PROGRAMMING CONCEPTS AND EMBEDDED PROGRAMMING IN C, C++ and JAVA:

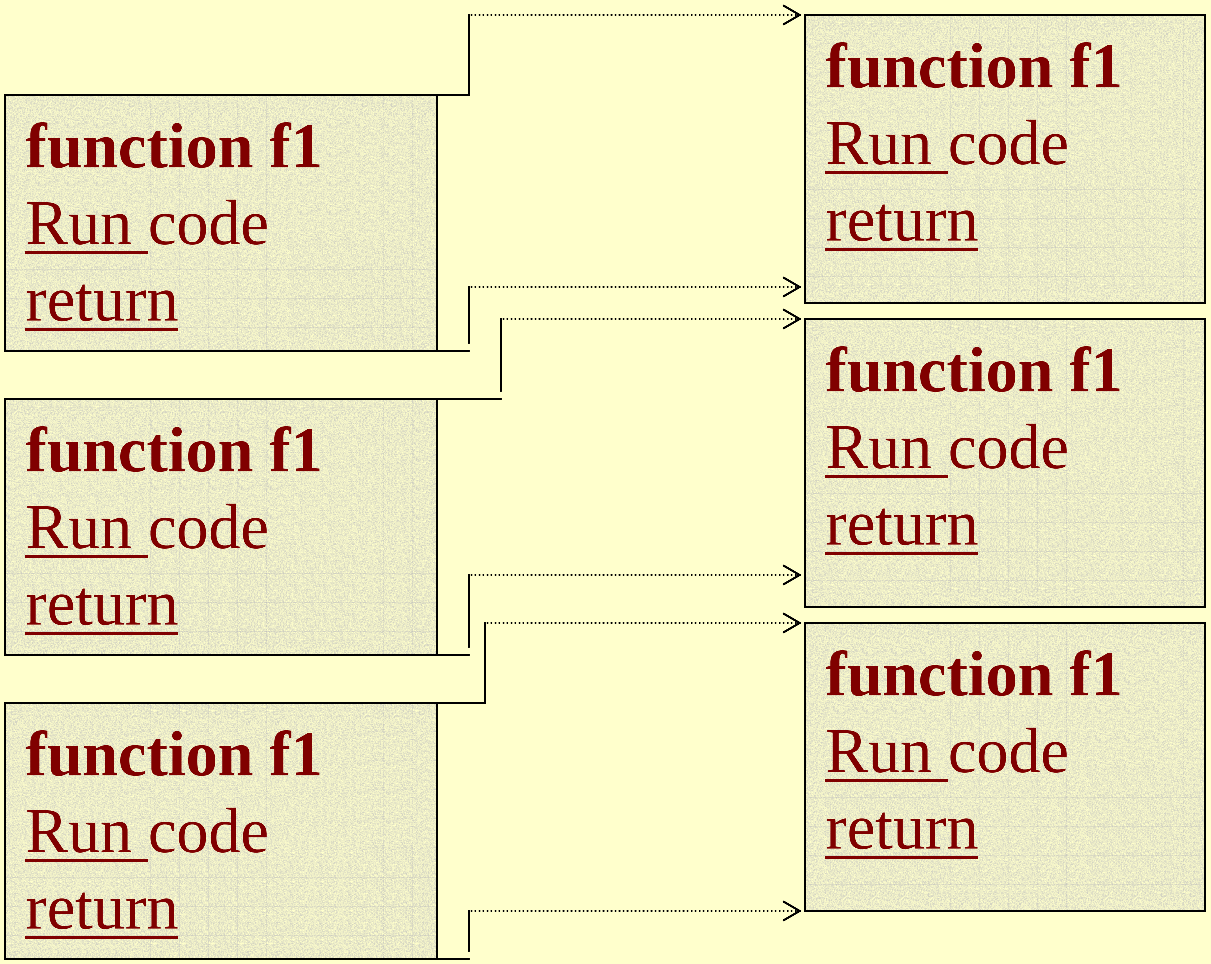
Lesson-8: Programming using functions and function queues

