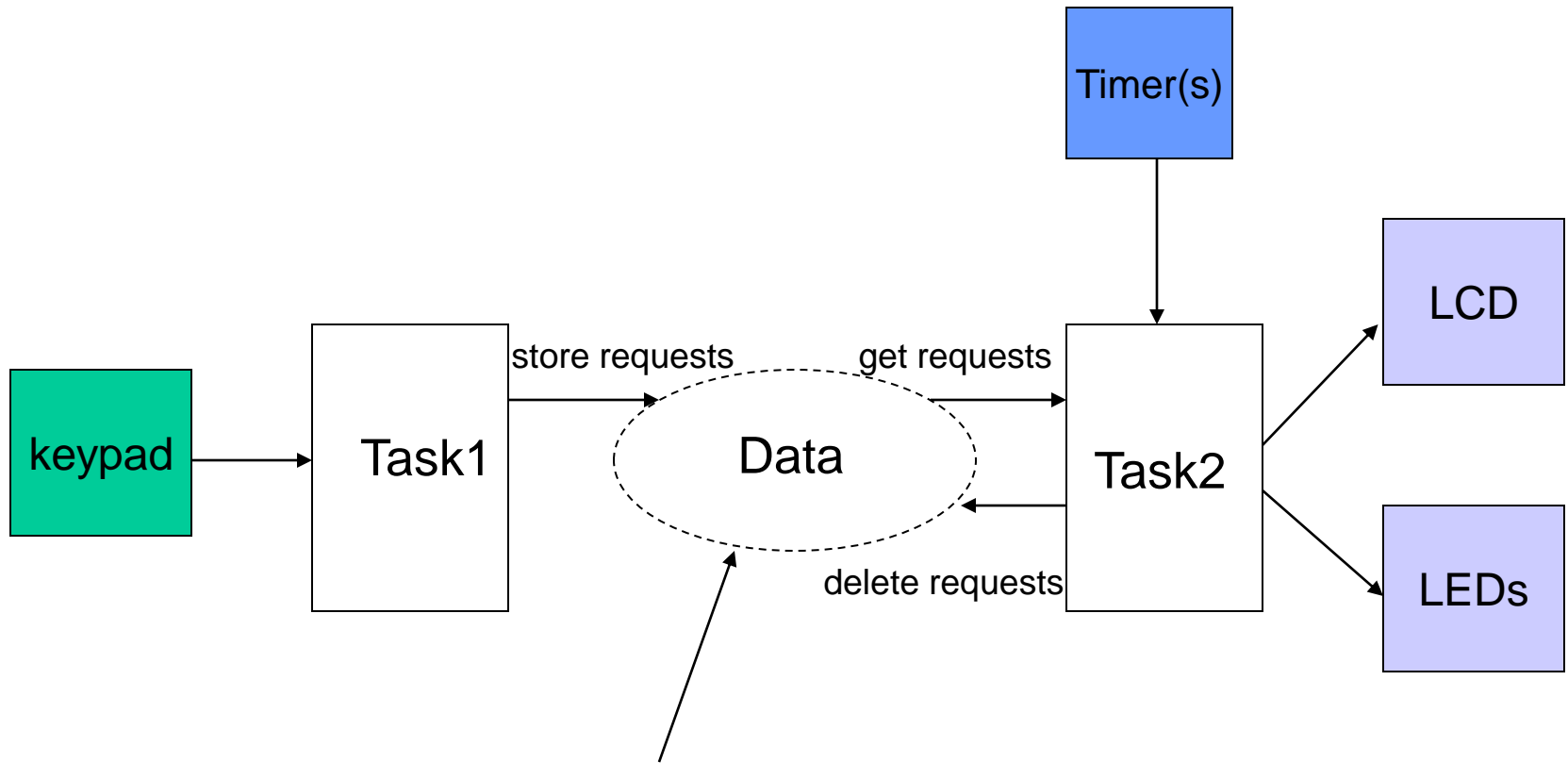


Example (cont.)

- Software design
 - How to meet the timing requirement?
 - By software approach
 - Execution of a certain number of instructions to form a delay period
 - By the timer interrupts
 - Often preferred
 - » reliable, accurate, easy, and
 - » CPU efficient

Example (cont.)

- Software design
 - How do the two tasks interact with each other?
 - Task 2 needs to know the requests from Task 1
 - Task 2 needs to delete requests that have been processed.
 - How do the two tasks interact with input/output devices?
 - Task 1 keeps polling input requests from the keypad
 - Task 2 outputs the control for lift status display.
 - An overview block diagram may be helpful for a complicated design.
 - An example is given in the next slide



How data is stored?

How the status of lift is represented?

Example (cont.)

- Software design
 - Data structures:
 - How information is represented?
 - Important
 - Can simplify your algorithms
 - Can affect the efficiency of your code
 - Should be carefully designed
 - Algorithms and flow diagrams
 - The high-level template of your code
 - Draw a flow diagram for each task
 - Verifying the control flow
 - Try to run some requests to see whether the control flow design can lead to the expected operations.

Example (cont.)

- Software design
 - Set up the code template
 - An example is given in the next slide

; description ...

;include files

.include "m2560def.inc"

;define constants and variables here

; .equ directives

; .def directives

...

; set up interrupt vector table

jmp RESET

...

; reset and other interrupt subroutines

RESET:

; main

; perform basic work including testing each function

;end_of_main

; normal functions


; func1

...

Example (cont.)

- Software design
 - Write code – complete each section
 - Start from the definitions
 - Develop and test each function individually
 - Not necessarily implemented in “called functions”
 - Can be in other formats such as macros
 - The “display function” may be the first function to be completed since it may be used for testing other functions

Example (cont.)

- Testing
 - Does it really work?
 - Compile and run your code on the lab board
 - Check for different operations
 - If it is not working ...
 - Check the wiring
 - Debug the code
 - Analyze the bugs
 - Work out the solutions
 - Until it is working
 - 

Some Techniques in Software Code Development

- Develop and individually test each of the functions in the flowchart
- Use a simple `main()` function to exercise each of the the individual input/output device functions
- When the individual functions are working correctly, write the overall software in a stepwise fashion – adding one additional function to the code and debugging it before adding more – until the entire project is functionally correctly using the real or simulated inputs.
- It is often useful to develop the code to run output devices first so that you can use output devices to display results.