Chapter 11

Real Time Operating System

Lesson 01

Multitasking Systems

Example 1: Two tasks in the ECG recorder system

- One task initiates sensing
- Converts the analog signals into digital
- Saves the recorded data along with the patient, time and date and other information
- The data saves in memory in an appropriate format.

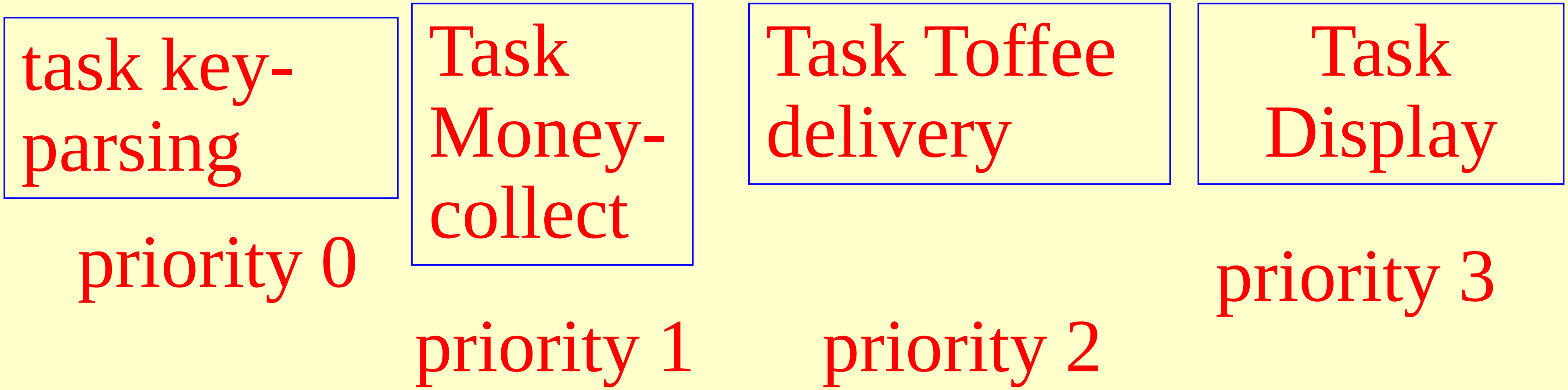
Two tasks in the ECG recorder system

- Second task retrieves from memory the patient information
- Displays the ECG on the screen with appropriate title and label

Example 2: Automatic Toffee Vending Machine

- Two tasks– one collecting coin and other delivering the toffee
- Task1 sends signal to a task 2 for toffee delivery after collecting the coin(s)

Automatic Toffee Vending Machine



task key-parsing

priority 0

Task Money-collect

priority 1

Task Toffee delivery

priority 2

Task Display

priority 3

Example 3: A controller

- Task 1 in a controller senses input and tasks sending signal on reaching the set temperature to another task 2
- Task 2 starts on receiving the waiting signal

First task in a digital camera system

- An image frame consists of pixels
- The pixels arrange horizontally in rows and vertically in columns
- One task initiates sensing of pixel intensities in the exposed rows of image frame
- Converts the analog signals for each pixel using an ADC
- Saves data in temporary memory-buffer M1

Second task in a digital camera system

- Second task senses the pixels in un-exposed rows (dark area rows) near the image frame
- Analog signals for each dark pixel converted using the ADC
- The data for each row averaged
- Average-values for each un-exposed row are saved in temporary memory-buffer M2
- Then M2 has the offset values for each row

Third task in a digital camera system

- Third task retrieves the pixel data from each row from M1
- Subtracts the offset for each row from M2
- Encodes (compresses) the off-set corrected image data for all rows in an appropriate format (jpg or gif)
- Saves the encoded image data along with image, time and date information in image data memory M.