Programming using functions and function queues

- Use of multiple function calls in the main ( )
- Use of multiple function calls in cyclic order
- use of pointer to a function
- Use of function queues and
- Use of the queues of the function pointers built by the ISRs. It reduces significantly the ISR latency periods. Each device ISR is therefore able to execute within its stipulated deadline

1. Multiple function calls

Programming model of multiple function calls

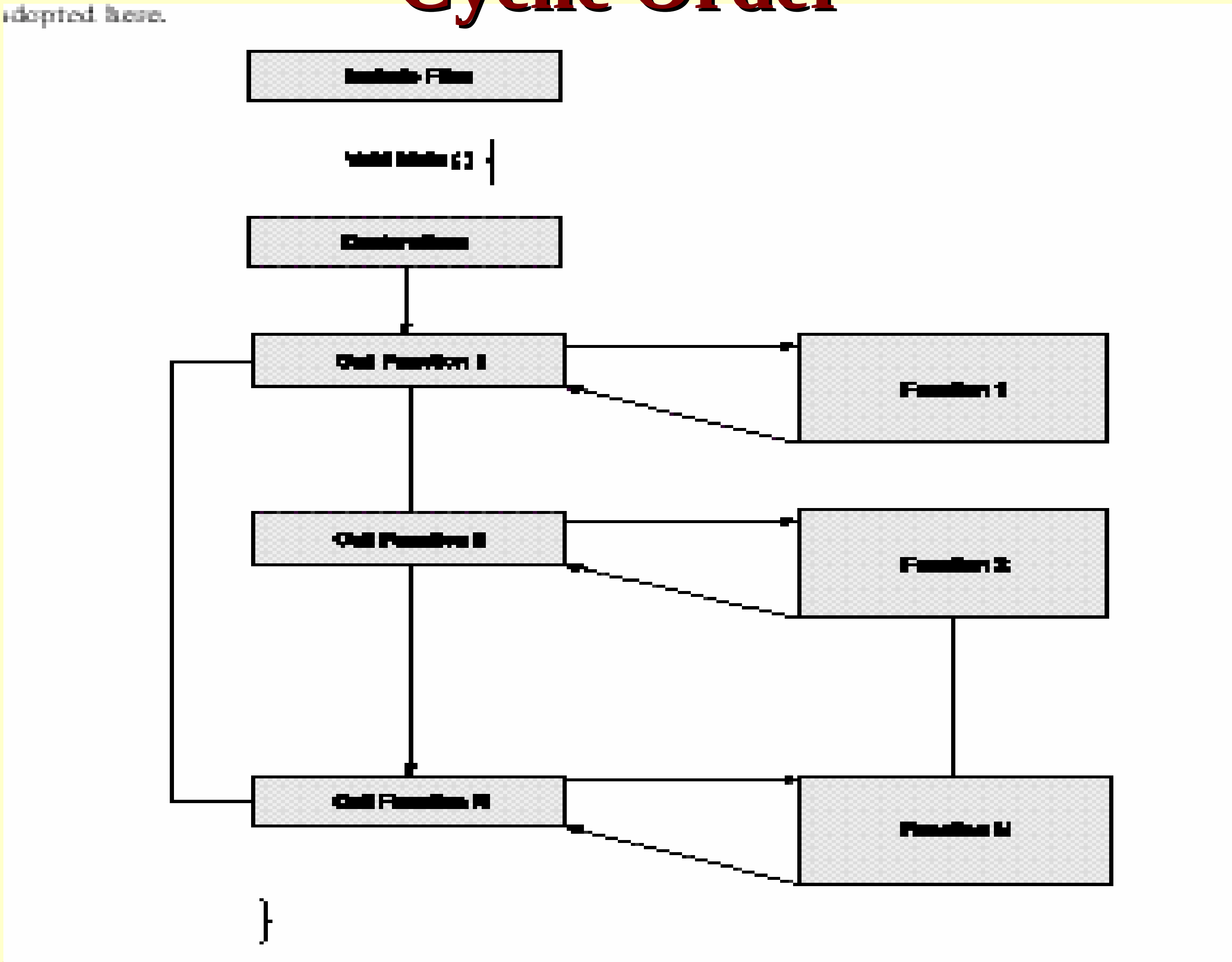


2. Multiple function calls in cyclic order

Use of **Multiple function calls in Cyclic Order**

- One of the most common methods is the use of multiple function-calls in a cyclic order in an infinite loop of the *main* ( ).

