Operating System Concepts

Che-Wei Chang

chewei@mail.cgu.edu.tw

Department of Computer Science and Information Engineering, Chang Gung University

Homework 4-
An Real-Time OS: μC/OS-II
Quick Overview

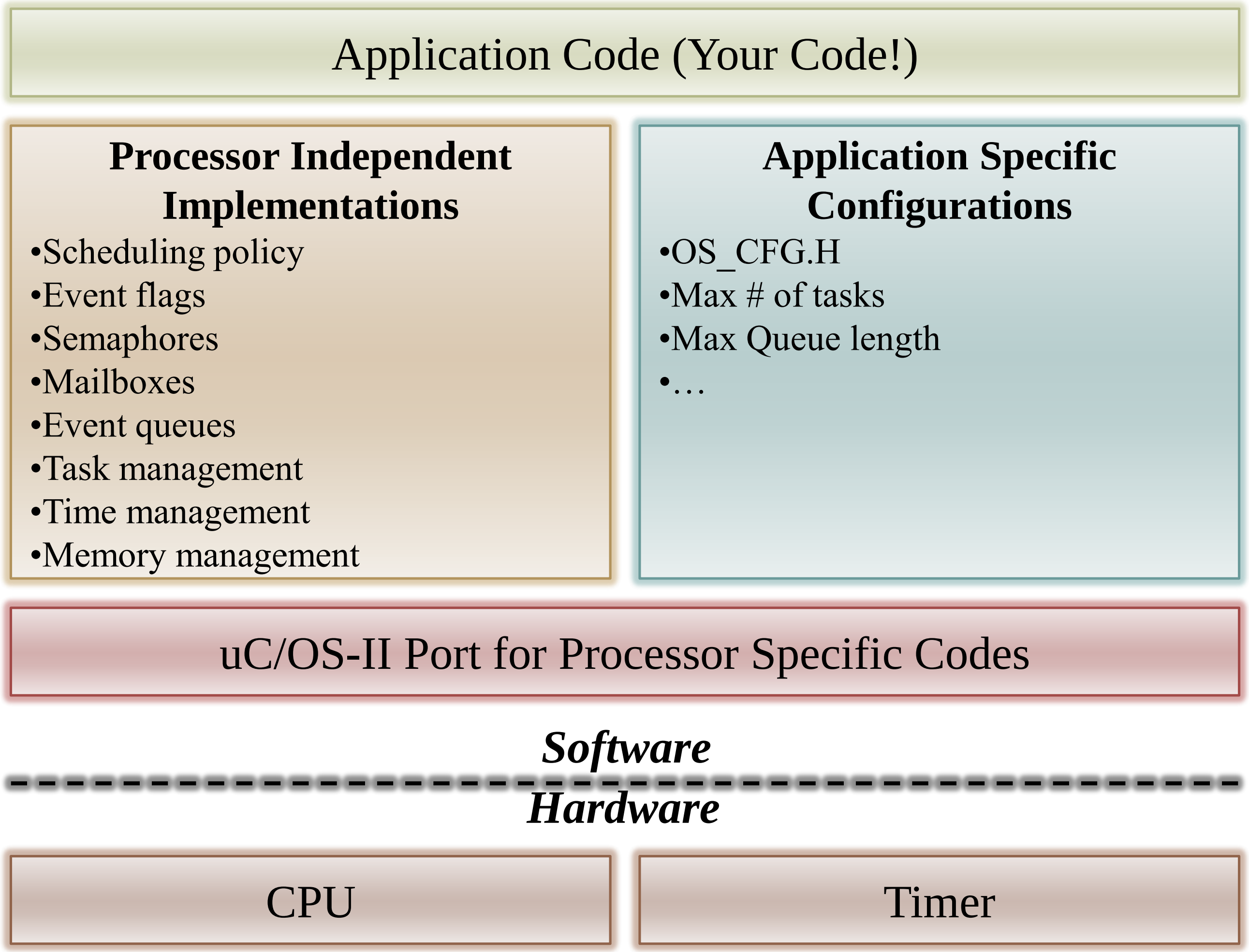
Introduction of µC/OS-II (1/2)

- 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



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
- Full Package
  - Download it from the course website with password: csie2020
  - https://www.csie.cgu.edu.tw/~chewei/files/ucOSII_ProjectPackage.zip
  - https://www.csie.cgu.edu.tw/~chewei/files/Files.zip

歡迎使用 VirtualBox!