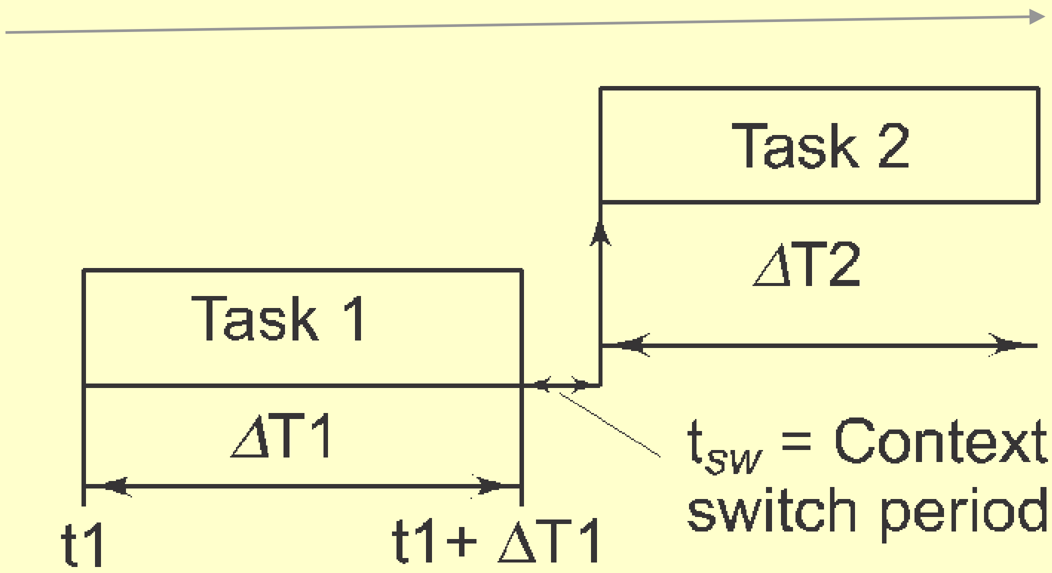
Fourth task in a digital camera system

- Fourth task retrieves from M the encoded information of image
- Displays the decoded image (decompressed data) on the LCD screen along with appropriate title and label

A two-task system

- task start at time t1 and second task start at time t2.
- Running of two tasks in a system CPU (or microcontroller) at two different time-intervals, $\Delta T1$ and $\Delta T2$
- t_{sw} is the task-context switching time interval. $(t2 – t1) = \Delta T1 + t_{sw}$

