**What is an Embedded C Program & Its Structure for Beginners?**

Earlier, many embedded applications were developed using assembly level programming. However, they did not provide portability. This disadvantage was overcome by the advent of various high-level languages like C, Pascal, and COBOL. However, it was the C language that got extensive acceptance for embedded systems, and it continues to do so. The C code written is more reliable, scalable, and portable; and in fact, much easier to understand. Embedded C Programming is the soul of the processor functioning inside each and every [embedded system](https://www.elprocus.com/ieee-projects-on-embedded-systems/) we come across in our daily life, such as mobile phones, washing machines, and digital cameras. Each processor is associated with embedded software. The first and foremost thing is the embedded software that decides to function of the embedded system. Embedded C language is most frequently used to [program the microcontroller](https://www.elprocus.com/how-to-program-the-microcontroller/).

## What is C Language?

C language was developed by Dennis Ritchie in 1969. It is a collection of one or more functions, and every function is a collection of statements performing a specific task.  
C language is a middle-level language as it supports high-level applications and low-level applications. Before going into the details of embedded C programming, we should know about RAM memory organization.

The main features of the C language include the following.

* C language is software designed with different keywords, data types, variables, constants, etc.
* Embedded C is a generic term given to a programming language written in C, which is associated with a particular hardware architecture.
* Embedded C is an extension to the C language with some additional header files. These header files may change from controller to controller.
* The [microcontroller 8051](https://www.elprocus.com/8051-microcontroller-architecture-and-applications/) #include<reg51.h> is used.

## **What is an Embedded C Programming?**

In every embedded system-based project, Embedded C programming plays a key role to make the microcontroller run & perform the preferred actions. At present, we normally utilize several electronic devices like mobile phones, washing machines, security systems, refrigerators, digital cameras, etc. The controlling of these embedded devices can be done with the help of an embedded C program. For example, in a digital camera, if we press a camera button to capture a photo then the microcontroller will execute the required function to click the image as well as to store it.

Embedded C programming builds with a set of functions where every function is a set of statements that are utilized to execute some particular tasks. Both the embedded C and C languages are the same and implemented through some fundamental elements like a variable, character set, keywords, data types, declaration of variables, expressions, statements. All these elements play a key role while writing an embedded C program.

The embedded system designers must know about the hardware architecture to write programs. These programs play a prominent role in monitoring and controlling external devices. They also directly operate and use the internal architecture of the microcontroller, such as interrupt handling, timers, serial communication, and other available features.

### **Embedded System Programming**

As we discussed earlier, the designing of an embedded system can be done using Hardware & Software. For instance, in a simple embedded system, the processor is the main module that works like the heart of the system. Here a processor is nothing but a microprocessor, DSP, microcontroller, CPLD & FPGA. All these processors are programmable so that it defines the working of the device.

An Embedded system program allows the hardware to check the inputs & control outputs accordingly. In this procedure, the embedded program may have to control the internal architecture of the processor directly like Timers, Interrupt Handling, I/O Ports, serial communications interface, etc.

So embedded system programming is very important to the processor. There are different programming languages are available for embedded systems such as C, C++, assembly language, JAVA, JAVA script, visual basic, etc. So this programming language plays a key role while making an embedded system but choosing the language is very essential.

## Steps to Build an Embedded C Program

There are different steps involved in designing an embedded c program like the following.

* Comments
* Directives of Processor
* Configuration of Port
* Global variables
* Core Function/Main Function
* Declaration of Variable
* The logic of the Program

### **Comments**

In programming languages, comments are very essential to describe the program’s function. The code of the comments is non-executable but used to provide program documentation. To understand the function of the program, this will make a simple method to understand the function of the program. In embedded C, comments are available in two types namely single line and mainline comment.

In an embedded C programming language, we can place comments in our code which helps the reader to understand the code easily.

C=a+b; /\* add two variables whose value is stored in another variable C\*/

#### **Single Line Comment**

Generally, for the programming languages, single-line comments are very useful to clarify a fraction of the program. These comments begin with a double slash (//) and it can be located anywhere within the programming language. By using this, the whole line can be ignored within a program.

#### **Multi-Line Comment**

Multi-line comments begin with a single slash (/) & an asterisk (/\*) in the programming languages which explains a block of code. These types of comments can be arranged anywhere within the programming language and mainly used to ignore a whole block of code within a program.

### **Directives of Processor**

The lines included within the program code are called preprocessor directives which can be followed through a hash symbol (#). These lines are the preprocessor directives but not programmed statements.  
The code can be examined through a preprocessor before real code compilation starts & resolves these directives before generating a code through regular statements. There are several special preprocessor directives are available although two directives are extremely helpful within the programming language like the following.

#include  
#include<reg51.h>  
Sbit LED = P2^3;  
Main();  
{  
LED = 0x0ff  
Delay();  
LED=0x00;  
}  
#define  
#include<reg51.h>  
#define LED P0  
Main();  
{  
LED = 0x0ff  
Delay();  
LED=0x00;  
}

In the above program, the #include directive is generally used to comprise standard libraries like study and. h is used to allow I/O functions using the library of ‘C’. The #define directive usually used to describe the series of variables & allocates the values by executing the process within a particular instruction like macros.

