Embedded Operating Systems

Che-Wei Chang

chewei@mail.cgu.edu.tw

Department of Computer Science and Information Engineering, Chang Gung University

An Real-Time OS: µC/OS-II Quick Overview

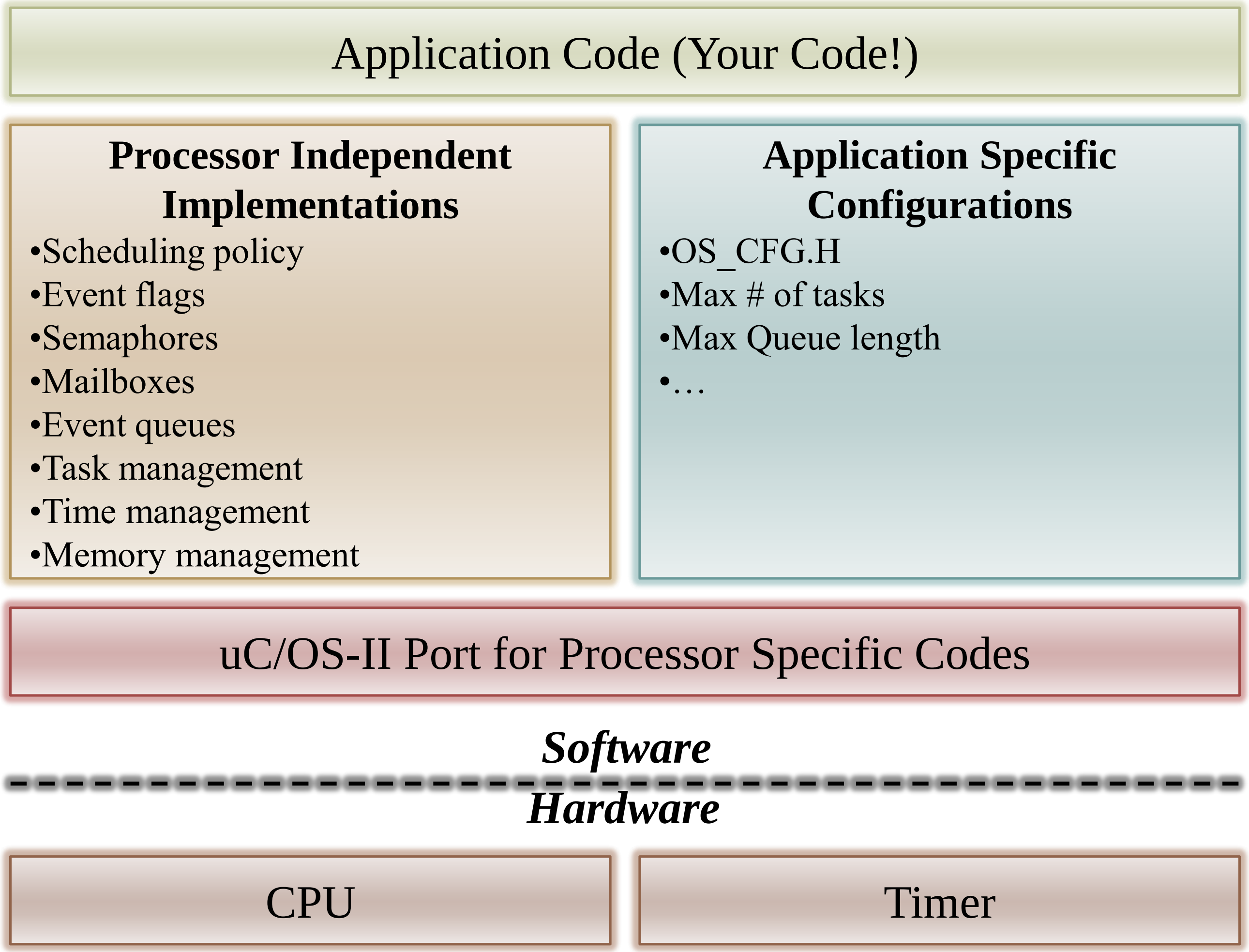
Introduction of µC/OS-II (1/2)

