INTER-PROCESS COMMUNICATION AND SYNCHRONISATION:

Lesson-05:
Inter process communication

1. Inter Process Communication

Need of Inter Process Communication

- Assume that there is need to send through the kernel an output data (a message of a known size with or without a header or a flag to notify an event) for processing or taking note of by another task

Inter Processor Communication

- Inter process communication from a process (task or thread to another) for a message
- Inter processor communication in a multiprocessor system─ used to generate information about certain sets of computations finishing on one processor and to let the other processors waiting for finishing the computations take note of the information

Need of Inter Process Communication

- Global variables problems
- Shared data
- No encapsulation of the data or message accessibility by other tasks
- OS Inter Process Communication (IPC) functions based solutions

Inter Process Communication (IPC)

- means that a process (scheduler, task or ISR) generates some information by signal (for other process start) or value (for example of semaphore) or generates an output so that it lets another process take note or use it through the kernel functions for the IPCs

Inter Process Communication (IPC) in Multitasking System

- Used to signal for other process to start or
- post a token or flag or
- generate message from the certain sets of computations finishing on one task and to let the other tasks take note of signal or get the message

Operating System Provisioning for the IPC functions

- Signal for other process to start
- Semaphore (*as token*, *mutex*) or counting semaphores for the inter task communication between tasks sharing a common buffer
- Queue,
- Mailbox
-

2. Example of mutex Inter Process Communication for print

Inter Process Communication for print

- *print* task can be shared among the multiple tasks, which use the mutex semaphore IPC in their critical sections
- When the buffer becomes available for new data, an IPC from the *print* task is generated and the kernel first takes note of it.
- Other tasks then take note of the IPC.

Inter Process Communication for print

- A task take note of IPC by OSSemPend ( ) function– used at the beginning of the critical section
- The task gets mutually exclusive access to the section to send messages into the print-buffer by using the OSSemPost ( ) function of the kernel at the end of the section

3. Example of semaphore and mailbox Inter Process Communication for display of date and time

Processes for display of updated date and time

- A mobile phone device *Update_Time task*
- A task, *Task_Display* for a multi-line display of outputs which displays current time on last line

Semaphore Inter Process Communication for display of date and time

- When the multi-line display task finishes the display of the last but one line, an IPC semaphore $s_{updateTD}$ from the *display* task is posted and the kernel takes note of it.
- The task — continuously updating time — then takes the $s_{updateTD}$

Coding Example of posting semaphore

```
static void Task_Display  (void
  *taskPointer)  {
.
  while (1) {
.
.   /* IPC for requesting TimeDate */
  OSSemPost (supdateTD) /* Post for the
   semaphore supdateTD.
```

Coding Example of waiting for mailbox message

```
. /* IPC for waiting time and date in the mailbox
     */ TimeDateMsg = OSMboxPend (timeDate)
     /* Wait for the mailbox message timeDate.
     The timeDate becomes null after the
     TimeDateMsg equals the updated time and
     date */
/* Code for display TimeDateMsg Time: hr:mm
     Date: month:date  */
.
};
```

Coding Example of waiting for the semaphore

```
static void Task_ Update_Time  (void
    *taskPointer)  {
.
while (1) {
OSSemPend (supdateTD) /* Wait the
    semaphore supdateTD. supdateT  becomes 0
    after taking the semaphore */
. /* Codes for updating time and date as per the
    number of clock interrupts received so far */
```

Coding Example for posting the mailbox message

/* Codes for writing into the mailbox */
OSMboxPost (*timeDate*) /* Post for the mailbox message and *timeDate, which equaled null earlier, now equals new updated time and date*/*

.

};

An Alternative for IPC functions

- Pipe device
- Socket device
- Remote procedure call (RPC) for distributed processes .

Summary

We learnt

- Inter process communication (IPC) means that a process (scheduler or task or ISR) generates some information by setting or resetting a Token or value, or generates an output so that it lets another process take note or to signal to OS for starting a process or use it under the control of an OS

End of Lesson 5 of Chapter 9 on Inter Process Communication