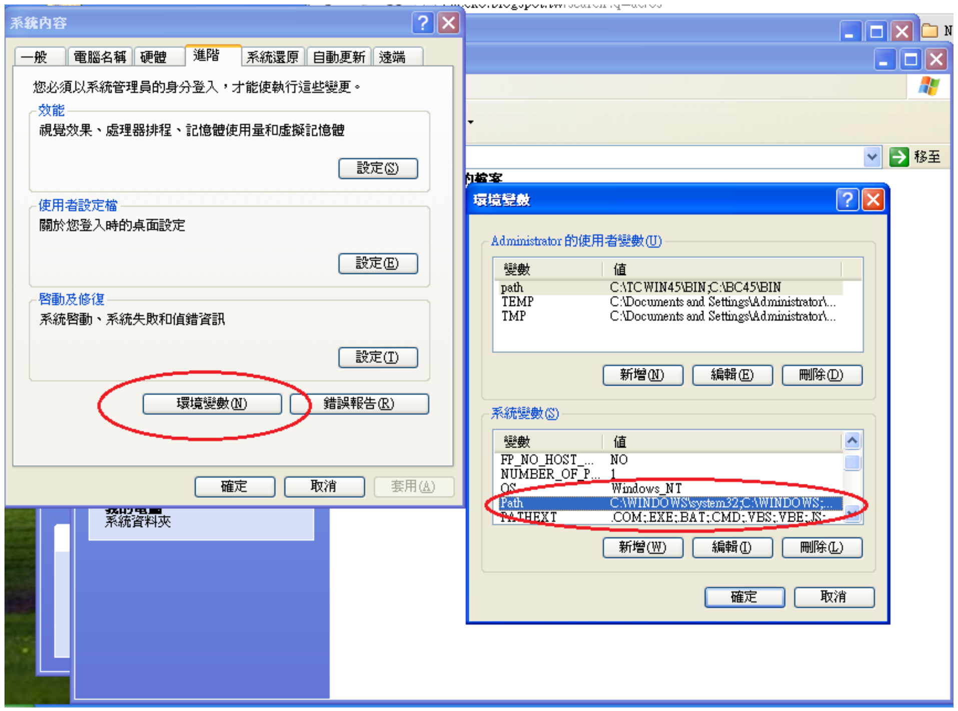
此視窗的左側是電腦中所有虛擬機器的清單。 此清單目前是空的因為尚未建立任何虛擬機器。

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



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

啓動及修復
系統啓動、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:\TC WIN45\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