Use of Multiple function calls in Cyclic Order



3. Use of function pointers

Use of Function Pointers

* sign when placed before the function name then it refers to all the compiled form of the statements in the memory that are specified within the curly braces when declaring the function.

Use of Function Pointers

- A returning data type specification (for example, void) followed by '*(*functionName) (functionArguments)*' calls the statements of the *functionName* using the *functionArguments*, and on a return, it returns the specified data object. We can thus use the function pointer for invoking a call to the function.

4. Queue of Function-pointers

Application of Queue of Function pointers inserted by ISRs

- Makes possible the designing of ISRs with short codes and by running the functions of the ISRs at later stages

Multiple ISRs insertion of Function pointers into a Queue

- The ISRs insert the function pointers
- The pointed functions in the queue execute at later stages by deleting from the queue
- These queued functions execute after the service to all pending ISRs finishes

Example

Interrupt Service Routine ISR_PortAInputI declaration for the functions

- void interrupt ISR_PortAInputI (QueueElArray In_A_Out_B) {
- disable_PortA_Intr ( ); /* Disable another interrupt from port A*/
- void inPortA (unsigned char *portAdata); /* Function for retransmit output to Port B*/
- void decipherPortAData (unsigned char *portAdata); /* Function for deciphering */

Interrupt Service Routine ISR_PortAInputI declaration for the functions

- void encryptPortAData (unsigned char *portAdata); /* Function for encrypting */
- void outPort B (unsigned char *portAdata); /* Function for Sending Output to Port B*/

