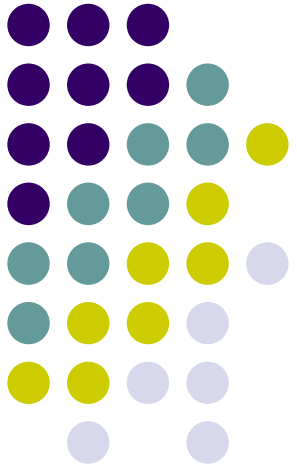
CH1

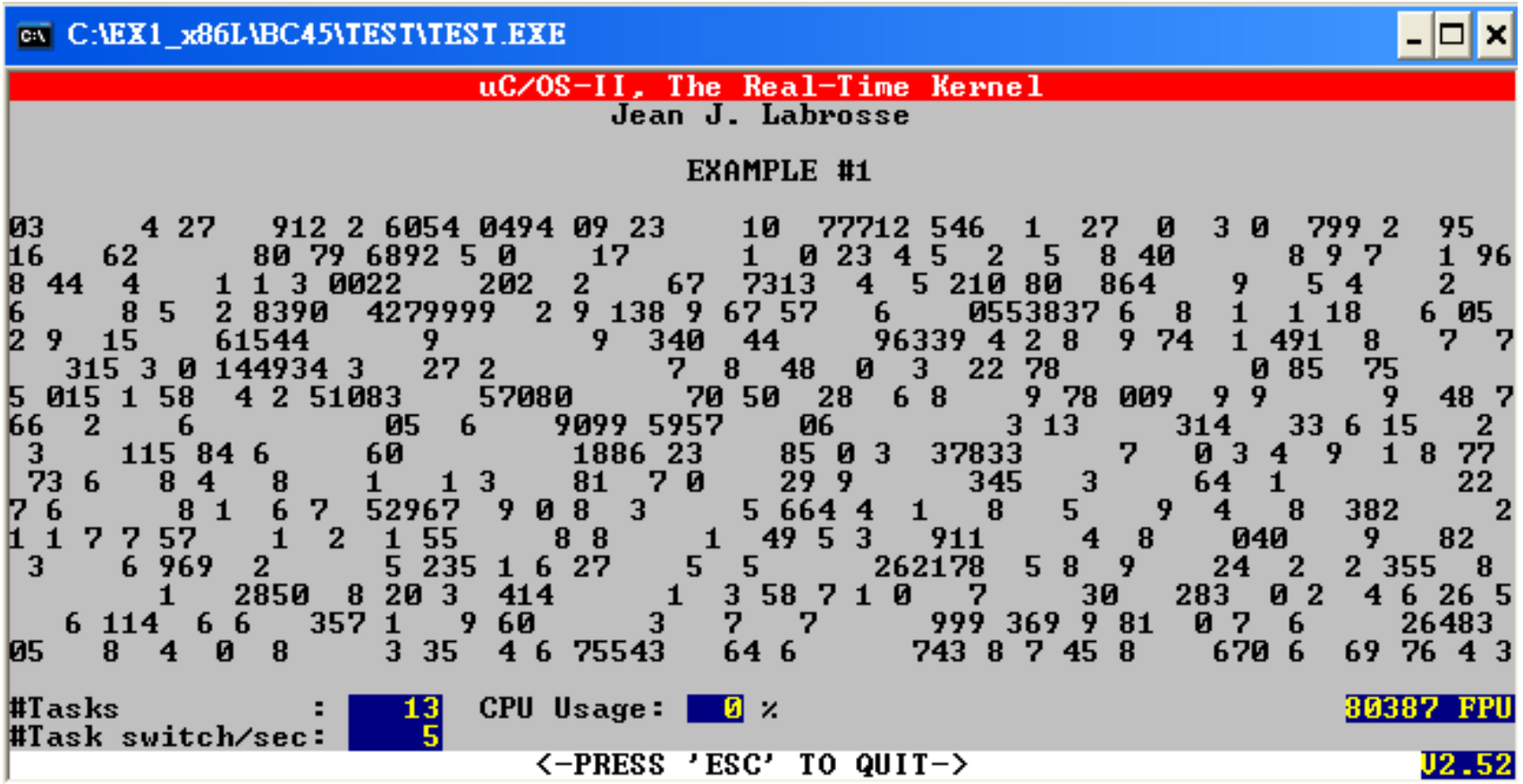
Getting Started with μC/OS-II



Date : 2007/07/18

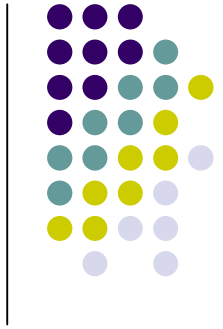
Speaker:Ming-Shyong Tsai

Example 1



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

Example 1

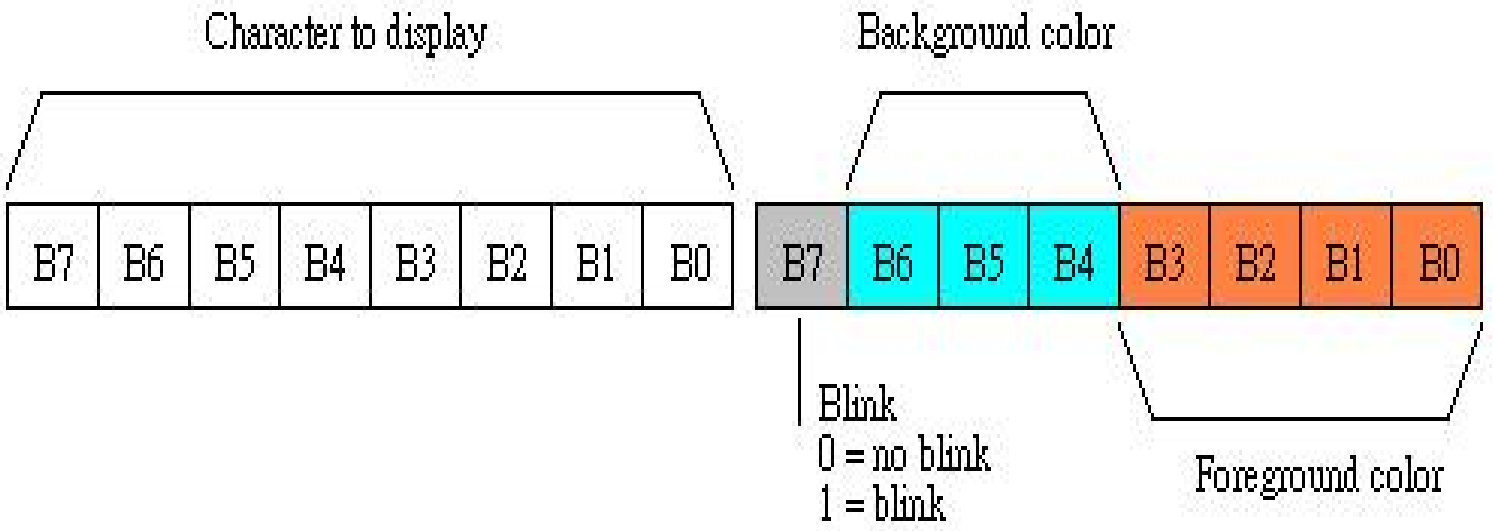


