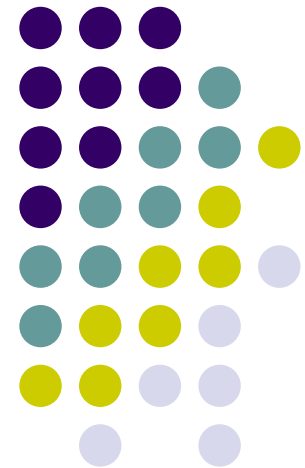


CH1

Getting Started with μ C/OS-II

Date : 2007/07/18

Speaker: Ming-Shyong Tsai





Example 1

```
C:\C:\AEX1_x86\ABC45\TEST\TEST.EXE
uC/OS-II, The Real-Time Kernel
Jean J. Labrosse

EXAMPLE #1

03      4 27      912 2 6054 0494 09 23      10 77712 546 1 27 0 3 0 799 2 95
16      62      80 79 6892 5 0      17      1 0 23 4 5 2 5 8 40      8 9 7 1 96
8 44 4      1 1 3 0022      202 2      67 7313 4 5 210 80 864      9 5 4 2
6      8 5 2 8390 4279999 2 9 138 9 67 57 6      0553837 6 8 1 1 18 6 05
2 9 15      61544      9      9 340 44      96339 4 2 8 9 74 1 491 8 7 7
      315 3 0 144934 3      27 2      7 8 48 0 3 22 78      0 85 75
5 015 1 58 4 2 51083      57080      70 50 28 6 8 9 78 009 9 9 9 48 7
66 2      6      05 6      9099 5957      06      3 13      314 33 6 15 2
3      115 84 6      60      1886 23      85 0 3 37833      7 0 3 4 9 1 8 77
73 6      8 4 8      1 1 3      81 7 0      29 9      345 3      64 1 22
7 6      8 1 6 7 52967 9 0 8 3      5 664 4 1 8 5 9 4 8 382 2
1 1 7 7 57      1 2 1 55      8 8      1 49 5 3 911      4 8 040 9 82
3      6 969 2      5 235 1 6 27      5 5      262178 5 8 9 24 2 2 355 8
      1 2850 8 20 3 414      1 3 58 7 1 0 7      30 283 0 2 4 6 26 5
6 114 6 6 357 1 9 60      3 7 7      999 369 9 81 0 7 6 26483
05 8 4 0 8      3 35 4 6 75543 64 6      743 8 7 45 8 670 6 69 76 4 3

#Tasks      : 13 CPU Usage: 0 % 80387 FPU
#Task switch/sec: 5

<-PRESS 'ESC' TO QUIT-> 02.52
```

- Example 1 show multitasking ability of uC/OS-II



Example 1

```
#include "includes.h"
```

```
#define TASK_STK_SIZE 512
```

```
#define N_TASKS 10
```

```
OS_STK TaskStk[N_TASKS][TASK_STK_SIZE];
```

```
OS_STK TaskStartStk[TASK_STK_SIZE];
```

```
char TaskData[N_TASKS];
```