- The name is from micro-controller operating system, version 2
- µC/OS-II is certified in an avionics product by FAA in July 2000 and is also used in the Mars Curiosity Rover
- It is a very small real-time kernel
  - Memory footprint is about 20KB for a fully functional kernel
  - Source code is about 5,500 lines, mostly in ANSI C
  - It’s source is open but not free for commercial usages
- Preemptible priority-driven real-time scheduling
  - 64 priority levels (max 64 tasks)
  - 8 reserved for µC/OS-II
  - Each task is an infinite loop

Introduction of μC/OS-II (2/2)

- Deterministic execution times for most μC/OS-II functions and services
- Nested interrupts could go up to 256 levels
- Supports of various 8-bit to 64-bit platforms: x86, ARM, MIPS, 8051, etc.
- Easy for development: Borland C++ compiler and DOS (optional)
- However, uC/OS-II still lacks of the following features:
  - Resource synchronization protocol
  - Soft-real-time support

The μC/OS-II File Structure



Application Code (Your Code!)

Processor Independent Implementations

- Scheduling policy
- Event flags
- Semaphores
- Mailboxes
- Event queues
- Task management
- Time management
- Memory management

Application Specific Configurations

- OS_CFG.H
- Max # of tasks
- Max Queue length
- …

uC/OS-II Port for Processor Specific Codes

Software

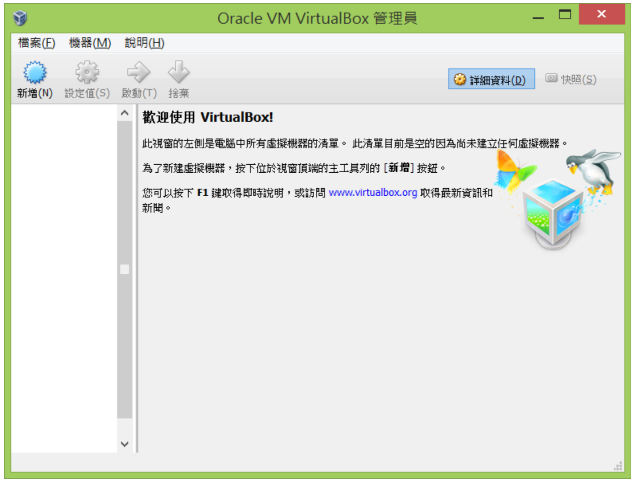
Hardware

CPU

Timer

Requirements of μC/OS-II Emulator

- Operating System
  - Windows XP 32bits
  - Use virtual machine to install the OS
  - Install “Guest Additions” for Virtualbox
- Tools
  - Borland C++ compiler (V4.5)
    - BC45 is the compiler
  - Turbo Assembler
    - The assembler is in tasm
  - The source code and the emulation environment of μC/OS-II
    - SOFTWARE is the package



歡迎使用 VirtualBox!