Time

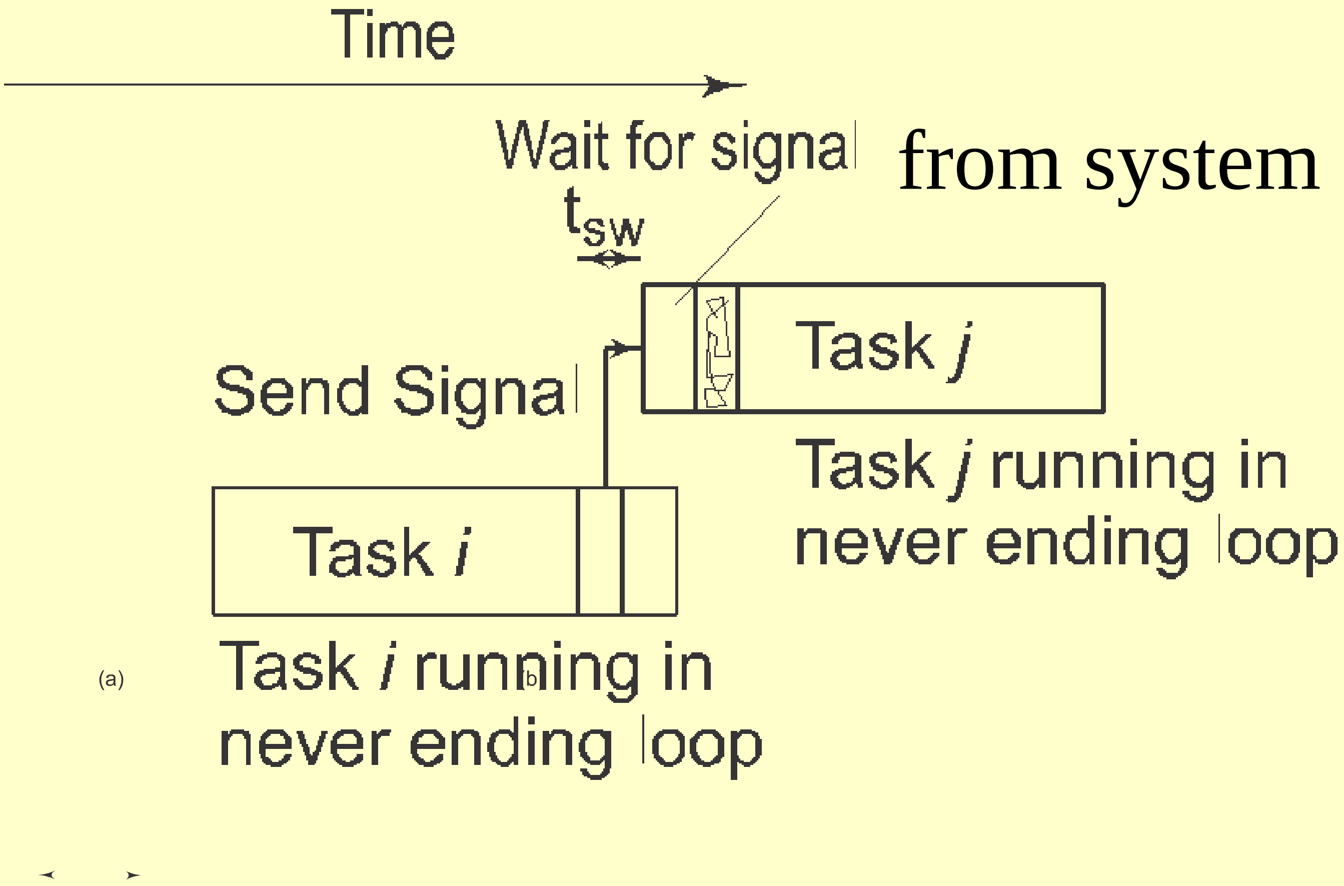


Task Context

- Context means information of the present program counter, stack pointer and other CPU registers
- When context of one task saves and the context of other task retrieve then that task runs
- Time called context-switching time, that is the time spent in saving the context and retrieving the new context