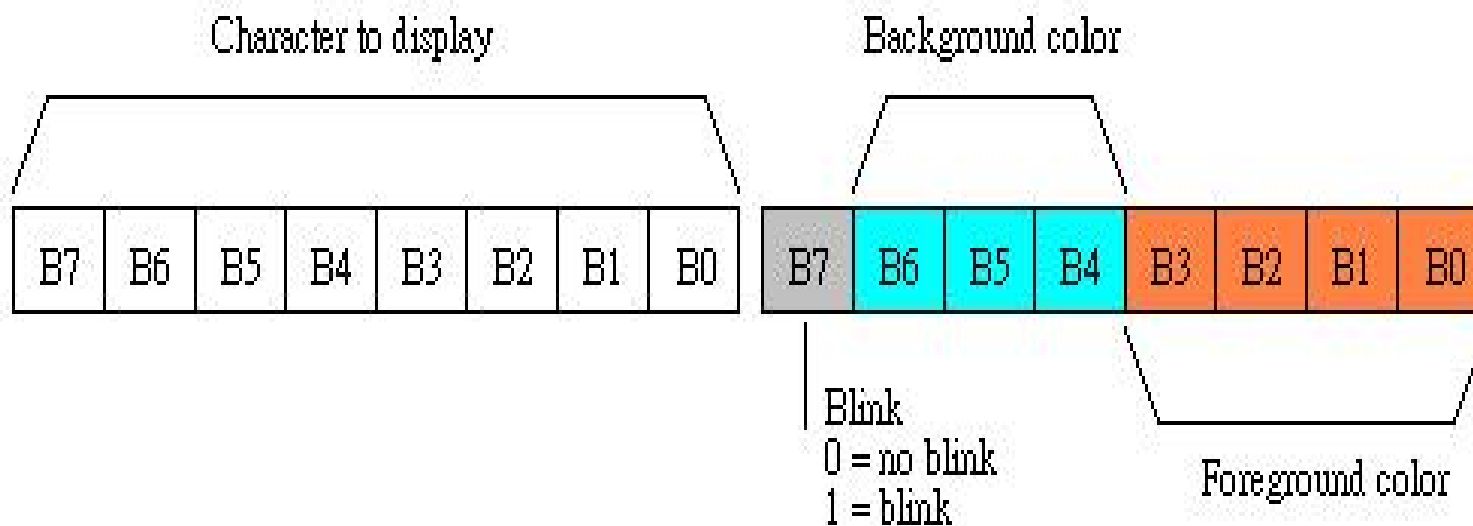
```
OS_EVENT *RandomSem;
```



Example 1 – main ()

```
void main(void)  
{  
  PC_DispClrScr(DISP_FGND_WHITE + DISP_BGND_BLACK);
```

OSInit(). Create 2 tasks : idle task -> execute when there is no ready task to run





Example 1

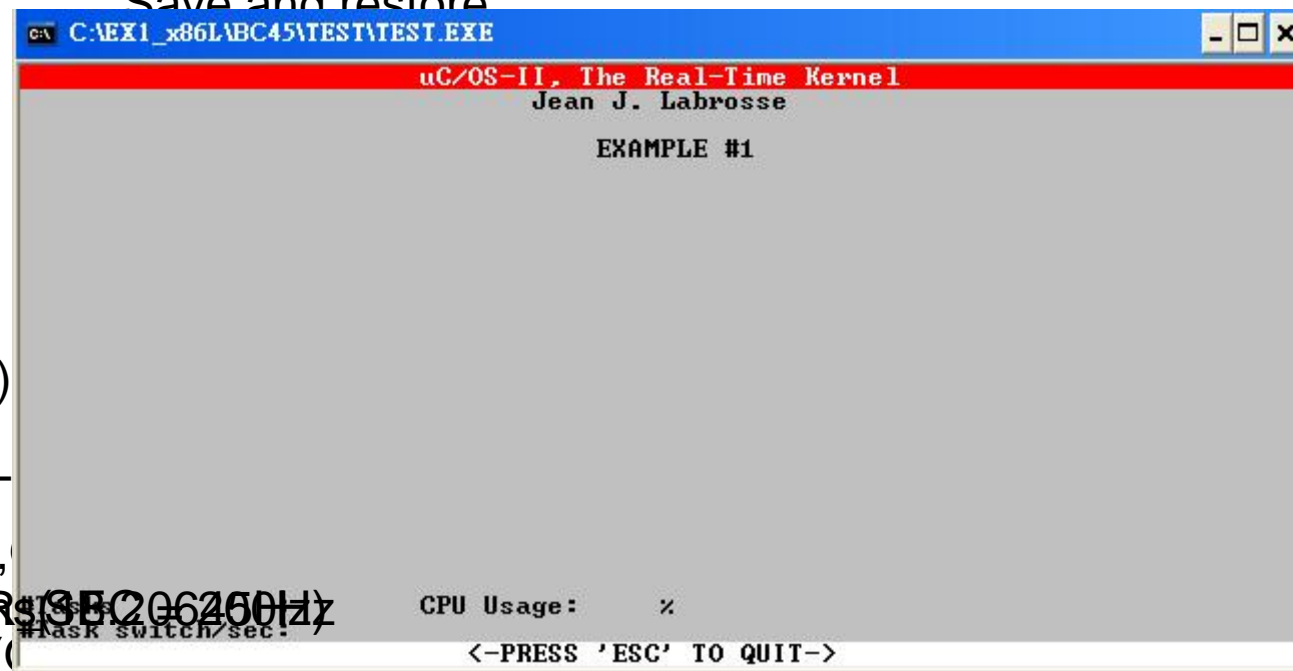
TaskStart(void *pdata)

```
void TaskStart(void *pdata)
{
    OSStatInit(); //be called to determine CPU speed
    #if OS_CRITICAL_METHOD == 3
        OS_CPU_SR cpu_sr; // Save and restore
    #endif
    char s[100];
    INT16S key;

    pdata = pdata;

    TaskStartDispInit()

    OS_ENTER_CRIT
    PC_VectSet(0x08,
    OS_TICKS_PER_SEC(3000000) // 206250Hz
    PC_SetTickRate(
    _SEC);
    OS_EXIT_CRITACL();
}
```



