Real Time Group

Real Time & Embedded Linux Solutions

Course: C for Embedded
Duration: 40 Hours
Hands-On-Training

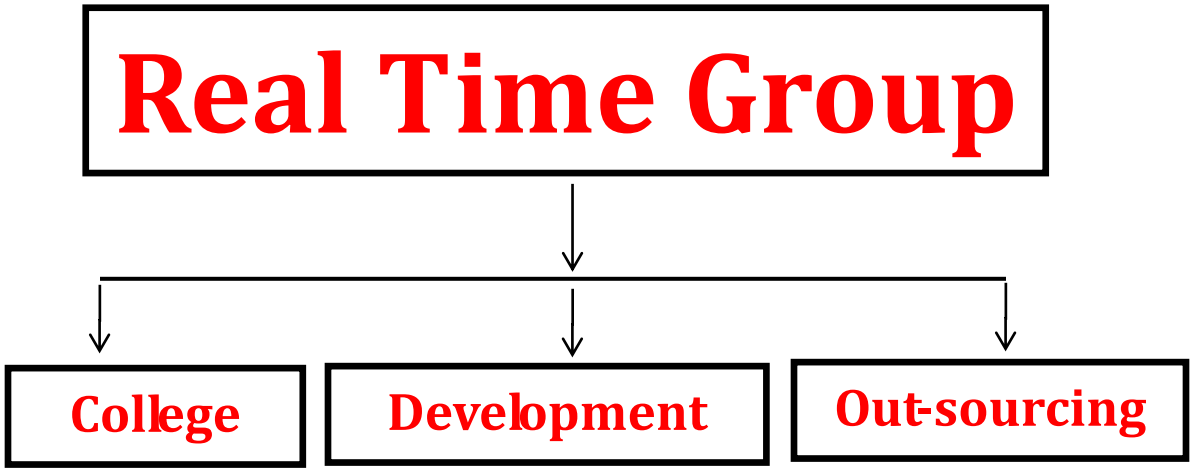
Real Time Group is a multi-disciplinary dynamic and innovative Real-Time O.S. and Embedded Software Solutions Center, established in 2007.

רח' רוזנסקי 14 ראשון לציון טל.077-7067057 / 050-3309318 פקס 077-5067058

www.rt-hr.co.il www.rt-ed.co.il www.rt-dev.com

Providing Bare-Metal and Embedded Linux solutions, professional services and consulting, end-to-end flexible system infrastructure, outsourcing, integration and training services for Hardware, Software and RT-OS \ Embedded Systems.

The company is divided into the following three Divisions:



rt-ed.co.il rt-dev.com rt-hr.co.il

Training Division:

Professional Training Services for Hardware, Software, RT-OS and Embedded systems industries.

We provide the knowledge and experience needed to enable professional engineers to Develop, Integrate and QA Hardware, Software and Networking Projects.

In order to insure experience, all courses are practical – hands-on-training.
The latest Development, QA and Automation equipment which are adopted by the industry are used.

All students are supplied with Development-Boards for home-work and course projects.

Course Overview:

This course will teach you to program the C language from the ground up.

You will learn everything from the very fundamentals of programming right through to the complexities of pointers, File IO , Structure , Linked list and more...

C is one of the most important of all programming languages.

It is used to program desktop applications, compilers, tools utilities and hardware devices.

Because The C language mostly used for low level programming , embedded linux and kernel programming , This course will emphasize on C language from an embedded point of view, How to increase performance, efficiency and flexibility when using C.

in this course we will have an introduction to C and move on to the most Complex elements and tools in the C language .

Who should attend:

- Anyone that wish to learn C programming language.
- Hardware/Electrical/software engineers who need to use C for Embedded systems.

Prerequisite:

- English language – must
- Knowledge in operation systems - advantage
- Background in software implement - advantage

C for Embedded:

1. **Introduction to Embedded Concepts:**
    a. How Embedded System Works
    b. Where is the Program Locate in Memory?
    c. The steps to run a Program
    d. How the Program Represent in RAM
    e. First C Program - "Hello World"
    f. The main () function.
    g. Basic I/O Commands.

