Unit -I

INTRODUCTION TOEMBEDDED SYSTEMS

1.2 Software for Embedding in a System

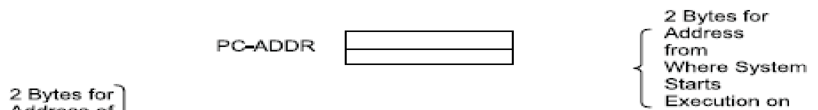
ROM image ,Programming Languages and Program models

1. ROM Image

• Final stage software also called ROM image

(Just as an image is a -+

Unique sequence and arrangement of pixels, embedded softwares also a unique placement and arrangement at each ROM address of bytes for instructions and data.)



Final machine software

- Bytes at each address defined for creating the ROM image.
- By changing this image , the same hardware platform work differently and can be used for entirely different applications or for new upgrades of the same system.
- Distinct ROM image in a distinct Embedded System

_Hardware elements between the distinct systems can be identical but it is the software that makes a system unique and distinct from the other.

- Compressed Codes and Data ROM image may alternatively be compressed software (for example, the zip format) and data (for example, the pictures in jpg or gif format) along with the software required for decompression algorithm

Programming Languages

1. Machine Language Coding Programmer defines the addresses and the corresponding bytes orbits at each address.
2. Used in configuring some specific physical device or subsystem like transceiver, the machine code-based coding is used

3. Assembly Language Coding Needed for Invoking Processor
   Specific Instructions Requires understanding of the processor and instruction set.

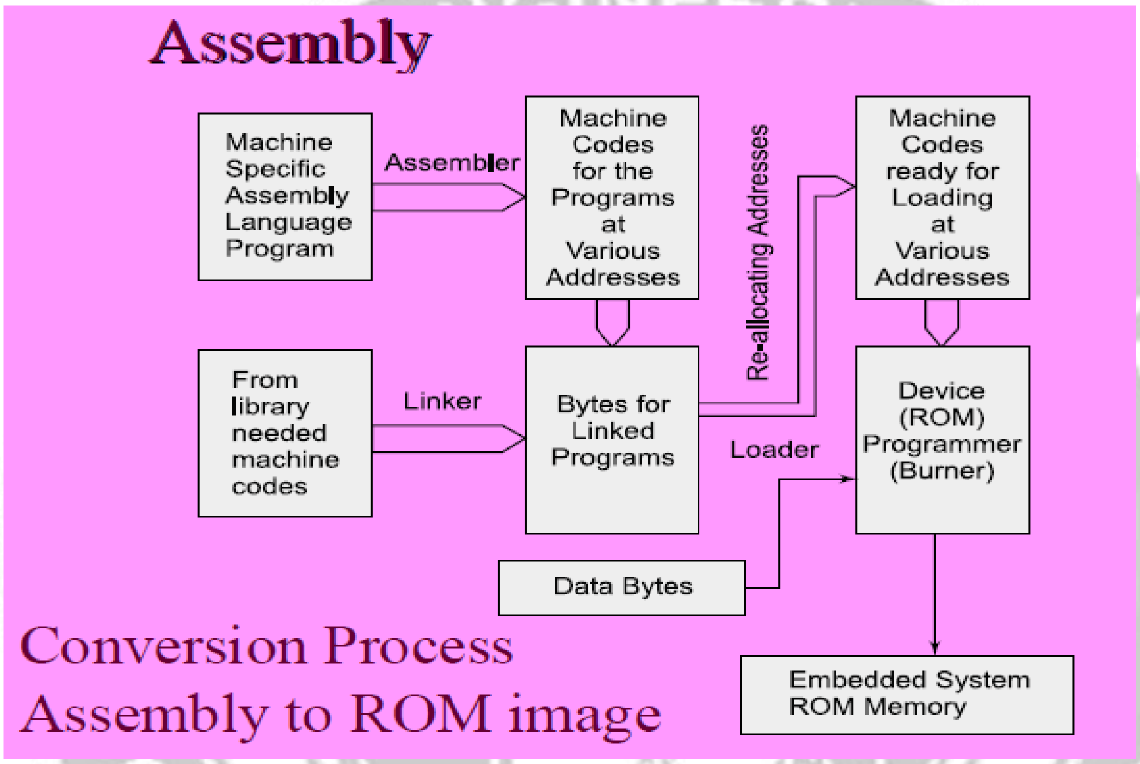
A program or a small specific part coded in the assembly language using an Assembler (

software used for developing codes in assembly).