Example 1 - static void TaskStartCreateTask(void)



```
static void TaskStartCreateTask(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);
    }
}
```



Example 1 - TaskStart(void *pdata)

```

C:\EX1_x86\ABC45\TEST\TEST.EXE
uC/OS-II, The Real-Time Kernel
Jean J. Labrosse

EXAMPLE #1

03      4 27      912 2 6054 0494 09 23      10 77712 546 1 27 0 3 0 799 2 95
16      62      80 79 6892 5 0      17      1 0 23 4 5 2 5 8 40      8 9 7 1 96
8 44 4      1 1 3 0022      202 2      67 7313 4 5 210 00 864      9 5 4 2
6      8 5 2 0390 4279999 2 9 138 9 67 57 6      0553837 6 8 1 1 18 6 05
2 9 15      61544      9      9 340 44      96339 4 2 0 9 74 1 491 0 7 7
      315 3 0 144934 3 27 2      7 8 48 0 3 22 70      0 85 75
5 015 1 58 4 2 51003      57000      70 50 28 6 8      9 78 009 9 9      9 48 7
66 2      6      05 6      9099 5957      06      3 13      314 33 6 15 2
3      115 84 6      60      1886 23      85 0 3 37833      7 0 3 4 9 1 8 77
73 6 8 4 8      1 1 3      81 7 0      29 9      345 3      64 1 22
7 6      8 1 6 7 52967 9 0 8 3      5 664 4 1 8 5 9 4 8 382 2
1 1 7 7 57 1 2 1 55      0 8      1 49 5 3 911      4 8 040 9 82
3      6 969 2      5 235 1 6 27      5 5      262170 5 8 9 24 2 2 355 0
      1 2050 8 20 3 414      1 3 58 7 1 0      7      30 203 0 2 4 6 26 5
6 114 6 6 357 1 9 60      3 7 7      999 369 9 81 0 7 6 26483
05 8 4 0 8      3 35 4 6 75543 64 6      743 8 7 45 8 670 6 69 76 4 3

#Tasks      : 13      CPU Usage: 0 %      80387 FPU
#Task switch/sec: 5
<-PRESS 'ESC' TO QUIT->      02.52

```

```

OSTimeDlyHMSM(0, 0, 1, 0);
}
}

```

Example 1 -void Task(void *pdata)



```
void Task(void *pdata)
{
    INT8U x;
    INT8U y;
    INT8U err;

    for(;;){
        OSSemPend(RandomSem, 0, &err);
        x = random(80);
        y = random(16);
        OSSemPost(RandomSem);

        PC_DispChar(x , y+5, *(char*)pdata, DISP_FGND_LIGHT_GRAY);
        OSTimeDly(1);
    }
}
```




Example 2

```
C:\EX2_x86\ABC45\TEST\TEST.EXE
uC/OS-II, The Real-Time Kernel
Jean J. Labrosse
EXAMPLE #2

Task          Total Stack  Free Stack  Used Stack  ExecTime (uS)
-----
TaskStart():      624         178         446          2
TaskClk()  :     1024         700         324          2
Task1()    :     1024         662         362          3
Task2()    :     1024         970          54          4
Task3()    :     1024         468         556          2
Task4()    :     1024         956          68          3
Task5()    :     1024         944          80          4

#Tasks      : 9      CPU Usage: 0%
#Task switch/sec: 67

80387 FPU
2007-07-17 23:02:10
V2.52
<-PRESS 'ESC' TO QUIT->
```