```
#include "includes.h"

#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;
```

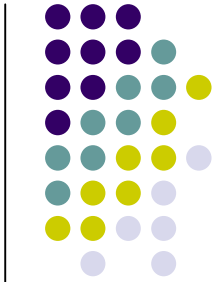
Example 1 – main（）

```
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

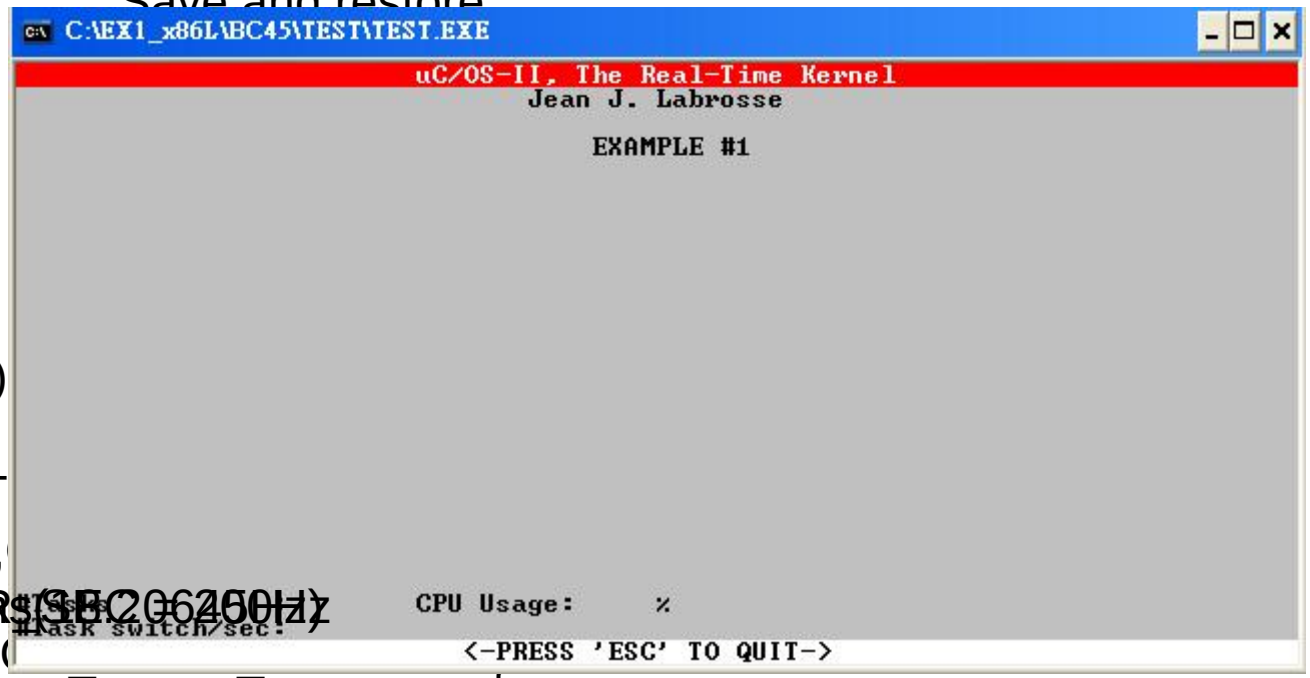


Example 1
TaskStart(void *pdata)