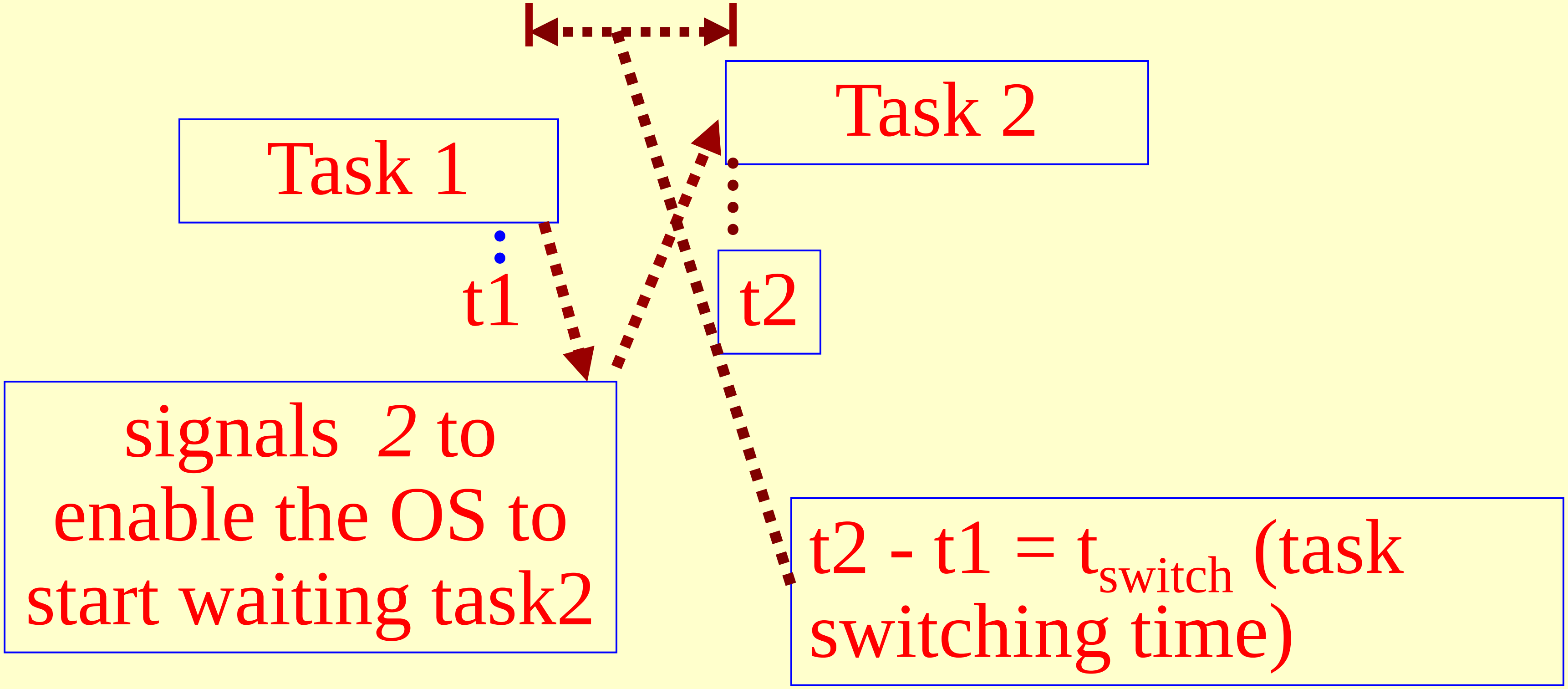
Task Waiting for a signal

- A task may wait for a signal


Time
Wait for signal
t_{sw}
Send Signal
Task *j*
Task *j* running in never ending loop
Task *i*
(a)
(b)