- Example 2 show stack-checking feature of uC/OS-II

Example 2



```
#include "includes.h"

#define TASK_STK_SIZE 512
#define TASK_START_ID 0
#define TASK_CLK_ID 1
#define TASK_1_ID 2
#define TASK_2_ID 3
#define TASK_3_ID 4
#define TASK_4_ID 5
#define TASK_5_ID 6

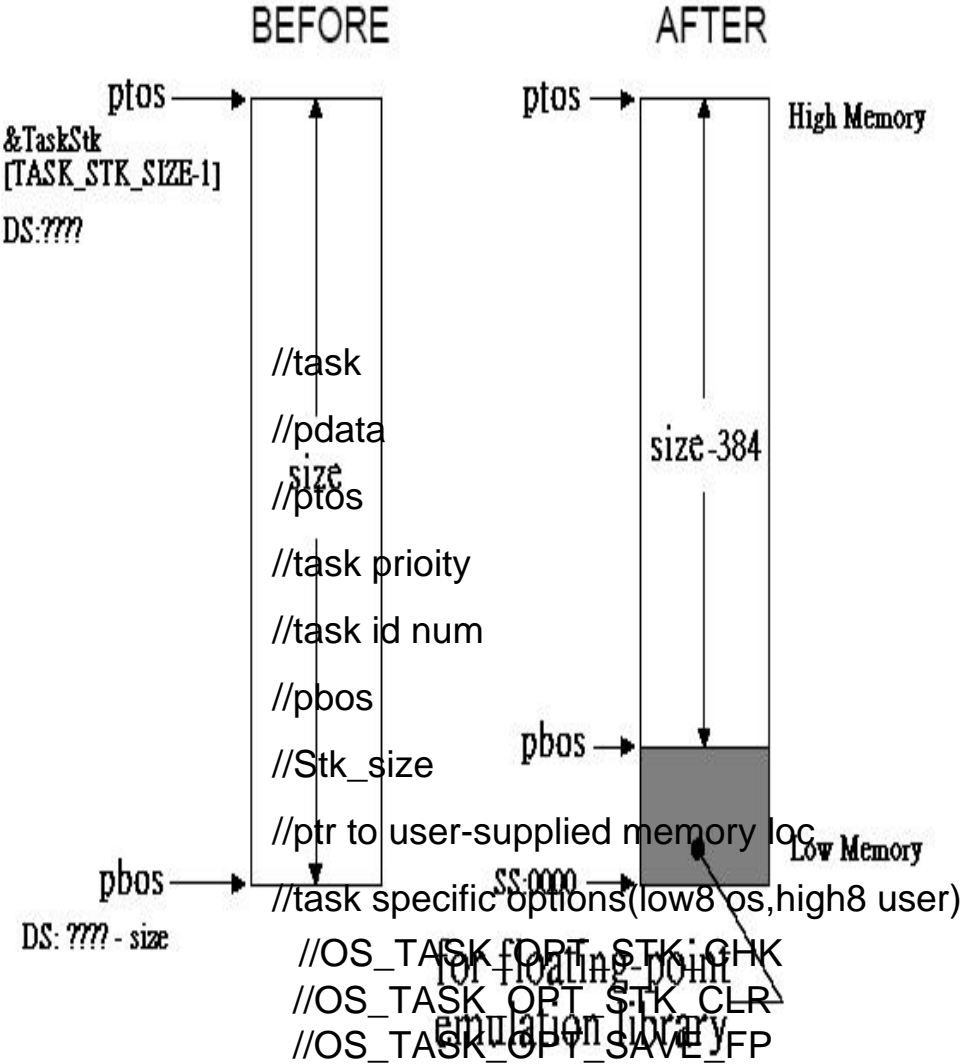
#define TASK_START_PRIO 10
#define TASK_CLK_PRIO 11
#define TASK_1_PRIO 12
#define TASK_2_PRIO 13
#define TASK_3_PRIO 14
#define TASK_4_PRIO 15
#define TASK_5_PRIO 16

OS_STK TaskStartStk[TASK_STK_SIZE];
OS_STK TaskClkStk[TASK_STK_SIZE];
OS_STK Task1Stk[TASK_STK_SIZE];
OS_STK Task2Stk[TASK_STK_SIZE];
OS_STK Task3Stk[TASK_STK_SIZE];
OS_STK Task4Stk[TASK_STK_SIZE];
OS_STK Task5Stk[TASK_STK_SIZE];

OS_EVENT *AckMbox;
OS_EVENT *TxMbox;
```



