DEVICE DRIVERS AND INTERRUPTS SERVICE MECHANISM

Lesson-3: Software Interrupts and Interrupt Service routines

Software Interrupt (Throw an Exception) Concept

- A program needs to detect error condition or run time exceptional condition encountered during the running.
- In a program either the hardware detects this condition or in a program detects this condition, then an instruction SWI (software interrupt) is used

Detection of exceptional run-time condition

- Called *throwing* an exception by the program.
- An interrupt service routine (exceptional handler routine) executes, which is called *catch* function as it executes on catching the exception thrown.

SWI

- Executes on detecting the exceptional run-time condition during computations or communication.
- For example, on detecting that the square root of a negative number is being calculated or detecting illegal argument in the function or detecting that *connection* to network not found.

Example: SWI a1 and SWI a2

- The SWI (software interrupt) instructions, *SWI a1* and *SWI a2* will be inserted for trapping (A- B) as –ve number and trapping y > 100 or less than 0

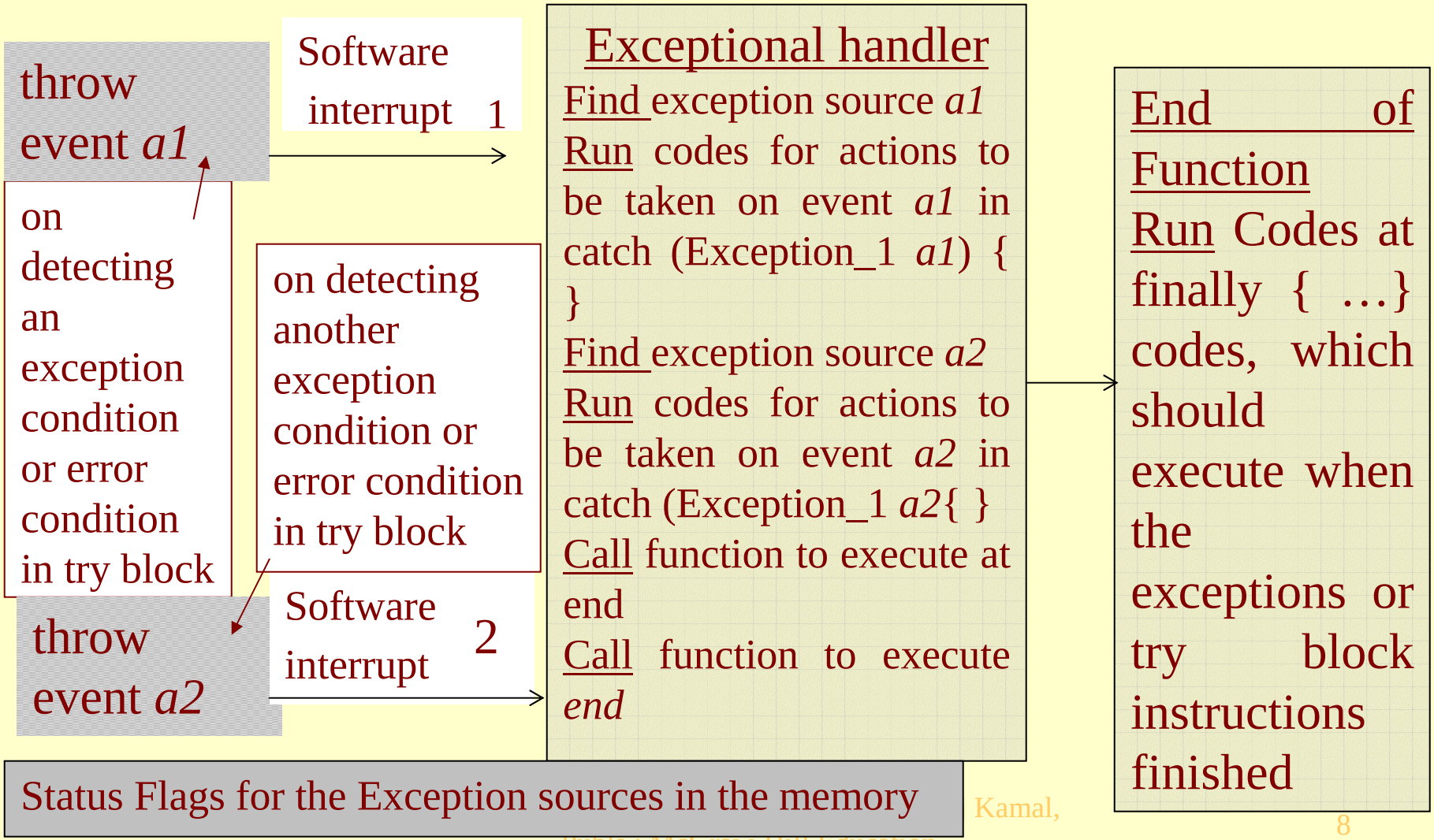
Software instruction SWI *a1*

- Causes processor interrupt.
- In response, the software ISR function ‘catch (Exception_1 a1) { }’ executes on throwing of the exception *a1* during try block execution.
- SWI *a1* is used after catching the exception *a1* whenever it is thrown