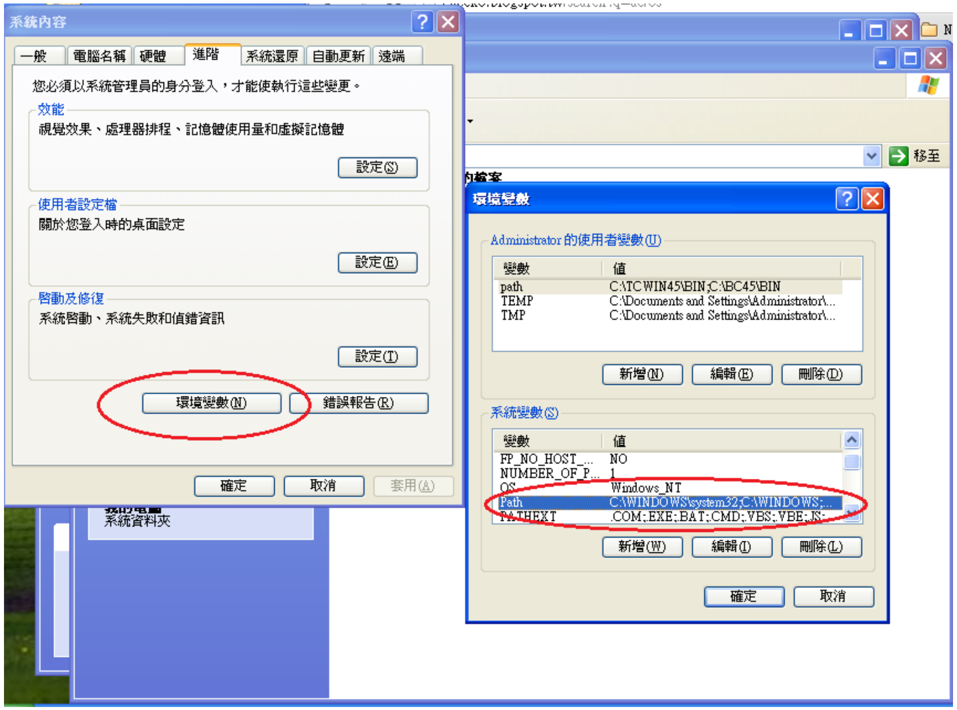
此視窗的左側是電腦中所有虛擬機器的清單。 此清單目前是空的因為尚未建立任何虛擬機器。

為了新建虛擬機器，按下位於視窗頂端的主工具列的 [**新增**] 按鈕。

您可以按下 **F1** 鍵取得即時說明，或訪問 www.virtualbox.org 取得最新資訊和新聞。

Borland C++ Compiler

- Download Borland C++ and install it on your windows XP environment
  - Double click the “INSTALL.EXE”
- Add “;C:\BC45\BIN” to your system Path



系統內容

您必須以系統管理員的身分登入，才能使執行這些變更。

效能

視覺效果、處理器排程、記憶體使用量和虛擬記憶體

使用者設定檔

關於您登入時的桌面設定

啟動及修復

系統啟動、系統失敗和偵錯資訊

環境變數

Administrator 的使用者變數(U)

變數	值
path	C:\TCWIN45\BIN;C:\BC45\BIN
TEMP	C:\Documents and Settings\Administrator\...
TMP	C:\Documents and Settings\Administrator\...

系統變數(S)

變數	值
FP_NO_HOST_...	NO
NUMBER_OF_P...	1
OS	Windows_NT
Path	C:\WINDOWS\system32;C:\WINDOWS;...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;...

Turbo Assembler

- Download Turbo assembler and unzip the file
- Copy “\tasm\BIN\TASM.EXE” to your “C:\BC45\BIN”
  - Include the missing assembler which is going to be used during we compile the source code of µC/OS-II

Compile μC/OS-II Example Code

- Download the source code and emulator μC/OS-II
  - It is recommended to put the source code package “SOFTWARE” directly in C:\
- Test the first example
  - Execute C:\SOFTWARE\uCOS-II\EX1_x86L\BC45\TEST\TEST.EXE
  - Press ECS to leave
- Rename or remove the executable file
  - Rename TEST.EXE
- Compile the μC/OS-II and the source code of the first example
  - Run C:\SOFTWARE\uCOS-II\EX1_x86L\BC45\TEST\MAKETEST.BAT
  - A new “TEST.EXE” will be created if we compile it successfully

Common Mistakes

- Did you directly put the package “SOFTWARE” in C:\ ?
- Have you copied the correct file “TASM.EXE” to your “C:\BC45\BIN” directory?
- Did you set the Path correctly?
  - See the picture in Page 7
  - There is no space