Example 2 – main()



```

ptos =
  &TaskStartStk[TASK_STK_SIZE - 1];
pbos = &TaskStartStk[0];
size = TASK_STK_SIZE;
OSTaskStkInit_FPE_x86(
  &ptos,& pbos,&size);

```

```

OSTaskCreateExt(TaskStart,
  (void *)0,
  ptos,
  TASK_START_PRIO,
  TASK_START_ID,
  pbos,
  size,
  (void *)0,
  OS_TASK_OPT_STK_CHK
  | OS_TASK_OPT_STK_CLR);
OSStart();

```

}

Example 2 – void TaskStart(void *pdata)



```
void TaskStart(void *pdata)
{
#if OS_CRITICAL_METHOD == OS_CPU_SR
    OS_CPU_SR cpu_sr;
#endif
    INT16S key;

    pdata = pdata;

    TaskStartDisplnit();

    OS_ENTER_CRITICAL();
    PC_VectSet(0x08, OS_VectTbl[PC_Vect]);
    PC_SetTickRate(OS_TickRate);
    OS_EXIT_CRITICAL();

    OSStatInit();
}
```

```
C:\EX2_x86\ABC45\TEST\TEST.EXE
uC/OS-II, The Real-Time Kernel
Jean J. Labrosse

EXAMPLE #2

Task          Total Stack  Free Stack  Used Stack  ExecTime (uS)
-----
TaskStart():
TaskClk()    :
Task1()      :
Task2()      :
Task3()      :
Task4()      :
Task5()      :

#Tasks      :      CPU Usage:      %
#Task switch/sec:

<-PRESS 'ESC' TO QUIT->
```

Example 2 – void Task1 (void *pdata)



```
void Task1 (void *pdata)
{
    INT8U      err;
    OS_STK_DATA data;
    INT16U     time;
    INT8U      i;
    char       s[80];

    pdata = pdata;

    for(;;){
        for(i=0; i<7; i++) {
            PC_ElapsedStart();
            err =
OSTaskStkChk(TASK_START_PRIO
+i , &data);
            time = PC_ElapsedSStop();
```

```
        if(err == OS_NO_ERR) {
            sprintf(s, "%4ld  %4ld  %4ld  %6d",
                data.OSFree + Data.OSUsed,
                data.OSFree,
                data.OSUsed,
                time);
            PC_DispStr(19, 12 + i, s,
                DISP_FGND_YELLOW);
            //the information is retrieved by
            //OSTaskStkChk and formatted into a
            //string and displayed.
        }
    }
    OSTimeDlyHMSM(0,0,0,100);
}
}
```

Example 2 – void Task2 (void *data)



```
void Task2 (void *data)
{
    data = data;
    for(;;) {
        PC_Dispatch(70, 15, '|', DISP_FGND_WHITE + DISP_BGND_RED);
        OSTimeDly(10);
        PC_Dispatch(70, 15, '/', DISP_FGND_WHITE + DISP_BGND_RED);
        OSTimeDly(10);
        PC_Dispatch(70, 15, '-', DISP_FGND_WHITE + DISP_BGND_RED);
        OSTimeDly(10);
        PC_Dispatch(70, 15, '\\', DISP_FGND_WHITE + DISP_BGND_RED);
        OSTimeDly(10);
    }
}
```

Example 2 – void Task3 (void *data)



```
void Task3 (void *data)
{
    char dummy[500];
    INT16U i;

    data = data;
    for(i=0;i<499;i++) {
        dummy[i] = '?';
    }

    for(;;) {
        PC_Dispatch(70, 16, '|', DISP_FGND_WHITE + DISP_BGND_BLUE);
        OSTimeDly(20);
        PC_Dispatch(70, 16, '\\', DISP_FGND_WHITE + DISP_BGND_BLUE);
        OSTimeDly(20);
        PC_Dispatch(70, 16, '-', DISP_FGND_WHITE + DISP_BGND_BLUE);
        OSTimeDly(20);
        PC_Dispatch(70, 16, '/', DISP_FGND_WHITE + DISP_BGND_BLUE);
        OSTimeDly(20);
    }
}
```