```
void TaskStart(void *pdata)
{
#if OS_CRITICAL_METHOD == 3
    OS_CPU_SR cpu_sr;
#endif
    char      s[100];
    INT16S    key;

    pdata = pdata;

    TaskStartDispInit()

    OS_ENTER_CRIT
    PC_VectSet(0x08,
      OSTickISR);
     PC_SetTickRate(
      _SEC);
    OS_EXIT_CRITIACL();
    OSStatInit();
      //be called to determine CPUspeed
      TaskStartCreateTasks();
```

Save and restore

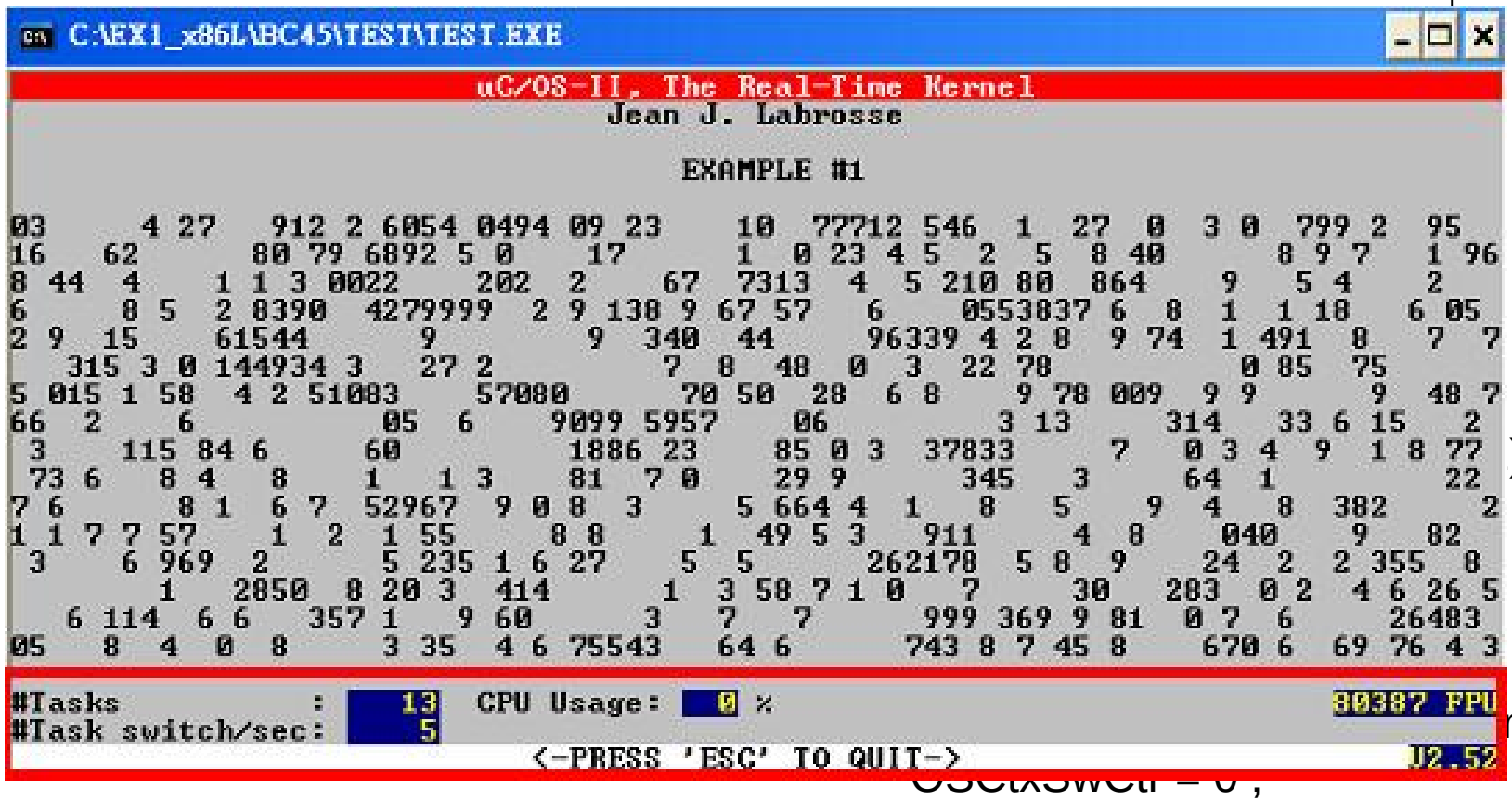