Task *i* running in never ending loop

from system

Signal between Two Tasks



Task 1

Task 2

t1

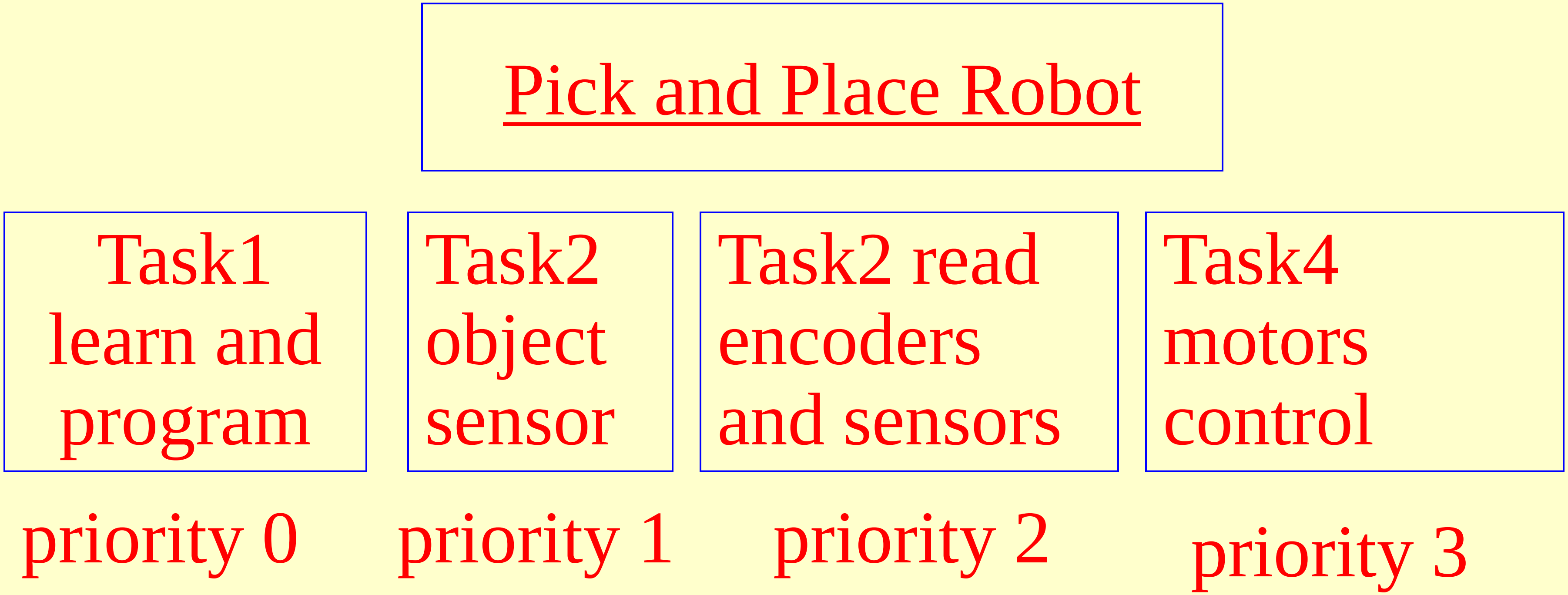
t2

signals 2 to enable the OS to start waiting task2

t2 - t1 = t_{switch} (task switching time)