### **Configuration of Port**

The microcontroller includes several ports where every port has different pins. These pins can be used for controlling the interfacing devices. The declaration of these pins can be done within a program with the help of keywords. The keywords in the embedded c program are standard as well as predefined like a bit, sbit, SFR which are used to state the bits & single pin within a program.

There are certain words that are reserved for doing specific tasks. These words are known as keywords. They are standard and predefined in the Embedded C. Keywords are always written in lowercase. These keywords must be defined before writing the main program. The main functions of the keywords include the following.

#include< >  
Sbit a = P 2^2;  
SFR 0x00 = PoRT0;  
Bit C;  
main()  
{  
……………..  
……………..  
}

#### **sbit**

This is one kind of data type, used to access a single bit within an SFR register.

The syntax for this data type is : sbit variable name = SFR bit ;

Example: sbit a=P2^1;

If we assign p2.1 as ‘a’ variable, then we can use ‘a’ instead of p2.1 anywhere in the program, which reduces the complexity of the program.

#### Bit

This type of data type is mainly used for allowing the bit addressable memory of random access memory like 20h to 2fh.

The syntax of this data type is : name of bit variable;

Example: bit c;

It is a bit series setting within a small data region that is mainly used with the help of a program to memorize something.

#### SFR

This kind of data type is used to obtain the peripheral ports of the SFR register through an additional name. So, the declaration of all the SFR registers can be done in capital letters.

The syntax of this data type is: SFR variable name = SFR address for SFR register;

Example: SFR port0 = 0×80;

If we allocate 0×80 like ‘port0’, after that we can utilize 0×80 in place of port0 wherever in the programming language to decrease the difficulty of the program.

### SFR Register

The SFR stands for Special Function Register. In 8051 microcontroller, it includes the RAM memory with 256 bytes, which is divided into two main elements: the first element of 128 bytes is mainly utilized for storing the data whereas the other element of 128 bytes is mainly utilized to SFR registers. All the peripheral devices such as timers, counters & I/O ports are stored within the SFR register & every element includes a single address.

### Global Variables

When the variable is declared before the key function is known as the global variable. This variable can be allowed on any function within the program. The global variable’s life span mainly depends on the programming until it reaches an end.

#include<reg51.h>  
Unsigned int a, c =10;  
Main()  
{  
……………  
…………..  
}

### **Core Function / Main Function**

The main function is a central part while executing any program and it begins with the main function simply. Each program utilizes simply one major function since if the program includes above one major function, next the compiler will be confused in begin the execution of the program.

#include<reg51.h>  
Main()  
{  
……………  
…………..  
}

### **Declaration of Variable**

The name like the variable is used for storing the values but this variable should be first declared before utilized within the program. The variable declaration states its name as well as a data type. Here, the data type is nothing but the representation of storage data. In embedded C programming, it uses four fundamental data types like integer, float, character for storing the data within the memory. The data type size, as well as range, can be defined depending on the compiler.

The data type refers to an extensive system for declaring variables of different types like integer, character, float, etc. The embedded C software uses four data types that are used to store data in memory.

The ‘char’ is used to store any single character; ‘int’ is used to store integer value, and ‘float’ is used to store any precision floating-point value. The size and range of different data types on a 32-bit machine are given in the following table. The size and range may vary on machines with different word sizes.

* The char/signed char data type size is 1 byte and its range is from -128 to +128
* The unsigned char data type size is 1 byte and its range is from 0 to 255
* Int/signed int data type size is 2 byte and its range is from -32768 to 32767
* Unsigned int data type size is 2 byte and its range is from 0 to 65535

Main();  
{  
Unsigned int a,b,c;  
}

The Structure of an Embedded C Program is shown below.

* comments
* preprocessor directives
* global variables
* main() function

{

* local variables
* statements
* …………..
* …………..

}

* fun(1)

{

* local variables
* statements
* …………..
* …………..

}

The logic of the Program

The logic of the program is a plan of the lane that appears in the theory behind & predictable outputs of actions of the program. It explains the statement otherwise theory regarding why the embedded program will work and shows the recognized effects of actions otherwise resources.

Main  
{  
LED = 0x0f;  
delay(100);  
LED = 0x00;  
delay(100);  
}

### Main Factors of Embedded C Program

The main factors to be considered while choosing the programming language for developing an embedded system include the following.

#### Program Size

Every programming language occupies some memory where embedded processor like microcontroller includes an extremely less amount of random access memory.

#### Speed of the Program

The programming language should be very fast, so should run as quickly as possible. The speed of embedded hardware should not be reduced because of the slow-running software.