```
}
```

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)

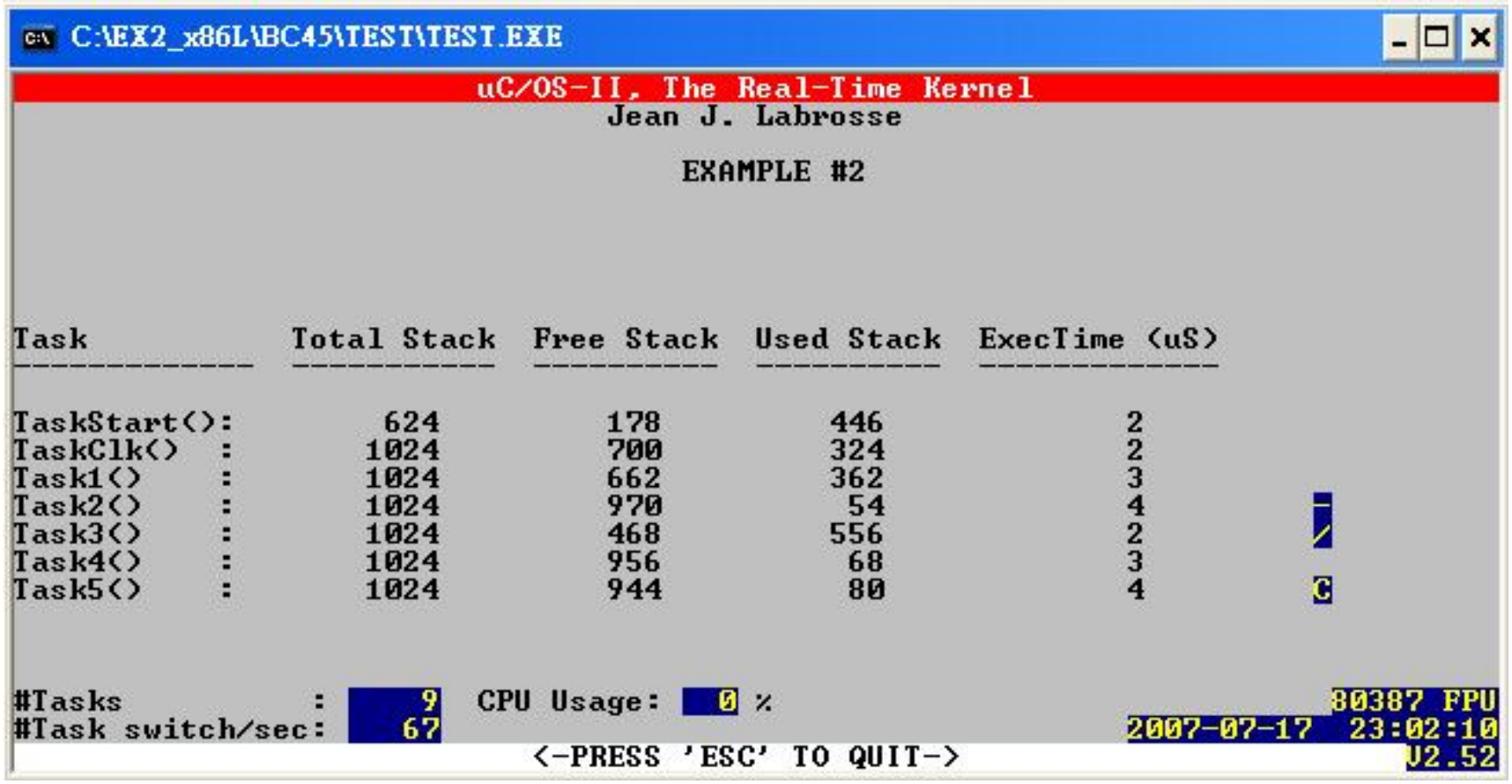



```
){
red
    OSCtxSwCtr = 0;
    OSTimeDlyHMSM(0,  0,  1,  0);
  }
}
```

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



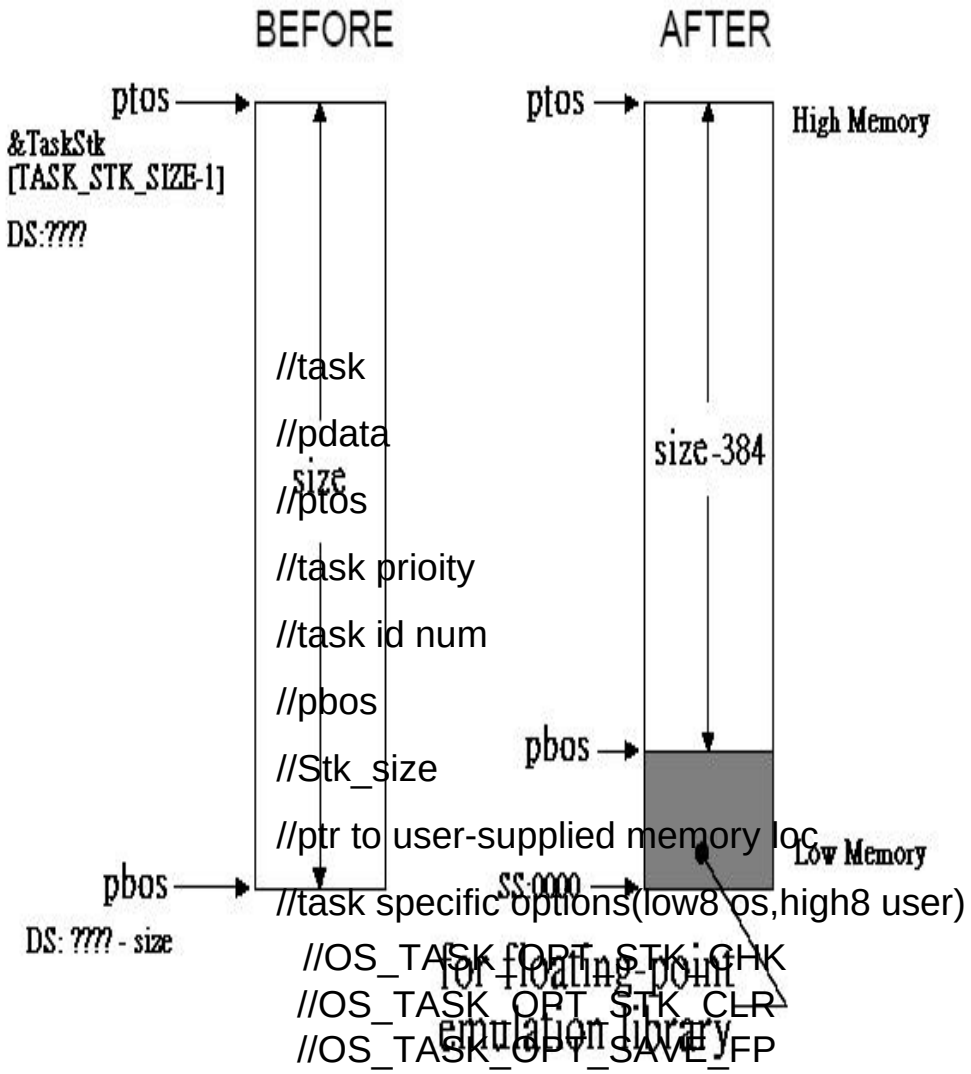
- Example 2 show stack-checking feature of uC/OS-II

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;
```