Project Requirements

CPU Scheduler

- Short-term scheduler selects a process among the processes in the ready queue, and allocates the CPU to the selected process
  - Queue may be ordered in various ways
- CPU scheduling decisions may take place when a process:
  1. Switches from running to waiting state
  2. Switches from running to ready state
  3. Switches from waiting to ready
  4. Terminates
- Scheduling under 1 and 4 is nonpreemptive
- All other scheduling is preemptive

Dispatcher

- Dispatcher module gives control of the CPU to the process selected by the short-term scheduler
  - switching context
  - switching to user mode
  - jumping to the proper location in the user program to resume that process
- Dispatch latency – the time it takes for the dispatcher to stop one process and start another running

Scheduling Algorithms

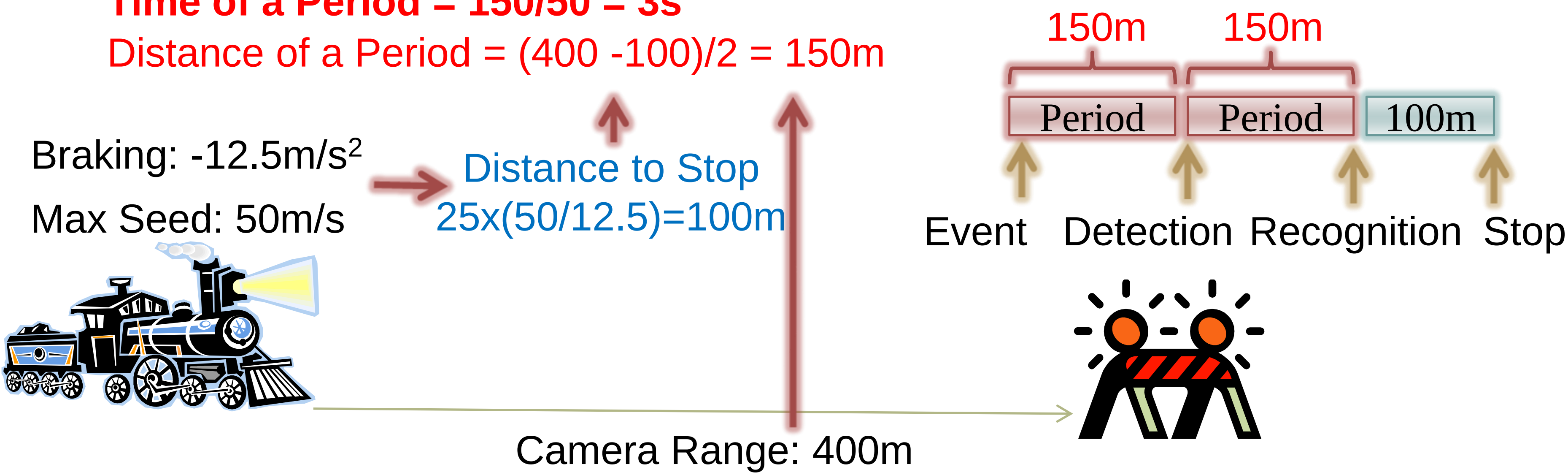
- First-Come, First-Served Scheduling (FIFO)
- Shortest-Job-First Scheduling (SJF)
- Priority Scheduling
- Round-Robin Scheduling (RR)
- Multilevel Queue Scheduling
- Multilevel Feedback Queue Scheduling
- Multiple-Processor Scheduling

An Example of Real-Time Tasks

- A camera periodically takes a photo
- The image recognition result will be produced before the next period
- If there is an obstacle, the train automatically brakes