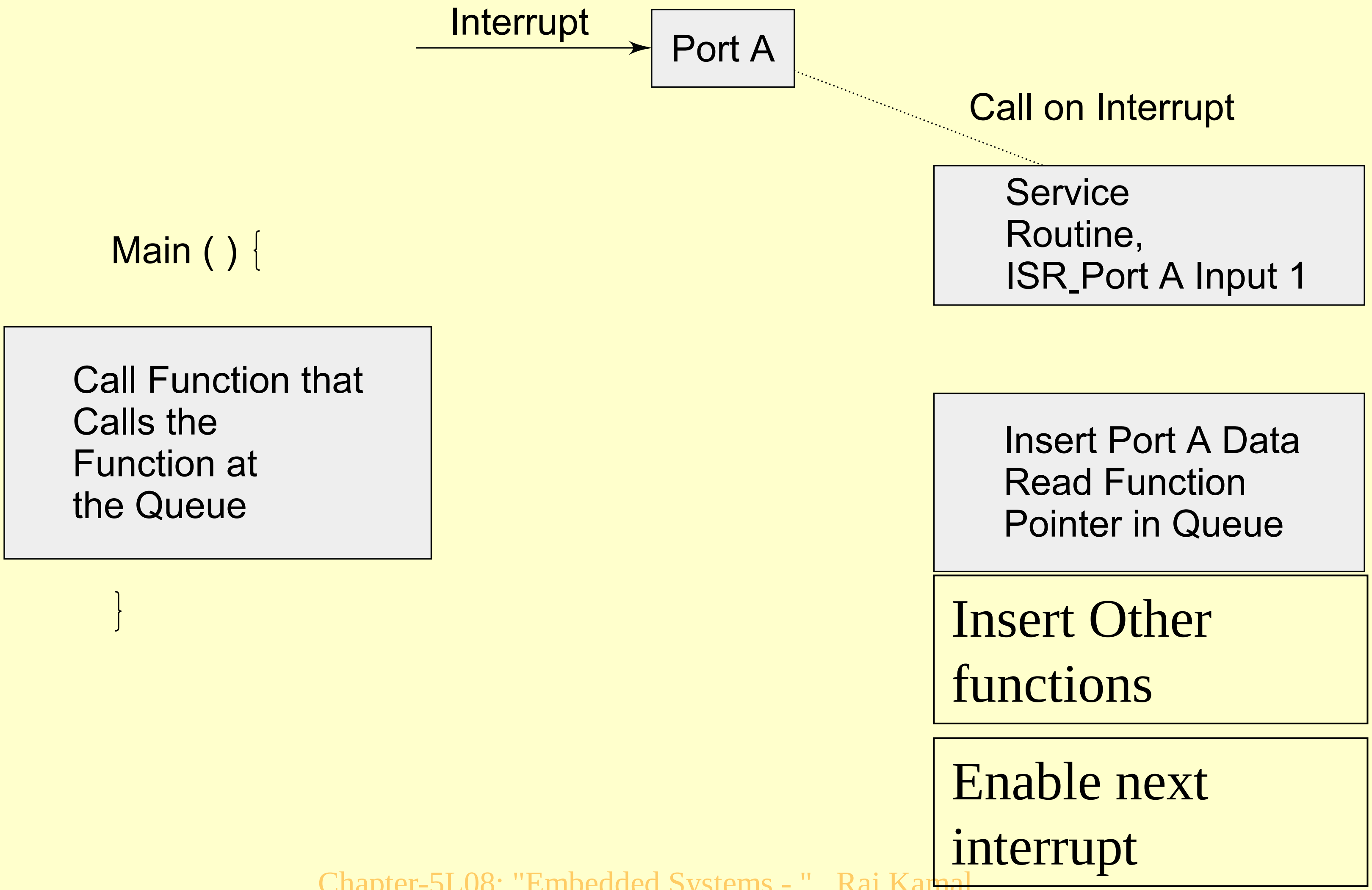
Interrupt Service Routine ISR_PortAInputI inserting function pointers

- /* Insert the function pointers into the queue */
- In_A_Out_B.QElinsert (const inPortA & *portAdata);
- In_A_Out_B.QElinsert (const decipherPortAData & *portAdata);
- In_A_Out_B.QElinsert (const encryptPortAData & *portAdata);
- In_A_Out_B.QElinsert (const outPort B & *portAdata);

Enable Function before return from ISR

- enable_PortA_Intr ( ); /* Enable another interrupt from port A*/
- }
- /*********************/