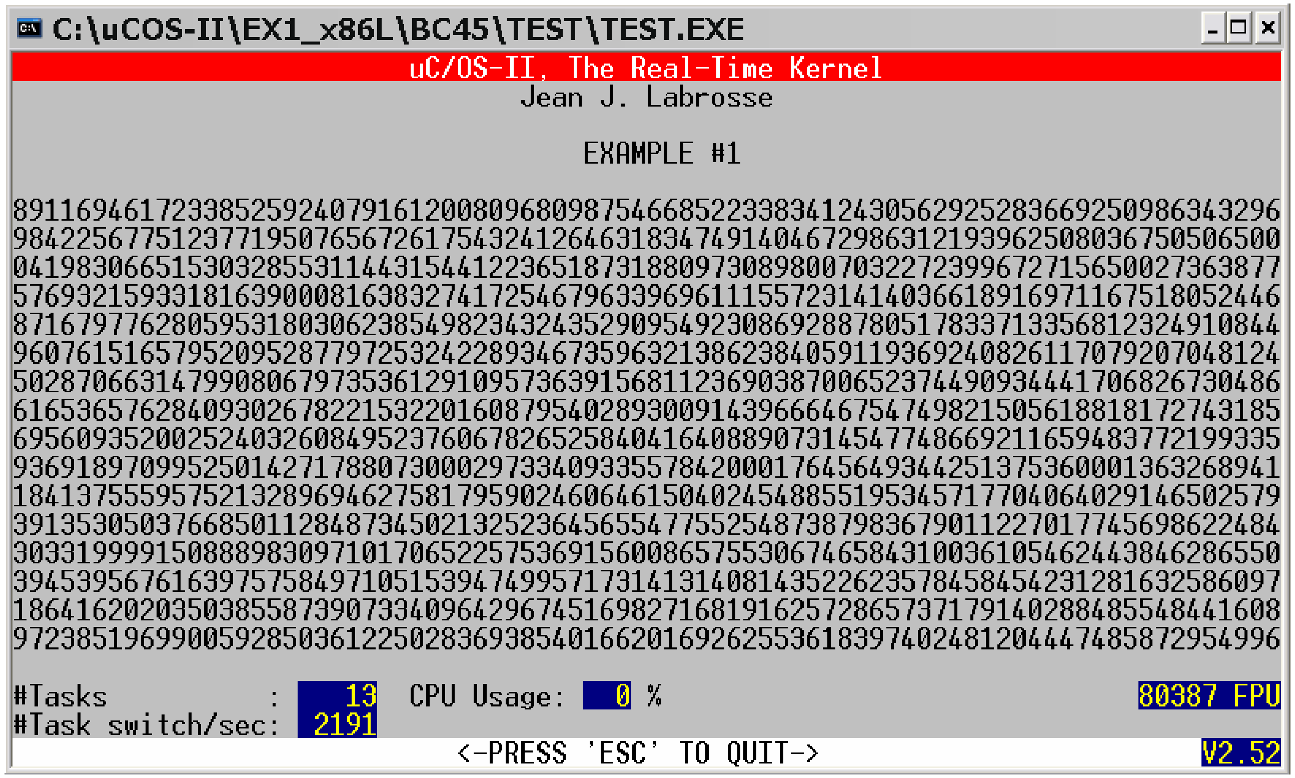
Compile µC/OS-II Example Code

- 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

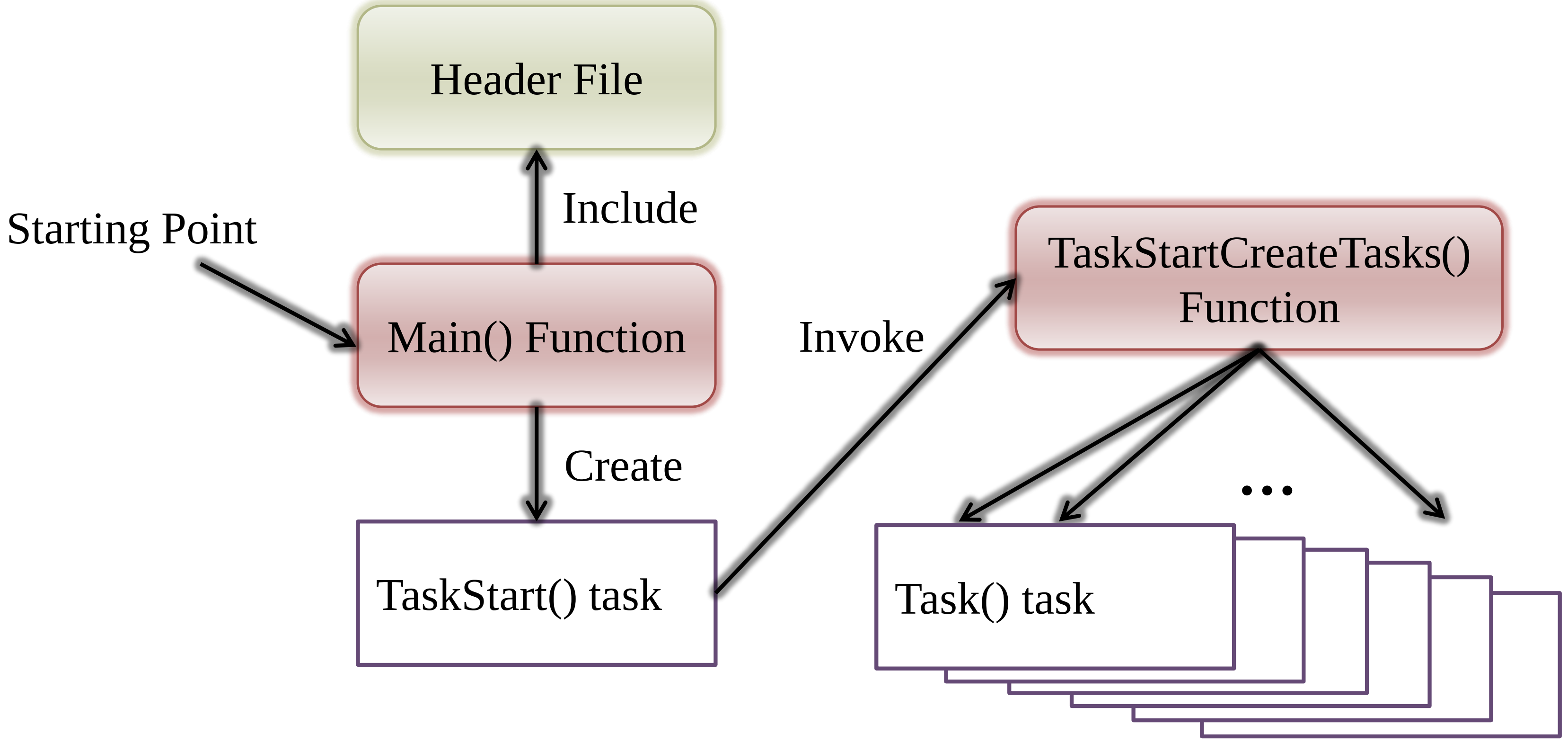
An Example on μC/OS-II: Multitasking



```
C:\uCOS-II\EX1_x86L\BC45\TEST\TEST.EXE
                          uC/OS-II, The Real-Time Kernel
                                Jean J. Labrosse

                                   EXAMPLE #1

89116946172338525924079161200809680987546685223383412430562925283669250986343296
98422567751237719507656726175432412646318347491404672986312193962508036750506500
04198306651530328553114431544122365187318809730898007032272399672715650027363877
57693215933181639000816383274172546796339696111557231414036618916971167518052446
87167977628059531803062385498234324352909549230869288780517833713356812324910844
96076151657952095287797253242289346735963213862384059119369240826117079207048124
50287066314799080679735361291095736391568112369038700652374490934441706826730486
61653657628409302678221532201608795402893009143966646754749821505618818172743185
69560935200252403260849523760678265258404164088907314547748669211659483772199335
93691897099525014271788073000297334093355784200017645649344251375360001363268941
18413755595752132896946275817959024606461504024548855195345717704064029146502579
39135305037668501128487345021325236456554775525487387983679011227017745698622484
30331999915088898309710170652257536915600865755306746584310036105462443846286550