Three steps when using assembly

language' Assembler',

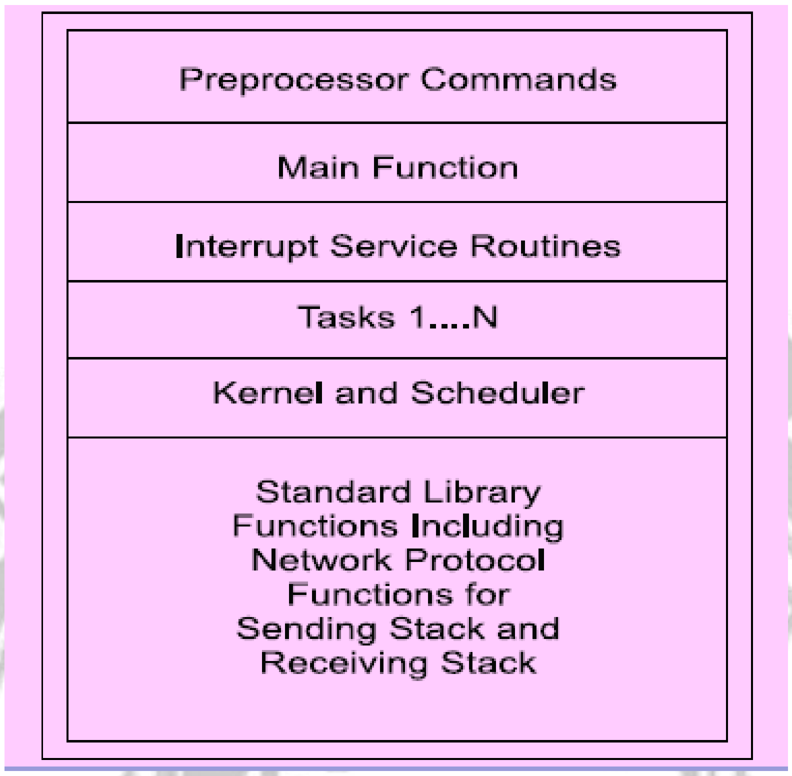
'Linker 'and


Assembly
Machine Specific Assembly Language Program
Assembler
Machine Codes for the Programs at Various Addresses
Re-allocating Addresses
Machine Codes ready for Loading at Various Addresses
From library needed machine codes
Linker
Bytes for Linked Programs
Loader
Device (ROM) Programmer (Burner)
Data Bytes
Embedded System ROM Memory
Conversion Process
Assembly to ROM image

'Locator 'before finally burned at the ROM

3. Programming language C or C++or Visual

C++or Java

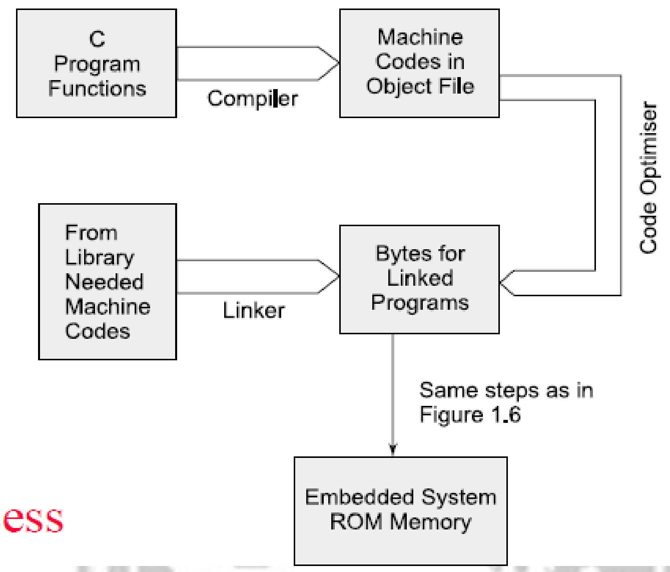

Preprocessor Commands
Main Function
Interrupt Service Routines
Tasks 1....N
Kernel and Scheduler
Standard Library Functions Including Network Protocol Functions for Sending Stack and Receiving Stack

Application Software - Different Program

Layers Program various layers–

- process or commands,
- main function,
- Task functions and
- Library functions,
- Interrupt service routines
- And kernel (scheduler),Compiler
- Generates an object file. Using linker and locator , the file for ROM image is created for the targeted hardware. C++and Java are other languages used for software coding

Converting a C program into ROM image


C Program Functions
Compiler
Machine Codes in Object File
Code Optimiser
From Library Needed Machine Codes
Linker
Bytes for Linked Programs
Same steps as in Figure 1.6
Embedded System ROM Memory

Process

Program Models

- Sequential Programming Model Object Oriented Programming Model
- Control and Data flow graphs or

Synchronous Data Flow(SDF) Graph or Multi Thread Graph (MTG)Model

- Finite State Machine for data path
- Multi threaded Model
- Concurrent Processing of processes or thread or tasks

SoftwareforembeddinginSystem-Part2

Device drivers, Device manager, OS, RTOS and

Software tools ``````Devices

- In an embedded system , there are number of *physical devices*.
- Physical devices– keypad, LCD display or touch screen, memory stick (flash memory), wireless net working device, parallel port and network card In an embedded system, there are number of *virtual devices*.

- Virtual devices–pipe, file ,RAM disk, socket,

A *device driver* is software for controlling (configuring), receiving and sending a byte or a stream of bytes from or to a device.

Asetofgenericfunctions,suchascreate(),open(),connect(),listen(),accept(),read ( ), write ( ), close ( ), delete ( ) for use by high level programmers Each generic function calls a specific software (interrupt service routine),which controls a device function or device input or output

Device controls and functions by:

1. Calling an ISR(also called Interrupt Handler Routine) on hardware or software interrupt
2. Placing appropriate bits at the control register or word.
3. Setting status flag(s) in the status register for interrupting, therefore running (driving) the ISR, Resetting the status flag after interrupt service.

Device Manager for the devices and drivers

Device Management software (usually a part of the OS) provide codes for detecting the presence of devices, for initializing (configuring) these and for testing the devices that are present.

Also includes software for allocating and registering port(s) or device codes and data at memory addresses for the various devices at distinctly different addresses, including codes for detecting any collision between the allocated addresses, if any