Example 5– Pick and place Robot

- Fourth task retrieves from M the encoded information of image
- Displays the decoded image (decompressed data) on the LCD screen along with appropriate title and label



Pick and Place Robot

Task1 learn and program

Task2 object sensor

Task2 read encoders and sensors

Task4 motors control

priority 0

priority 1

priority 2

priority 3

Task

- Task - a process or thread in a multitasking system and the process or thread execution is controlled by an operating system

Task as a process or thread

- Process and thread–the terms used in UNIX and Linux
- Thread is a term used in Java
- Process and threads execute under the control of the OS or RTOS

Task as a process

- A *task* can be called a *process* because it performs a series of sequential operations to arrive at a result

Task as a process

- Process does not branch to another process by calling that process
- Process can only send signal or message to the system and the system passes it to another task.

Task as a thread

- A *task* can also be called a program *thread* because it performs a series of sequential operations and arrives at a result
- Use of the term thread is because a thread has no in-between branches that lead to another end point and thus has only one start point and one end point

Task as a thread

- Thread does not branch to another thread by calling that thread
- Thread can only send signal or message to the system and the system pass it to another thread

Task

- Task - a program under supervision of an OS (operating system)

Task

- Task in a multitasking system scheduled by operating system (OS) to run or blocked from running

Task characteristics

Only one task can run at an instance on a processor

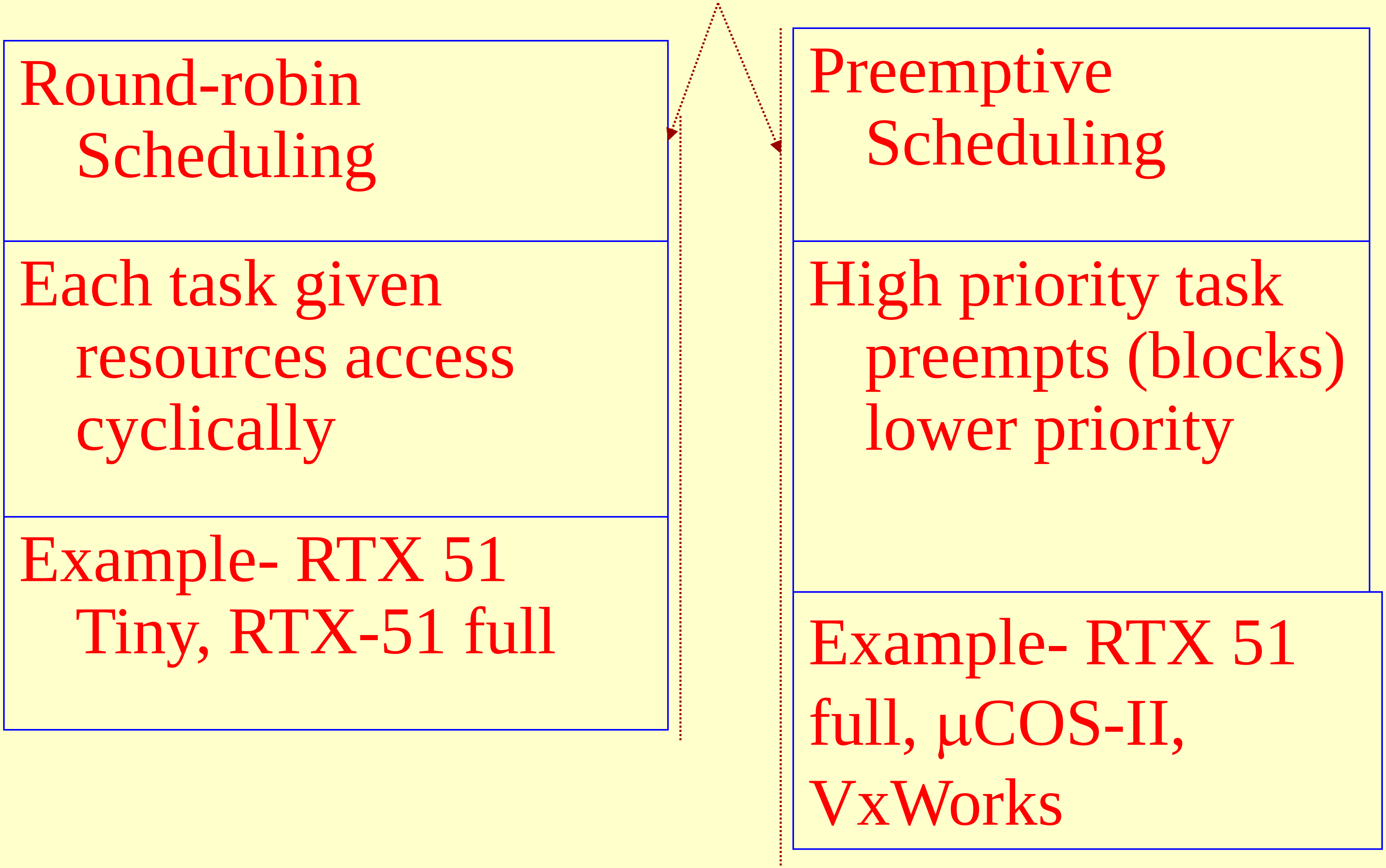
A task can’t call another task

A task can run functions and call the OS functions

A task can send an IPC- (for example, signal, token or message) for another task to enable OS unblocking that task [IPC (Inter-Process Communication)]