Time of a Period = 150/50 = 3s

Distance of a Period = (400 -100)/2 = 150m



Braking: -12.5m/s^2

Max Seed: 50m/s

Distance to Stop

25x(50/12.5)=100m

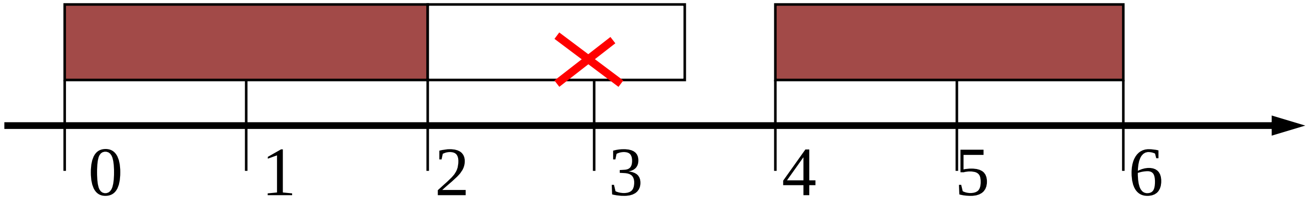
Camera Range: 400m

Periodic Task Scheduling

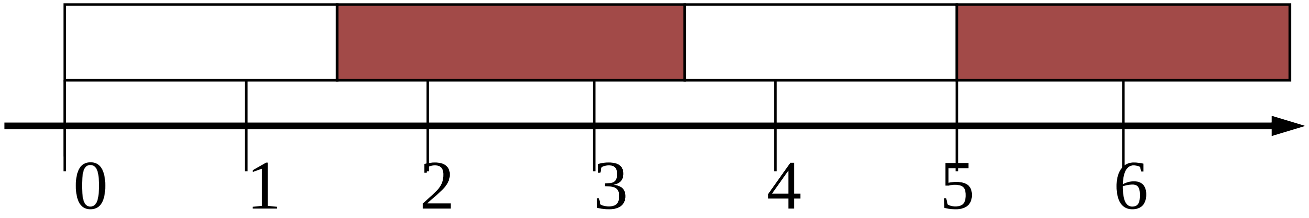
- Studying: 2 days per 4 days 🟥

  Playing Basketball: 1.5 days per 3 days □

- Case 1: Studying is always more important



- Case 2: Doing whatever is more urgent



Project Requirements

- Read the input file and create the periodic tasks
- Implement the rate monotonic (RM) scheduler
- Call the functions for using semaphores
- Bonus 1 (10%): Implement the priority inheritance protocol (PIP)
- Bonus 1 (10%): Implement the priority ceiling protocol (PCP)

Input File Format

- The total required CPU time of all tasks are no more than 80%
- Input file format: (all are integers)

$$
\begin{array}{l}
N \\
S_1\ T_1\ E_1\ I_{1,1}\ L_{1,1}\ R_{1,1}\ \ldots I_{1,\,s1}\ L_{1,s1}\ R_{1,s1} \\
\qquad\qquad \vdots \\
S_N\ T_N\ E_N\ I_{N,1}\ L_{N,1}\ R_{N,1}\ \ldots I_{N,sN}\ L_{N,sN}\ R_{N,sN}
\end{array}
$$

- N: Number of periodic tasks
- S_x: Number of semaphores used by task i
- T_x: Period of task i
- E_x: Execution time of task i
- $I_{x,y}$: The index number of the y-th semaphore of x-th task
- $L_{x,y}$: The request time of the y-th semaphore of x-th task
- $R_{x,y}$: The using time of the y-th semaphore of x-th task

Example of Input

```
3
2 20 5 1 0 1 2 1 2
3 25 6 2 1 2 3 3 1 1 4 2
2 30 6 1 0 2 2 3 2
```

Needed Files

- Source code of your project (SOFTWARE)
- Report: 4 pages
- Deadline: 20:00 on 2024/12/16
- Upload to the e-learning system
- The grading baseline: 90

File Formats

- File name: EOS-Project-StudentID-Report
- File type: PDF
- File name: EOS-Project-StudentID-Source
- File type: ZIP