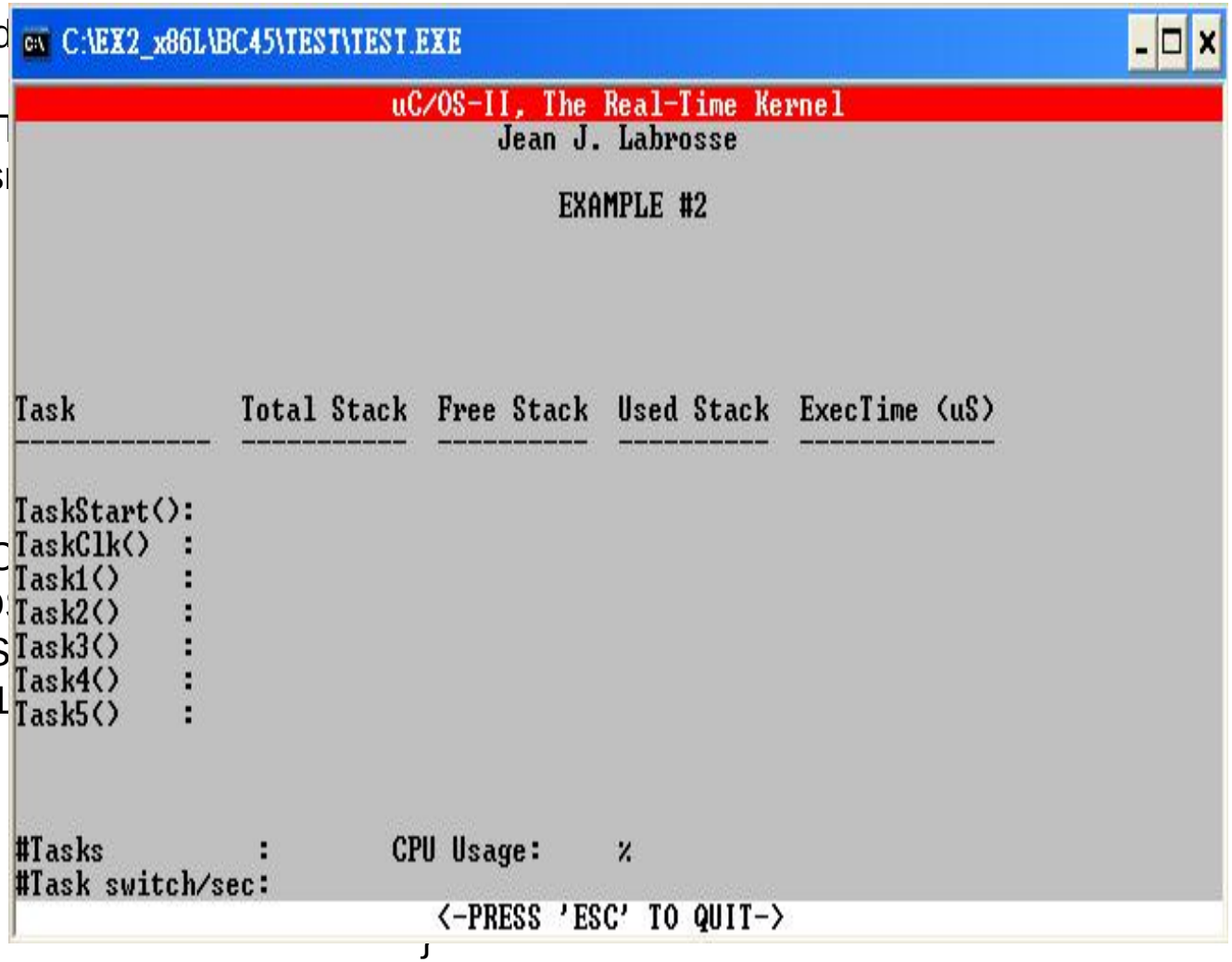
Example 2 – main()