2. **Compiling and Linking:**
    a. What is the "Compiler"
    b. How it is work
    c. The Pre-Processor
    d. The Binary file

3. **Variables and Constant:**
    a. Declaring a variable.
    b. Variables Names
    c. Variables Types
    d. How Variables store in Memory
    e. Sizeof () Usage
    f. Signed VS Unsigned
    g. Typedef Usage & Declaration
    h. Casting
    i. I/O commands – printf () & scanf ()
    j. Printf () Format specifier
    k. Constants & Enumerate

4. **Expressions**:
    a. What is an Expression in C?
    b. Operators
    c. Assignment Operators
    d. Mathematical Operators
    e. Integer Division and Modulus

f. Increments and Decrement
g. Prefix and Suffix
h. Logical Operators

5. **Statements**:
    a. The if Statement
    b. The else Keyword
    c. Nested if Statement
    d. Else if Statement
    e. Ternary operators
    f. The switch Statement

6. **Loops**:
    a. Types of Loops
    b. While Loops - basic
    c. Complicated While Loops
    d. The break Statement
    e. The continue Statement
    f. The do…. while Statement
    g. The for Loop
    h. Advance for Loop
    i. "for" VS "while"
    j. Nested for Loops

7. **Functions and Headers:**
    a. What is a Function in C?
    b. Declaring and Defining a Function
    c. Modular usage of functions
    d. Variable Scope
    e. Local Variable
    f. Global Variable
    g. Recursion

8. **Arrays**:
    a. What is an Array
    b. How to Declare an Array in C?

c. How to access to Array Elements
d. Array Definition in Memory
e. Initialize Arrays
f. Multidimensions Arrays

9. Pointers:

a. What is a pointer?
b. The Indirection Operator
c. Using pointers
d. By-ref parameters
e. Invalid pointers
f. Pointers to Functions – Declaring and Usage

10. Strings:

a. How "string" Define in C?
b. String location Memory
c. Strings Initialization
d. Char [] VS char*
e. Useful <string.h> routines
f. Working String on bytes resolutions

11. Dynamic Memory Allocation

a. The Free Store (Heap)
b. Malloc () – Definitions and Usage
c. Realloc () - Definitions and Usage
d. Free (void *ptr)
e. Pointers calculations
f. Memory Leaks
g. Malloc/free Warning
h. Dynamic Arrays on the Free Store
i. Stray or Dangling Pointers

12. Argument to main (): (Base on Time constraints)

a. argc & argv – What are they?
b. Usage of args to main ()

13. Structure in C:

a. What is a Structure in C?
b. the struct keyword
c. Accessing Structure Members Directly
d. Accessing Structure Members Indirectly
e. Dynamic Memory Allocation of Structures

14.Bitwise Operations:

a. Bitwise Operations – Usage
b. bitwise Operators
c. bitwise shift Operator
d. Changing Specific Bits
e. Read Modify Write

15.Big & Little Endian: (Base on Time constraints)

a. Big & Little Endian – Usage
b. Endianness in Networking

16.Bit fields & Unions: (Base on Time constraints)

a. What is bit-fields in C?
b. bit-field Declaration
c. How to Access to bit-field Members.
d. Bit fields – Example in Embedded System
e. Bit fields – Note of Caution
f. What is a Unions in c?
g. Union Definition
h. Unions Usage in embedded system.

17.I/O commands:

a. Background to I/O
b. FILE structure
c. FILE API
d. Standard Stream in Unix
e. Getchar () VS getc ()
f. Putchar () VS putc ()
g. Scanf () – Related Function

h. printf () – Related Function

18.The extern Keyword: (Base on Time constraints)

a. The extern Keyword - Usage
b. The extern Keyword - Declaration
c. extern – compiling and linking
d. extern – Behind the scene

19.Static Variable: (Base on Time constraints)

a. Static Variable – What is it?
b. Static Variable – Declaration and Usage.
c. register Variable
d. Volatile Variable

20. Linked List:

a. Singly Linked List
b. add () & remove () examples
c. Link list VS Array
d. Double Linked List
e. Using Stacks & Queue with Linked list
f. Binary Trees in c
g. Advance Tree Concept

21.Hash Tables: (Base on Time constraints)

a) Hash Tables – Usage
b) Hash Tables – implementation
c) Hash Function Example

22.Sorting Methods: (Base on Time constraints)

e. Bubble Sort
f. Insertion Sort
g. Selection Sort
h. Shell Sort
i. Merge Sort
j. Binary Tree Sort
k. Heap Sort

l. Quick Sort