39453956761639757584971051539474995717314131408143522623578458454231281632586097
18641620203503855873907334096429674516982716819162572865737179140288485548441608
97238519699005928503612250283693854016620169262553618397402481204447485872954996

#Tasks          :    13  CPU Usage:    0 %                              80387 FPU
#Task switch/sec:  2191
                       <-PRESS 'ESC' TO QUIT->                                V2.52
```

- Three system tasks
- Ten application tasks randomly prints its number

Multitasking: Workflow



Multitasking: TEST.C

(\SOFTWARE\uCOS-II\EX1_x86L\BC45\SOURCE\TEST.C)

```
#include "includes.h"
/*
************************************************************
CONSTANTS
************************************************************
*/
#define TASK_STK_SIZE 512
#define N_TASKS 10
/*
************************************************************
VARIABLES
************************************************************
*/
OS_STK TaskStk[N_TASKS][TASK_STK_SIZE];
OS_STK TaskStartStk[TASK_STK_SIZE];
char TaskData[N_TASKS];
OS_EVENT *RandomSem;
```

實 樸 勞 勤

Multitasking: Main()

```
void main (void)
{
        PC_DispClrScr(DISP_FGND_WHITE + ISP_BGND_BLACK);
        OSInit();
        PC_DOSSaveReturn();
        PC_VectSet(uCOS, OSCtxSw);
        RandomSem = OSSemCreate(1);
        OSTaskCreate( TaskStart,
                      (void *)0,
                      (void *)&TaskStartStk[TASK_STK_SIZE-1],
                      0);
        OSStart();
}
```