Example 2 – void Task4(void *data)



```
void Task4(void *data)
{
    char txmsg;
    INT8U err;

    data = data;
    txmsg = 'A';

    for(;;) {
        OSMboxPost(TxMbox ,(void *)&txmsg); //send msg by TxMbox
        OSMboxPend(AckMbox, 0, &err); //receive ack on AckMbox
        txmsg++;
        if(txmsg == 'Z'){
            txmsg ='A';
        }
    }
}
```


Example 2 – void Task5(void *data)



```
void Task5(void *data)
{
    char rxmsg;
    INT8U err;

    for(;;) {
        rxmsg = (char*)OSMboxPend(TxMbox, 0, &err);
        PC_Dispatch(70, 18, *rxmsg, DISP_FGND_YELLOW
+DISP_BGND_RED);
        OSTimeDlyHMSM(0, 0, 1, 0);
        OSMboxPost(AckMbox ,(void *)1);
    }
}
```

Example 2 – void TaskClk(void *data)



```
void TaskClk(void *data)
{
    char s[40];

    data = data;
    for(;;){
        PC_GetDateTime(s);
        PC_DispStr(60, 23, s, DISP_FGND_BLUE + DISP_BGND_CTAN);
        OSTimeDly(OS_TICKS_PER_SEC);
    }
}
```



Example 3

```
C:\C:\EX3_x86\ABC45\TEST\TEST.EXE
uC/OS-II, The Real-Time Kernel
Jean J. Labrosse

EXAMPLE #3

Task Name          Counter  Exec.Time(uS)  Tot.Exec.Time(uS)  %Tot.
-----
StartTask          00015    18             293                21 %
Clock Task         00024    22             439                32 %
MsgQ Rx Task       00097     2             329                24 %
MsgQ Tx Task #2    00048     1              64                 4 %
MsgQ Tx Task #3    00024     2              33                 2 %
MsgQ Tx Task #4    00024     1              32                 2 %
TimeDlyTask        00120     1             176                12 %

#Tasks      : 9   CPU Usage: 4 %
#Task switch/sec: 41

80387 FPU
2007-07-18 03:19:51
U2.52

<-PRESS 'ESC' TO QUIT->
```

- Example 3 show extended functionality of uC/OS-II

Example 3



```
#include "includes.h"

#define TASK_STK_SIZE 512

#define TASK_START_ID 0
#define TASK_CLK_ID 1
#define TASK_1_ID 2
#define TASK_2_ID 3
#define TASK_3_ID 4
#define TASK_4_ID 5
#define TASK_5_ID 6

#define TASK_START_PRIO 10
#define TASK_CLK_PRIO 11
#define TASK_1_PRIO 12
#define TASK_2_PRIO 13
#define TASK_3_PRIO 14
#define TASK_4_PRIO 15
#define TASK_5_PRIO 16
#define MSG_QUEUE_SIZE 20
```

```
typedef struct{
    char TaskName[30];
    INT16U TaskCtr;
    INT16U TaskExecTime;
    INT32U TaskTotExecTime;
} TASK_USER_DATA;

OS_STK TaskStartStk[TASK_STK_SIZE];
OS_STK TaskClkStk[TASK_STK_SIZE];
OS_STK Task1Stk[TASK_STK_SIZE];
OS_STK Task2Stk[TASK_STK_SIZE];
OS_STK Task3Stk[TASK_STK_SIZE];
OS_STK Task4Stk[TASK_STK_SIZE];
OS_STK Task5Stk[TASK_STK_SIZE];

TASK_USER_DATA TaskUserData[7];

OS_EVENT *MsgQueue;
void *MsgQueueTbl[20];
```

