REAL TIME OPERATING SYSTEMS

Lesson-2:
PROCESS MANAGEMENT

1. Process Creation

Process creation

- Step 1: At the reset of the processor. in a computer system, an OS is initialized first—enabling the use of the OS functions, which includes the function to create the processes

Step 2: Using OS process creation function, a process, which can be called initial process, is created.

Step 3: OS started and that calls the initial process to run.

Process creation

- **Step 4: When the initial process runs, it creates subsequent processes.**
  Processes can be created hierarchically.
- **OS schedules the threads and provide for context switching between the threads (or tasks).**

Creation of a process

- Means defining the resources for the process and address spaces (memory blocks) for the created process, its stack, its data and its heap and placing the process initial information at a PCB

PCB

- (a) Context
- (ii) Process stack pointer
- (iii) Current state [Is it created, activated or spawned? Is it running? Is it blocked?]
- (iv) Addresses that are allocated and that are presently in use

PCB

- (v) Pointer for the parent process
- (vi) Pointer to a list of daughter processes.
- (vii) Pointer to a list of resources, which are usable (consumed) only once. For examples, input data, memory buffer or pipe, mailbox message, semaphore
- (viii) Pointer to a list of resources, which are usable (consumed) only once

PCB

- *(ix) Pointer to queue of messages.*
- *(x) Pointer to Access-permissions descriptor*
- *(xi) ID*

2. Example of Process Creation

OS_Task_Create ( ) function

- OS function creates a process using OS_Task_Create ( ) function Task_Send_Card_Info in the mai
- Task_Send_Card_Info task creates two other tasks, Task_Send_Port_Output and Task_Read_Port_Input.
- OS then controls the context switching between the processes

OS Process Creation function

OS function first creates the *Display_process*.

Display_process creates—

- Display_Time_DateThread
- Display_BatteryThread
- Display_SignalThread
- Display_ProfileThread
- Display_MessageThread
- Display_Call StatusThread
- Display_MenuThread

3. Message passing and System call to OS by Processes

Message Passing by process running in user mode

- Generates and puts (sends) a message
- OS lets the requested resource (for example, input from a device or from a queue) use or run an OS service function (for example, define a delay period after which process needs to be run again).
- A message can be sent for the OS to let the LCD display be used by a task or thread for sending the output.
- An ISR sends a message to a waiting thread to start on return from the ISR

System call by process running in user mode

- Call to a function defined at the OS.
- For example, OSTaskCreate ( )─ to create a task.
- First an SWI instruction is issued to trap the processor and switch to supervisory mode.
- OS then executes a function like a library function
- Processor on finishing the instructions of a called function, the processor again switches back to user mode and lets the calling process run further

4. Process Manager Functions

Process manager

- (i) makes it feasible to let for a process to sequentially execute or block when needing a resource and to resume when it becomes available,
- (ii) implements the logical link to the resource manager for resources management (including scheduling of process on the CPU),

Process manager

- (iii) allows specific resources sharing between specified processes only,
- (*iiiv*) allocates the resources as per the resource allocation- mechanism of the system and
- (iv) manages the processes and resources of the given system.

Summary

We learnt

- Process manager has functions to create the processes,
- allocates a PCB to each process,
- manages access to the resources and facilitates the switching from one process state to another.
- The PCB defines the process structure for a process state.

We learnt

- Process can send a message and make a system call to enable OS actions and run OS function

End of Lesson 2 of Chapter 8