LANGUAGES USED FOR EMBEDDED FIRMWARE DEVELOPMENT

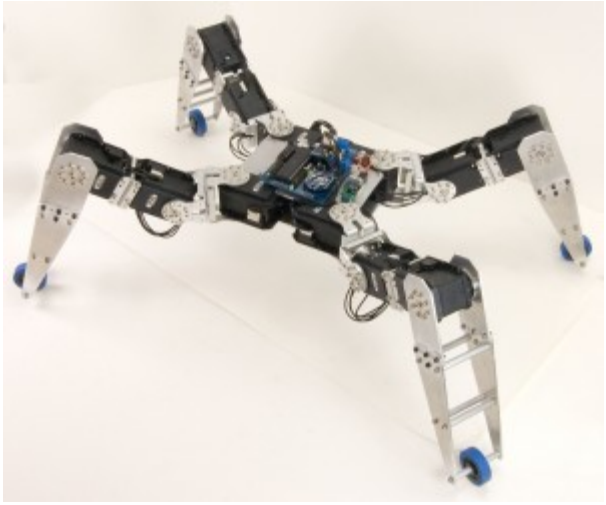
Introduction

Every digital device that is driven by a CPU core processes some sort of digital information. Not all of them are digital data. In fact, major portion of them are some kind of instruction to the CPU. As we all know, they are patterns of zeros and ones. In groups, those zeros and ones are called as nibbles (a group of 4), bytes (A group of 8), half word (group of 16) and word (group of 32). Every microcontroller comes with an inbuilt memory. They store instruction and data. When their CPU runs, they fetch the code from the memory. They obtain data either from memory or from external I/O.

The collection of the data and the code together is called **firmware.** And the algorithm that the programmer follows is called a **program.** The final output of a program is a binary bit stream that is executed by the CPU. We also call them as hex code.

If someone writes an error free program and generates hex code by himself/herself, no doubt it will run. But its laborious, time consuming and prone to error. So as humans always do, less laborious and less problematic solution of this particular problem has been found out! Computer languages are brought to work for code generation.

The basic target of any compiler is to generate hex code. Whether it is an executable file or the collection of the hexadecimal code stored in a file called “HEX File”. For embedded system, it always comes out in form of HEX files that is executed by the CPU (Note that few processors like ARM supports direct JAVA byte code execution). It is later downloaded into target microcontrollers (we call this process as “***burning program to microcontroller***”).



Embedded C

C is the immediate advance language to assembly language for software dehvelopment. Though microcontrollers provided limited processing power, embedded C developers managed to integrate most popular functions of ANSI C to embedded C. As discussed previously, sometimes the IDE developers provide some extensions. They use keywords, various macros and functions to facilitate programming for microcontrollers.

Each manufacturers may publish multiple architectures. And each architecture must have at least one compiler to support programming. Their architectural difference stops to use one single compiler for all families. Few specific difference may demand a huge programming in the compilers framework. So usually manufacturers avoid publishing a micro controller with very different architecture. The compilers of manufacturers' usually doesn't support other IC manufacturers' microcontrollers. The major reason is that it is very difficult to develop a single standard IDE due to large variety of microcontroller family. Here is a list of few IDE published by popular manufacturers.

Publisher	**IDE**
Renesas	Cube Suite Plus
Texas Instruments	IAR Workbench
National Instruments	Lab VIEW
NXP (a Phillips undertaken company)	LPC Xpresso
Microchip	MPLab SIM
Atmel	AVR Studio
ARM	Keil Microvision

JAVA

Most of the embedded IDE doesn't support java programming. Yet, many JAVA application runs successfully in many devices. Recently, few IDEs are adopting JAVA for hex code generation. But this is not the major key to the success of JAVA in the field of embedded systems. Instead of adapting JAVA as a program development language, ARM adapted a technique to allow JAVA Byte code to be executed by a single microcontroller. JAVA byte code is the usual output of a java program. This allowed software professionals to develop JAVA based software and operating system to run on ARM powered devices. This allowed to use the existing talents in the IT sectors. This, in turn, pumped up production of gadgets running with JAVA based systems. Almost 90% of the mobile and portable devices comes with firmware developed with JAVA.

The impact of adapting JAVA by a microcontroller is a milestone. But efforts are there to make HEX code generation possible wit JAVA. The pioneer of this effort is Microchip Corporation. Their IDE MP Lab Sim supports embedded JAVA to develop a program. Nevertheless, they have to generate hex code for their microcontrollers.

Source : http://www.circuitstoday.com/languages-used-for-embedded-firmware-development