Example 3 - main()



```
void main(void)
{
    PC_DispClrScr(DIS_BGND_BLACK);
    OSInit();
    PCDOSSaveReturn();
    PC_VectSet(uCOS, OSCtxSw);
    PC_ElapsedInit();
    strcpy(
TaskUserData[TASK_START_ID].TaskName,
    "StartTask");
    OSTaskCreateExt(TaskStart,
                    (void *)0,
                    &TaskStartStk[TASK_STK_SIZE -
                    1],
                    TASK_START_PRIO,
                    TASK_START_ID,
                    &TaskStartStk[0],
                    TASK_STK_SIZE,
                    &TaskUserData[TASK_START_ID],
                    0);
    //the TCB of each task can store
    // a ptr to its user-provided data
    //structure
    OSStart();}
//task ,pdata ,ptos ,task prioity ,task id num
//pbos ,Stk_size
//ptr to user-supplied memory loc
//task specific options(low8 os,high8 user)
```

Example 3 – void TaskStart(void *pdata)



```
void TaskStart(void *pdata)
{
#if OS_CRITICAL_METHOD == 3
    OS_CPU_SR cpu_sr;
#endif
    INT16S key;

    pdata = pdata;

    TaskStartDisplnit();

    OS_ENTER_CRITICAL();
    PC_VectSet(0x08,OSTickISP);

    PC_SetTickRate(OS_TICKS_PER_SE
    C);
    OS_EXIT_CRITIACL();

    OSStartInit();
```

```
MsgQueue =
    OSQCreate(&MsgQueueTbl[0],
    MSG_QUEUE_SIZE);
// OSQCreate( **start ,size)
// Task1~Task4 ,by strcpy
TaskStartCreateTasks();
//using OSTaskCreateExt
//TaskClk,Task1 ~ Task5
for(;;){
    TaskStartDisp();

    if(PC_GetKey(&key) == TURE){
        if(key == 0x1B){
            PC_DOSReturn();
        }
    }
    OSCtxtSwCtr = 0 ;
    OSTimeDly(OS_TICKS_PER_SEC);
}
}
```

Example 3 – void Task1,2 (void *pdata)



```
void Task1 (void *pdata)
{
    char    *msg;
    INT8U   err;

    pdata = pdata;

    for(;;){
        msg = (char
        *)OSQPend(MsgQueue, 0, &err);
        PC_Dispatch(70, 13, msg,
        DISP_FGND_YELLOW +
        DISP_BGND_BLUE);
        OSTimeDlyHMSM(0,0,0,100);
    }
}
```

```
void Task2 (void *pdata)
{
    char    msg[20];

    pdata = pdata;
    strcpy(&msg[0], "Task 2");
    for(;;){
        OSQPost(MsgQueue, (void
        *)&msg[0]);
        OSTimeDlyHMSM(0,0,0,500);
    }
}
```

Example 3 – void Task3,4 (void *pdata)



```
void Task3 (void *pdata)
{
    char    msg[20];

    pdata = pdata;
    strcpy(&msg[0] ,"Task 3");
    for(;;){
        OSQPost(MsgQueue, (void
        *)&msg[0]);

        OSTimeDlyHMSM(0,0,0,500);
    }
}
```

```
void Task4 (void *pdata)
{
    char    msg[20];

    pdata = pdata;
    strcpy(&msg[0] ,"Task 4");
    for(;;){
        OSQPost(MsgQueue, (void
        *)&msg[0]);

        OSTimeDlyHMSM(0,0,0,500);
    }
}
```


Example 3 – uC /OS-II hooks



```
void OSInitHookBegin(void);
void OSInitHookEnd(void);
void OSTaskCreateHook(OS_TCB *ptcb);
void OSTaskDelHook(OS_TCB *ptcb);
void OSTaskIdleHook(void);
void OSTaskStatHook(void);
void OSTaskSwHook(void);
void OSTCBInitHook(OS_TCB *ptcb);
void OSTimeTickHook(void);
```

```
void OSInitHookBegin(void)
{
}
```

```
void OSInitHookEnd(void)
{
}
```

```
void OSTaskCreateHook(OS_TCB *ptcb)
{
    ptcb = ptcb;
}
void OSTaskDelHook(OS_TCB *ptcb)
{
    ptcb = ptcb;
}
void OSTaskIdleHook(void)
{
}
void OSTCBInitHook(OS_TCB *ptcb)
{
    ptcb = ptcb;
}
void OSTimeTickHook(void)
{
}
```