A task can wait for an IPC for OS unblocking

Scheduling of Tasks



Round-robin Scheduling

Each task given resources access cyclically

Example- RTX 51 Tiny, RTX-51 full

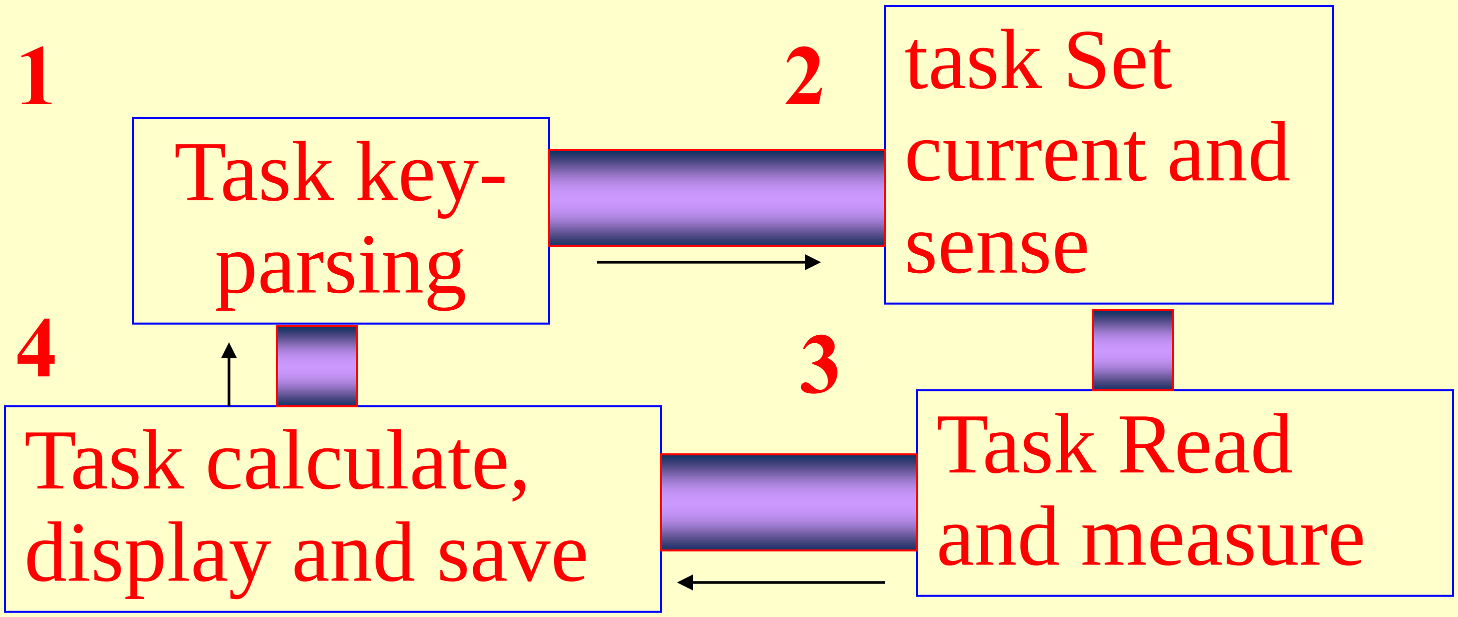
Preemptive Scheduling

High priority task preempts (blocks) lower priority

Example- RTX 51 full, µCOS-II, VxWorks

Round Robin Example

A continuous measuring instrument



1

Task key-parsing

2

task Set current and sense

3

Task Read and measure

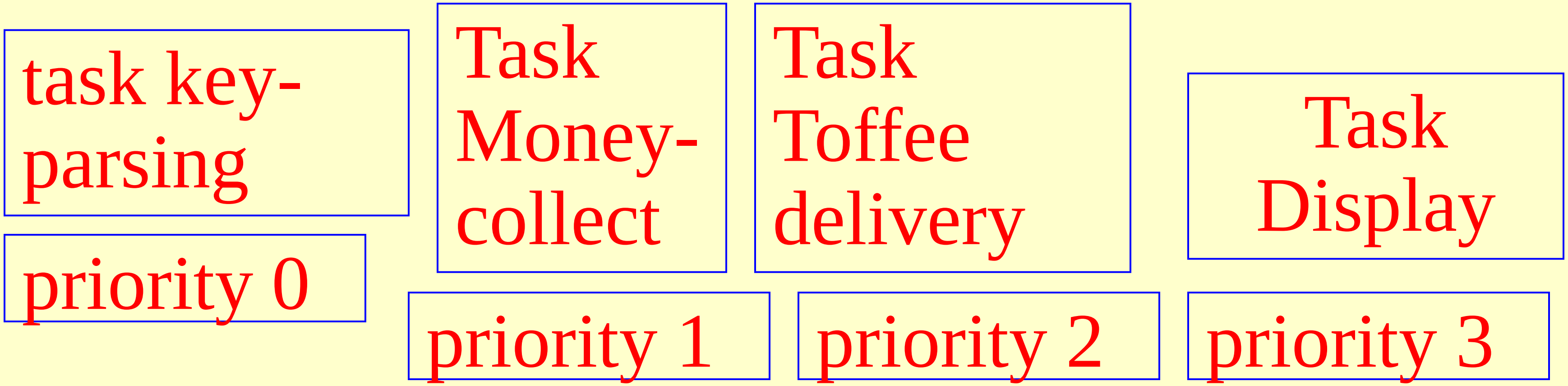
4

Task calculate, display and save

All tasks same priority

Priority Scheduling

When high priority task activates, then that preempts the low priority task running. l



task key-parsing

priority 0

Task Money-collect

priority 1

Task Toffee delivery

priority 2

Task Display

priority 3

Automatic Toffee Vending Machine

Task States

- One of the following states at any given instance —ready, deleted, running, timeout, waiting for a specific period, and waiting till some signal or message before some specific or undefined period
- An OS controls a task state

Task States

- Ready means gets a signal for which it is waiting or gets a message for which it is waiting or time out when its waiting-period is over

Summary

We learnt

- A system may consists of multiple tasks
- Task - a process or thread in a multitasking system
- Process or thread execution controlled by an operating system
- Task Context
- Context Switch

We learnt

- Task in One of the following states at any given instance —ready, deleted, running, timeout, waiting for a specific period, and waiting till some signal or message before some specific or undefined period
- An OS controls the task state
- Scheduling for running by the OS
- Round Robin scheduling
- Preemptive Priority based scheduling

End of Lesson 01 on

Multitasking Systems