Ex. No: 8	Implementation of the concept of Process Scheduling in any IoT Environment
13-08-2024	

Aim: To prepare a report on ***implemention of concepts of process scheduling*** in any IoT environment**.**

Introduction

In an Internet of Things (IoT) environment, multiple devices and sensors generate a vast amount of data, which needs to be processed and analyzed in real-time to extract valuable insights. Process scheduling plays a crucial role in managing these processes efficiently, ensuring that tasks are executed in a timely and organized manner. This report outlines the concept of process scheduling in an IoT environment, its importance, and a detailed implementation plan.

What is Process Scheduling?

Process scheduling is the mechanism by which the operating system allocates system resources, such as CPU time, memory, and I/O devices, to various processes or tasks. In an IoT environment, process scheduling is essential to manage the concurrent execution of multiple tasks, such as data processing, sensor readings, and communication with the cloud or other devices.

Process Scheduling Algorithms

Several process scheduling algorithms can be used in IoT environments, including:

1. **First-Come-First-Served (FCFS):** Each process is executed in the order it arrives in the ready queue.
2. **Shortest Job First (SJF):** The process with the shortest burst time is executed first.
3. **Priority Scheduling:** Processes are assigned priorities based on their importance, and the highest-priority process is executed first.
4. **Round Robin (RR):** Each process is allocated a fixed time slice (called a time quantum) to execute before the next process is scheduled.

Implementation

```
#include   <stdio.h>
#include  <stdlib.h>
#include <unistd.h>

#define  TIME_QUANTUM  10

typedef struct {
   int pid;
   int burst_time;
} Task;

Task task_queue[5];

void add_task(int pid, int burst_time) { Task
   task;
   task.pid            =            pid;
   task.burst_time = burst_time;
   task_queue[pid] = task;
}

void execute_task(int pid) {
   printf("Executing  task  %d  for  %d  milliseconds\n",  pid,  task_queue[pid].burst_time);
   usleep(task_queue[pid].burst_time * 1000);
}

void  rr_scheduling()  {
   int i = 0;
   while (1) {
      for (i = 0; i < 5; i++) {
         if  (task_queue[i].burst_time  >  0)  {
            execute_task(i);
            task_queue[i].burst_time  -=  TIME_QUANTUM;
            if (task_queue[i].burst_time <= 0) {
               task_queue[i].burst_time = 0;
            }
         }
      }
   }
}

int         main()         {
   add_task(0, 20);
   add_task(1, 30);
```

```
    add_task(2, 10);
    add_task(3, 25);
    add_task(4, 15);

    rr_scheduling();
    return 0;
}
```

Code Explanation:

Code snippet in C that demonstrates the **implementation of a simple Round Robin (RR) scheduling algorithm on a Raspberry Pi with Raspbian OS.**

This code creates a task queue with five tasks, each with a burst time in milliseconds. The rr_scheduling() function implements the Round Robin scheduling algorithm, executing each task for a time slice of 10 milliseconds before moving on to the next task.

Advantages and Challenges

Process scheduling in IoT environments offers several advantages, including:

Improved System Performance: Efficient process scheduling can improve system performance, reduce latency, and increase throughput.

Energy Efficiency: Process scheduling can help optimize energy consumption in IoT devices, which is critical for battery-powered devices.

However, process scheduling in IoT environments also poses several challenges, including:

Resource Constraints: IoT devices have limited resources, which can make it challenging to implement efficient process scheduling algorithms.

Scalability: IoT systems need to be scalable to accommodate increasing numbers of devices and sensors, which can make it challenging to design and implement efficient process scheduling algorithms

Conclusion

Process scheduling is a critical component of IoT environments, ensuring that tasks are executed efficiently and in a timely manner. By choosing a suitable process scheduling algorithm and implementing it on an IoT device, developers can optimize system performance, reduce latency, and improve overall system reliability. However, process scheduling in IoT environments also poses several challenges, including resource constraints, complexity, and scalability, which need to be addressed through careful design and implementation of process scheduling algorithms.

Result: Thus, a detailed report on implementation of concepts of process scheduling is prepared.