#### Portability

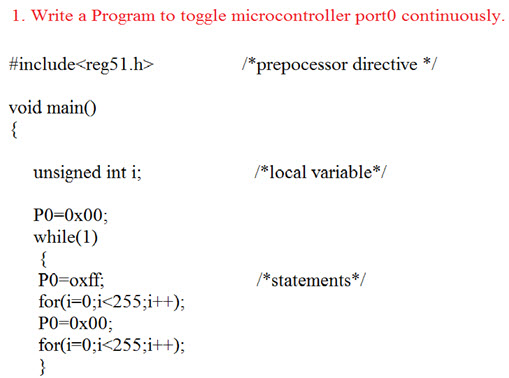
For the different embedded processors, the compilation of similar programs can be done.

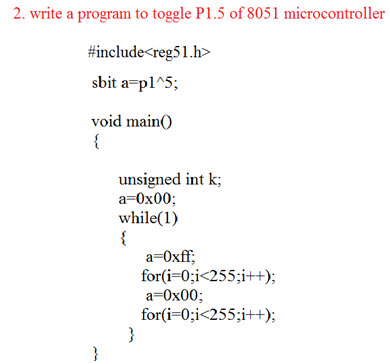
* Simple Implementation
* Simple Maintenance
* Readability

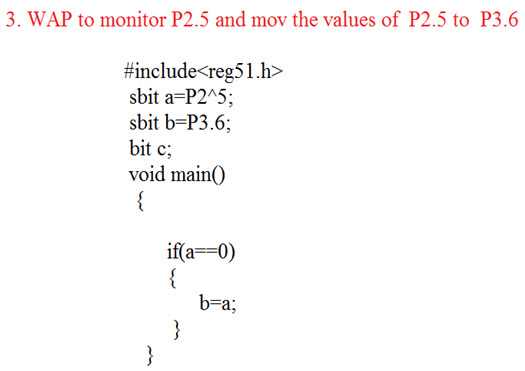
### **Differences between C Program and Embedded C Program**

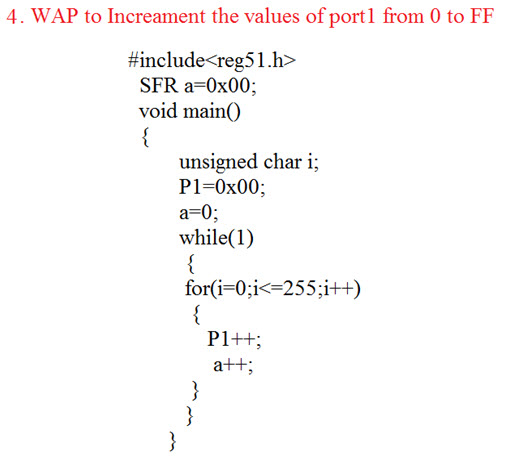
The difference between embedded C and C programming is not much actually apart from the operating environment and some extensions. These programming languages are ISO Standards and also have approximately similar syntax, functions, data types, etc. The main differences between C programming and embedded c programming include the following.

|  |  |
| --- | --- |
| **C Language** | **Embedded C Language** |
| Generally, this language is used to develop desktop-based applications | Embedded C language is used to develop microcontroller-based applications. |
| C language is not an extension to any programming language, but a general-purpose programming language | Embedded C is an extension to the C programming language including different features such as addressing I/O, fixed-point arithmetic, multiple-memory addressing, etc. |
| It processes native development in nature | It processes cross development in nature |
| It is independent for hardware architecture | It depends on the hardware architecture of the microcontroller & other devices |
| The compilers of C language depends on the operating system | Embedded C compilers are OS independent |
| In C language, the standard compilers are used for executing a program | In embedded C language, specific compilers are used. |
| The popular compilers used in this language are GCC, Borland turbo C, Intel C++, etc | The popular compilers used in this language are Keil, BiPOM Electronics & green hill |
| The format of C language is free-format | Its format mainly depends on the kind of microprocessor used. |
| Optimization of this language is normal | Optimization of this language is a high level |
| It is very easy to modify & read | It is not easy to modify & read |
| Bug fixing is easy | Bug fixing of this language is complicated |









### Advantages

The **advantages of embedded c programmin**g include the following.

* It is very simple to understand.
* It executes a similar task continually so there is no requirement for changing hardware like additional memory otherwise storage space.
* It executes simply a single task at once
* The cost of the hardware used in the embedded c is typically so much low.
* The applications of embedded are extremely appropriate in industries.
* It takes less time to develop an application program.
* It reduces the complexity of the program.
* It is easy to verify and understand.
* It is portable from one controller to another.

### Disadvantages

The **disadvantages of embedded c programming** include the following.

* At a time, it executes only one task but can’t execute the multi-tasks
* If we change the program then need to change the hardware as well
* It supports only the hardware system.
* It has a scalability issue
* It has a restriction like limited memory otherwise compatibility of the computer.

### Applications of Embedded C Program

The **applications of embedded c programming** include the following.

* Embedded C programming is used in industries for different purposes
* The programming language used in the applications is speed checker on the highway, controlling of traffic lights, controlling of street lights, tracking the vehicle, artificial intelligence, home automation, and auto intensity control.