Software instruction SWI *a2*

- Causes processor interrupt.
- In response, the software ISR function ‘catch (Exception_2 a2) { }’ executes on throwing of the exception *a1* during try block execution.
- SWI *a2* is used after catching the exception *a2* whenever it is thrown

Use of SWI software interrupt-instruction for calling an ISR in the software on throwing and catching the exceptional run-time conditions a1 and a2 encountered during computations



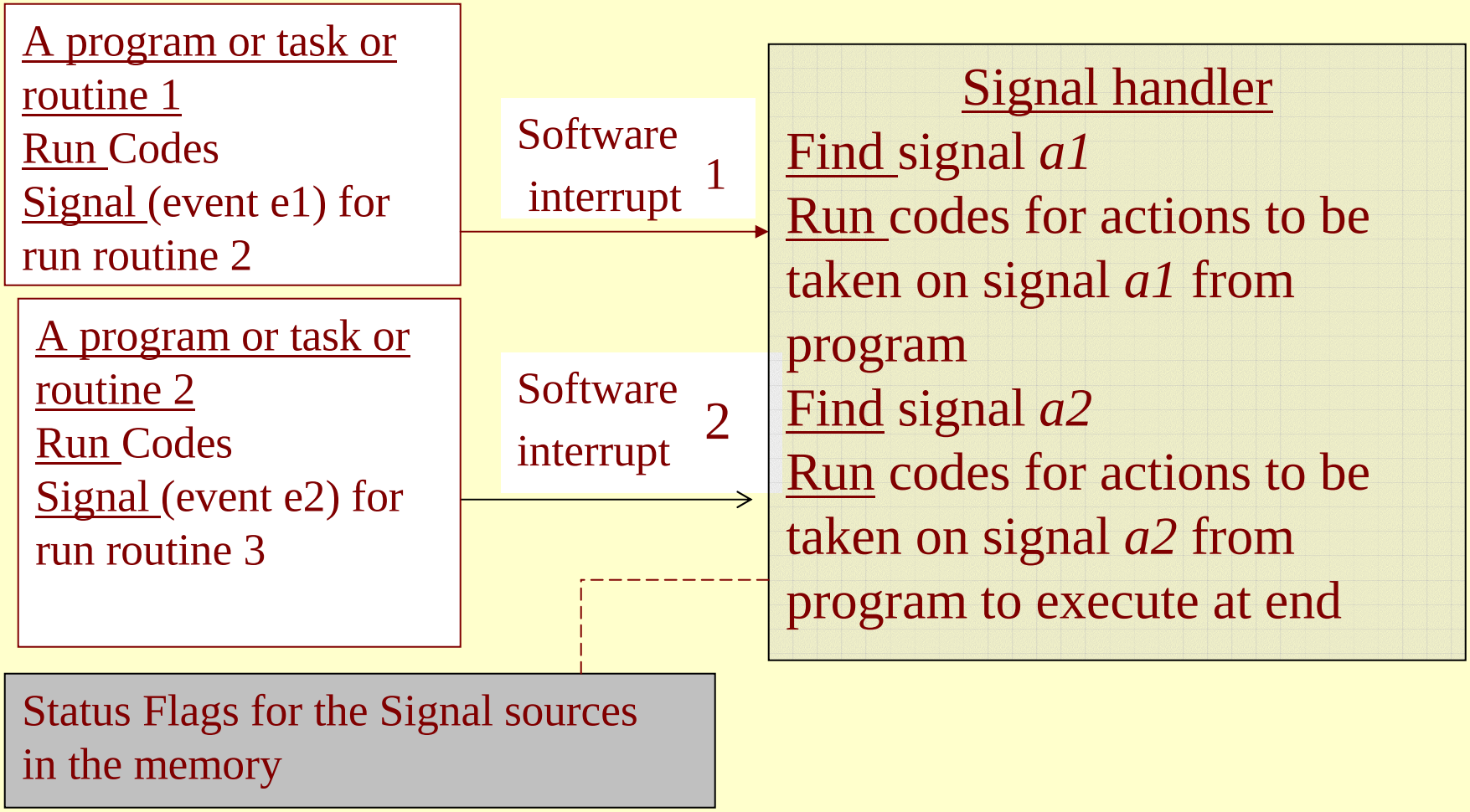
SWI *a3*

- Software ISR function ‘finally { }’ executes either at the end of the try or at the end of catch function codes.
- SWI *a3* is used after the try and catch functions finish, then *finally* function will perform final task, for example, exit from the program or call another function

Signal from a thread for Signal handler Interrupt Service Routine

- ISR is also called signal handler in case of a routine or program thread or task sends a signal using an SWI
- Signals are used to notify error conditions or notifying end of an action to enable signal handler thread or task to initiate action on that.

Action on Signal generated by SWI and signal handling



Summary

We learnt

- SWIs are used on catching on throwing an exception in a program and SWI handler (ISR) executes
- Another SWI executes at the final stage
- SWI is also used for signal from the program thread or task

End of Lesson 3 of Chapter 4