Function



Interrupt

Port A

Call on Interrupt

Service
Routine,
ISR_Port A Input 1

Insert Port A Data
Read Function
Pointer in Queue

Insert Other
functions

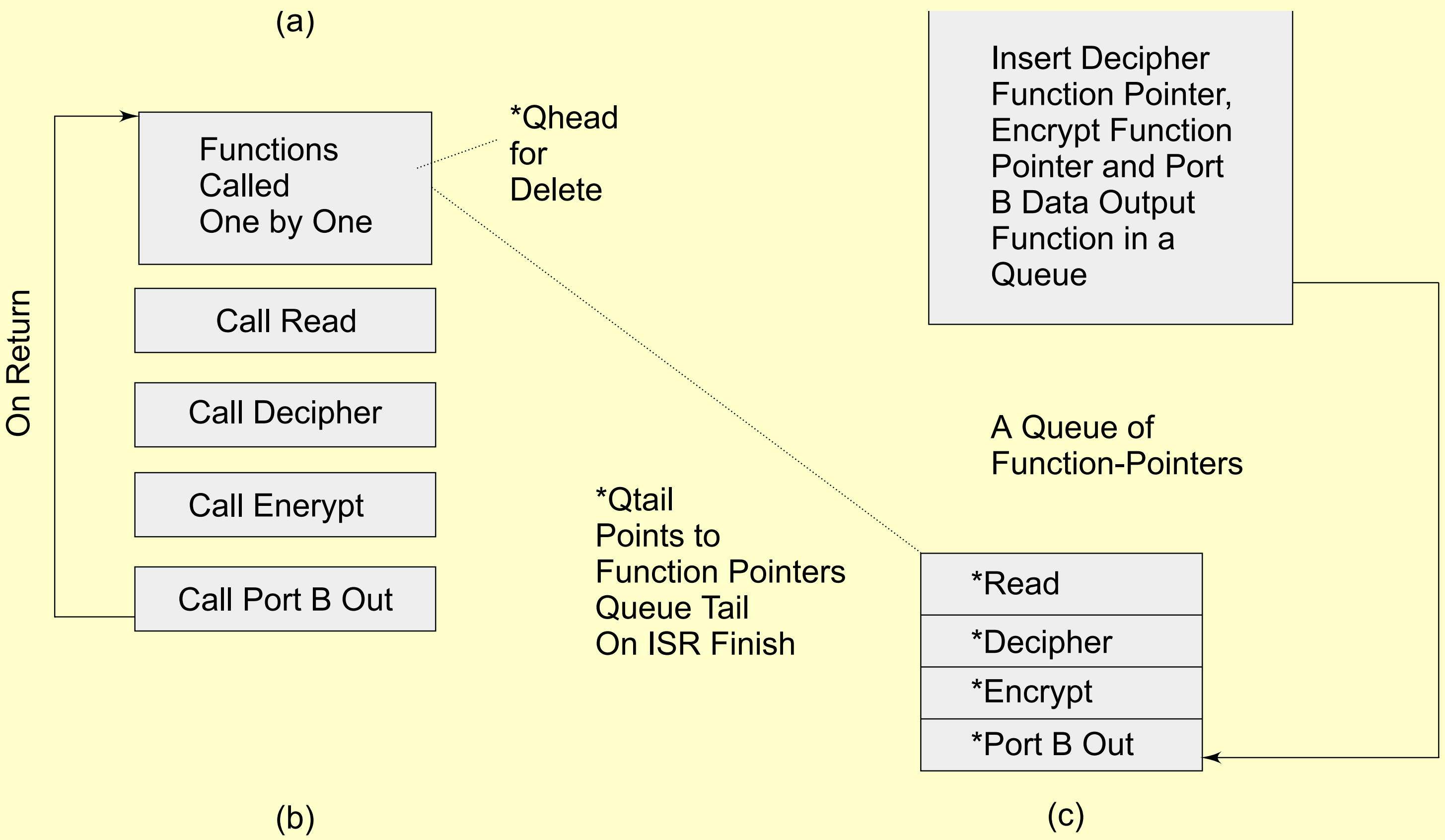
Enable next
interrupt

Main ( ) {

Call Function that
Calls the
Function at
the Queue

}

Example of Use of Function Queue created by inserting functions


(a)
Insert Decipher Function Pointer, Encrypt Function Pointer and Port B Data Output Function in a Queue
*Qhead for Delete
Functions Called One by One
On Return
Call Read
Call Decipher
Call Enerypt
Call Port B Out
A Queue of Function-Pointers
*Qtail Points to Function Pointers Queue Tail On ISR Finish
*Read
*Decipher
*Encrypt
*Port B Out
(b)
(c)

Priority Function Queue of Multiple ISRs

- When there are multiple ISRs, a high priority interrupt service routine is executed first and the lowest priority.
- The ISRs insert the function pointers into a priority queue of function pointers

[ISR can now be designed short enough so that other source don’t miss a deadline for service]

Summary

We learnt

- Three programming models
- Use of multiple function calls in the main ( )
- Use of multiple function calls in cyclic order
- use of pointer to a function
- Use of function queues and
- Use of the queues of the function pointers built by the ISRs. It reduces significantly the ISR latency periods. Each device ISR is therefore able to execute within its stipulated deadline

End of Lesson 8 of Chapter 5