Example 3 – OSTaskSwHook(void)



```
void OSTaskSwHook(void)
{
    INT16U      time;
    TASK_USER_DATA *puser;

    time = PC_ElapsedStop();
    PC_ElapsedStart();
    puser = OSTCBCur->OSTCBExtPtr;
    if(puser != (TASK_USER_DATA *)0){
        puser->TaskCtr++;
        puser->TaskExecTime =time;
        puser->TaskToExecTime +=time;
    }
}
```

Example 3 – OSTaskStatHook(void)

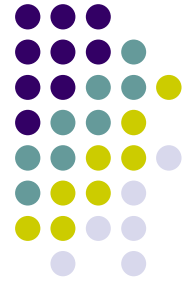


```
void OSTaskStatHook(void)
{
    char  s[80];
    INT8U i;
    INT32U total;
    INT8U pct;
    total =0L;

    for(i=0; i<7; i++) {
        total += TaskUserData[i].TaskToExecTime;
        DispTaskStat(i);
        //user-define function, PC_DispatchStr in
        //the proper location
    }

    if(total > 0){
        pct = 100 *
            TaskUserData[i].TaskToExecTime / total;

        sprintf(s,"%3d %%",pct);
        PC_DispatchStr(62,
            i+11,
            s,
            DISP_FGND_BLACK +
            DISP_BGND_LIGHT_GRAY);
    }
    if(total > 1000000000L){
        for (i = 0; i < 7; i++) {
            TaskUserData[i].TaskTotExecTime = 0L;
        }
    }
}
```



Example 4

```
C:\EX4_x86L_FP\BC45\TEST\TEST.EXE
uC/OS-II, The Real-Time Kernel
Jean J. Labrosse

EXAMPLE #4

TaskPrio      Angle      cos(Angle)  sin(Angle)
-----
 1           48.978      0.656        0.754
 2           84.983      0.087        0.996
 3          120.983     -0.515       0.857
 4          156.973     -0.920       0.391
 5          192.973     -0.974      -0.224
 6          228.973     -0.656      -0.754
 7          264.987     -0.087      -0.996
 8          300.993      0.515      -0.857
 9          336.993      0.920      -0.391
10           12.970      0.974       0.224

#Tasks       : 13   CPU Usage: 0 %
#Task switch/sec: 2202

80387 FPU
V2.52
<-PRESS 'ESC' TO QUIT->
```

- Example 4 use Ix86L-FP(13tasks ,1,1,1,10)

Example 4 –static void TaskStartCreateTasks (void)



```
static void TaskStartCreateTasks (void)
{
    INT8U i;
    INT8U prio;                                //task

                                            //pdata

    for (i = 0; i < N_TASKS; i++) {           //ptos
        prio    = i + 1;                       //task prioity
        TaskData[i] = prio;                    //task id num
        OSTaskCreateExt(Task,                  //pbos
            (void *)&TaskData[i],
            &TaskStk[i][TASK_STK_SIZE - 1], //Stk_size
            prio,                               //ptr to user-supplied memory loc
            0,                                  //task specific options(low8 os,high8 user)
            &TaskStk[i][0],
            TASK_STK_SIZE,                      //OS_TASK_OPT_STK_CHK
            (void *)0,                          //OS_TASK_OPT_STK_CLR
            OS_TASK_OPT_SAVE_FP);              //OS_TASK_OPT_SAVE_FP
    }
}
```

Example 4

void Task (void *pdata)



```
void Task (void *pdata)
{
    FP32 x;
    FP32 y;
    FP32 angle;
    FP32 radians;
    char s[81];
    INT8U ypos;

    ypos = *(INT8U *)pdata + 7;
    angle = (FP32)*(INT8U *)pdata *
            (FP32)36.0;
```

```
    for (;;) {
        radians = (FP32)2.0 *
            (FP32)3.141592 * angle / (FP32)360.0;
        x = cos(radians);
        y = sin(radians);
        sprintf(s, " %2d %8.3f %8.3f
%8.3f", *(INT8U *)pdata, angle, x, y);
        PC_DispatchStr(0, ypos, s,
            DISP_FGND_BLACK +
            DISP_BGND_LIGHT_GRAY);
        if (angle >= (FP32)360.0) {
            angle = (FP32)0.0;
        } else {
            angle += (FP32)0.01;
        }
        OSTimeDly(1);
    }
}
```