Entry point of the task (a pointer to a function)

User-specified data

Top of stack

Priority (0=hightest)

Multitasking: TaskStart()

```
void TaskStart (void *pdata)
{
        /*skip the details of setting*/
        OSStatInit();
        TaskStartCreateTasks();
        for (;;)
        {
                if (PC_GetKey(&key) == TRUE)
                {
                        if (key == 0x1B) { PC_DOSReturn(); }
                }
                OSTimeDlyHMSM(0, 0, 1, 0);
        }
}
```

Call the function to create the other tasks

See if the ESCAPE key has been pressed

Wait one second

Multitasking: TaskStartCreateTasks()

```
static void TaskStartCreateTasks (void)
{
        INT8U i;
        for (i = 0; i < N_TASKS; i++)
        {
                TaskData[i] = '0' + i;
                OSTaskCreate(
                        Task,
                        (void *)&TaskData[i],
                        &TaskStk[i][TASK_STK_SIZE - 1],
                        i + 1 );
        }
}
```

Entry point of the task (a pointer to function)

Argument: character to print

Top of stack

Priority

Multitasking: Task()

```
void Task (void *pdata)
{
        INT8U x;
        INT8U y;
        INT8U err;
        for (;;)
        {
                OSSemPend(RandomSem, 0, &err);
                /* Acquire semaphore to perform random numbers */
                x = random(80);
                /* Find X position where task number will appear */
                y = random(16);
                /* Find Y position where task number will appear */
                OSSemPost(RandomSem);
                /* Release semaphore */
                PC_DispChar(x, y + 5, *(char *)pdata, DISP_FGND_BLACK +DISP_BGND_LIGHT_GRAY);
                /* Display the task number on the screen */
                OSTimeDly(1);
                /* Delay 1 clock tick */
        }
}
```

Randomly pick up the position to print its data

Print & delay

OSinit()

(\SOFTWARE\uCOS-II\SOURCE\OS_CORE.C)

- Initialize the internal structures of μC/OS-II and MUST be called before any services
- Internal structures of μC/OS-2
  - Task ready list
  - Priority table
  - Task control blocks (TCB)
  - Free pool
- Create housekeeping tasks
  - The idle task
  - The statistics task

PC_DOSSaveReturn()
(\SOFTWARE\BLOCKS\PC\BC45\PC.C)

- Save the current status of DOS for the future restoration
  - Interrupt vectors and the RTC tick rate
- Set a global returning point by calling setjump()
  - μC/OS-II can come back here when it terminates.
  - PC_DOSReturn()

PC_VectSet(uCOS,OSCtxSw)
(\SOFTWARE\BLOCKS\PC\BC45\PC.C)

- Install the context switch handler
- Interrupt 0x08 (timer) under 80x86 family
  - Invoked by INT instruction

OSStart()

(SOFTWARE\uCOS-II\EX1_x86L\BC45\SOURCE\CORE.C)

- Start multitasking of µC/OS-II
- It never returns to main()
- µC/OS-II is terminated if PC_DOSReturn() is called

Report

1. The steps for your implementation
2. The problem you met, and how you solved it
3. **The reference of this homework**

- The report is limited within 4 pages in PDF

Extra Exercise

- Read the e-book of μC/OS-II
  - Try to read and understand the first chapter
- Read the source code to understand the application
  - The application source code is in C:\SOFTWARE\uCOS-II\EX1_x86L\BC45\SOURCE
- Browse the source code of μC/OS-II
  - The source code of μC/OS-II is in C:\SOFTWARE\uCOS-II\SOURCE
- 準時繳交且實作完成第九頁的內容，提供截圖或相關說明 ➔ 標準分數為80正負10分
- 有做Extra Exercise，並寫入報告心得且說明精確者最多加20分

Grading

- Implementation
  - Install the environment for running μC/OS-II 30%
  - Compile and run the first example 30%
- Report
  - 20%
- Bonus
  - Extra exercise 20%
- Demo Q&A
  - 20%

Submission

- Homework 4 deadline: at 23:00 on 2024-12-2

➔**NO DELAY!**

- Upload to e-learning system
- The title of the report: OSHomework4StudentID
- **Point deduction for wrong format: 10%**

➔DEMO will be arranged!