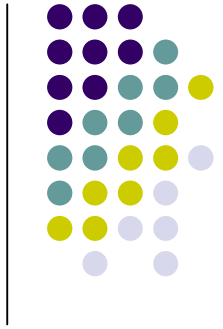

```
ptos    =
  &TaskStartStk[TASK_STK_SIZE - 1];
pbos    = &TaskStartStk[0];
size    = TASK_STK_SIZE;
OSTaskStkInit_FPE_x86(
                &ptos,& pbos,&size);

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 *pd
{
#if OS_CRITICAL_ME
    OS_CPU_SR cpu_s
#endif
    INT16S    key;

    pdata = pdata;

    TaskStartDispInit();

    OS_ENTER_CRITIC
    PC_VectSet(0x08,O
    PC_SetTickRate(OS
    OS_EXIT_CRITIACL

    OSStatInit();
```

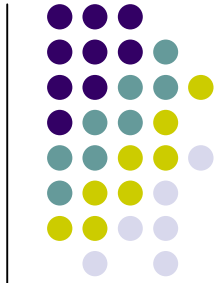


Example 2 – void Task1 (void *pdata)



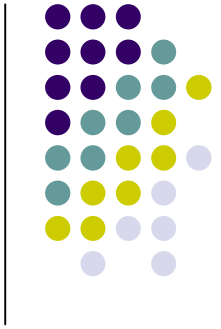
```
void Task1 (void *pdata)
{
    INT8U              err;
    OS_STK_DATA data;
    INT16U             time;
    INT8U              i;
    char               s[80];

    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)



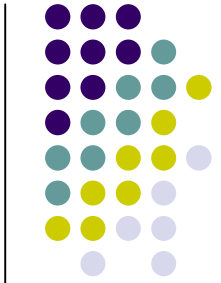
```
void Task2 (void *data)
{
   data = data;
   for(;;) {
      PC_DispChar(70, 15, '|',  DISP_FGND_WHITE + DISP_BGND_RED);
      OSTimeDly(10);
      PC_DispChar(70, 15, '/',  DISP_FGND_WHITE + DISP_BGND_RED);
      OSTimeDly(10);
      PC_DispChar(70, 15, '-',  DISP_FGND_WHITE + DISP_BGND_RED);
 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

Example 2 – void Task3 (void *data)



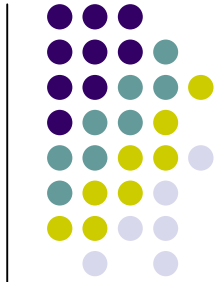
```
void Task3 (void *data)
{
   char dummy[500];
   INT16U i;

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

   for(;;) {
      PC_DispChar(70, 16, '|',  DISP_FGND_WHITE + DISP_BGND_BLUE);
      OSTimeDly(20);
      PC_DispChar(70, 16, '\\', DISP_FGND_WHITE + DISP_BGND_BLUE);
      OSTimeDly(20);
      PC_DispChar(70, 16, '-',  DISP_FGND_WHITE + DISP_BGND_BLUE);
      OSTimeDly(20);
      PC_DispChar(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)

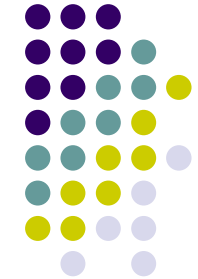


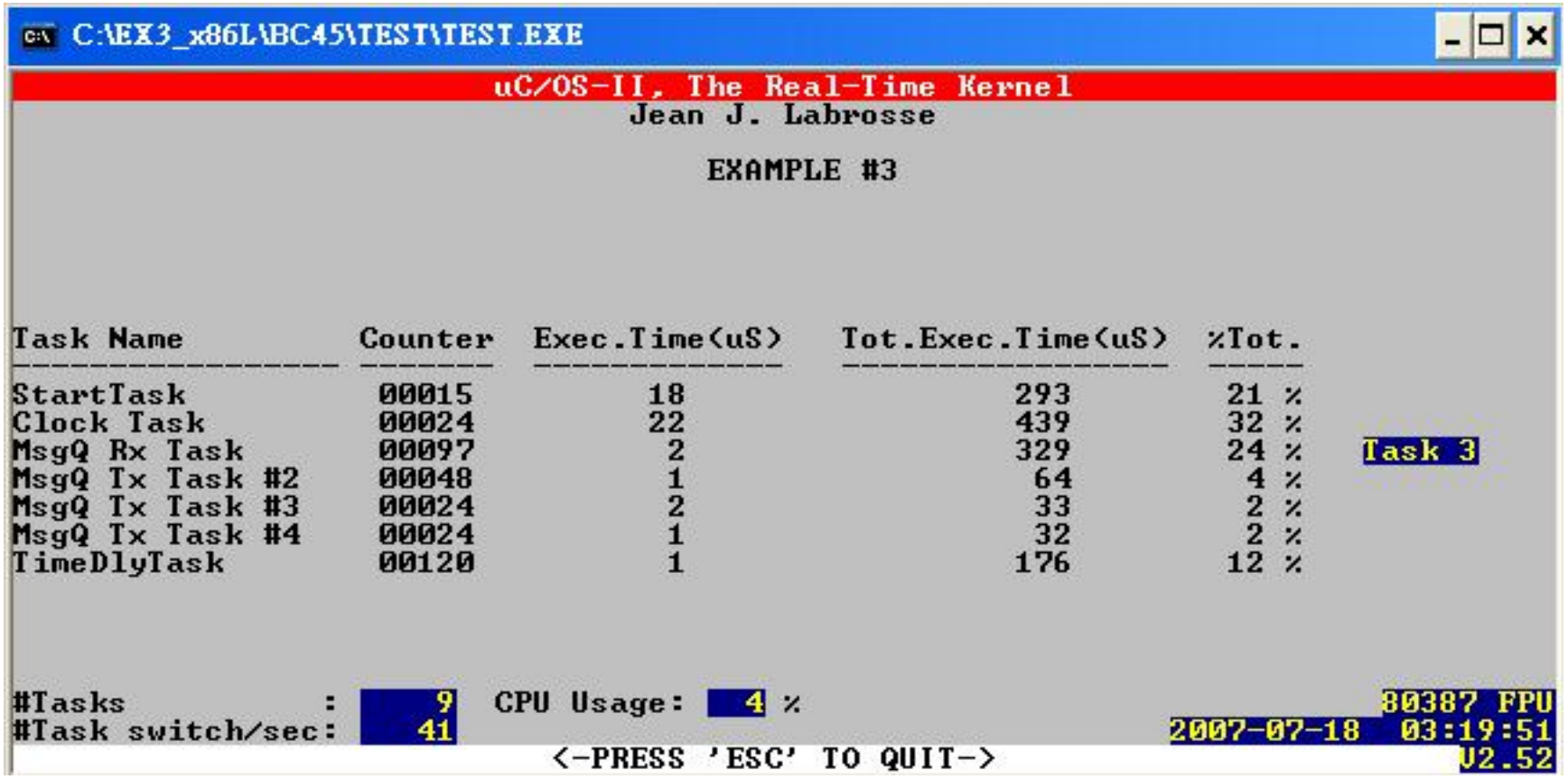
```
void Task5(void *data)
{
    char   rxmsg;
    INT8U  err;

    for(;;) {
      rxmsg = (char*)OSMboxPend(TxMbox, 0, &err);
      PC_DispChar(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);
    }
}
```

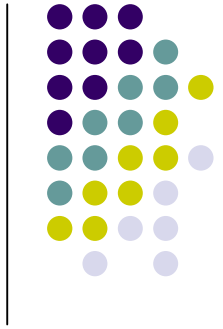
Example 3





- 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];
```

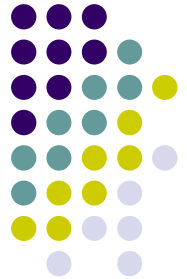
Example 3 - main()

```
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)
```

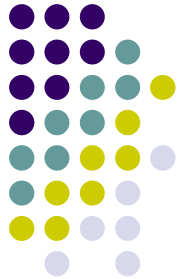
Example 3 – void TaskStart(void *pdata)



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

    pdata = pdata;

    TaskStartDispInit();

    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);
    }
}
```

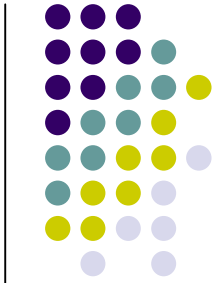
Example 3 – void Task1,2 (void *pdata)



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

    pdata = pdata;

    for(;;){
      msg = (char
    *)OSQPend(MsgQueue, 0, &err);
      PC_DispStr(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

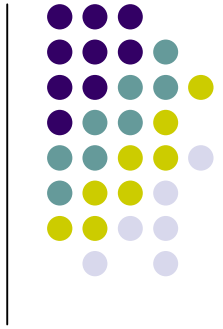
```
void OSInitHookBegin(void);
void OSInitHookEnd(void);
void OSTaskCreateHook(OS_TCB *ptcb);
void OSTaskDelHook(OS_TCB *ptcb);
void OSTaskIdleHook(void);
void OSTaskStatHook(void);
void OSTaskSwHook(void);
void OSTCBInitHook(OS_TCB *ptcb);
void OSTimeTickHook(void);

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)

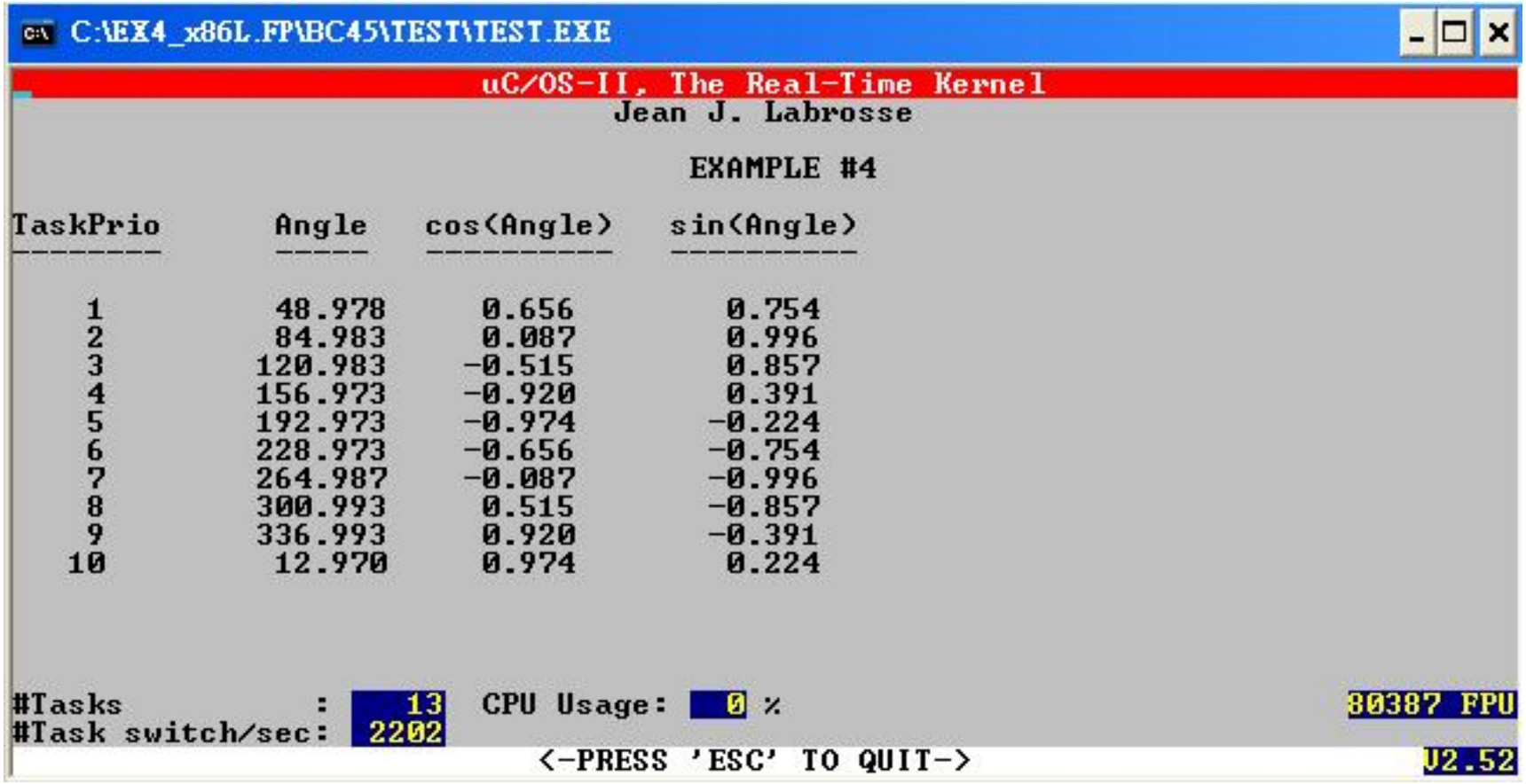
```
void OSTaskSwHook(void)
{
    INT16U          time;
    TASK_USER_DATA   *puser;

    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)



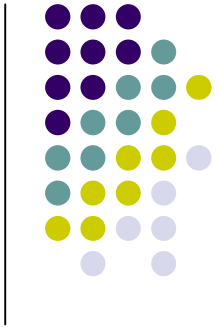
```
void OSTaskStatHook(void)
{
    char    s[80];
    INT8U   i;
    INT32U  total;
    INT8U   pct;
    total  =0L;

    for(i=0; i<7; i++) {
       total += TaskUserData[i].TaskToExecTime;
       DispTaskStat(i);
      //user-define function, PC_DispStr in
      //the proper location
    }
if(total > 0){
      pct = 100 *
      TaskUserData[i].TaskToExecTime / total;
sprintf(s,"%3d %%",pct);
      PC_DispStr(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



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

Example 4 –static void TaskStartCreateTasks (void)



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

    for (i = 0; i < N_TASKS; i++) {
        prio       = i + 1;
        TaskData[i] = prio;
        OSTaskCreateExt(Task,
                        (void *)&TaskData[i],
                        &TaskStk[i][TASK_STK_SIZE - 1],
                        prio,
                        0,
                        &TaskStk[i][0],
                        TASK_STK_SIZE,
                        (void *)0,
                        OS_TASK_OPT_SAVE_FP);
    }
}
```

//task

//pdata

//ptos

//task prioity

//task id num

//pbos

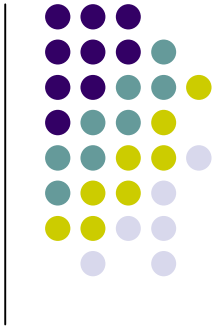
//Stk_size

//ptr to user-supplied memory loc

//task specific options(low8 os,high8 user)

//OS_TASK_OPT_STK_CHK
//OS_TASK_OPT_STK_CLR
//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_DispStr(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);
    }
}
```