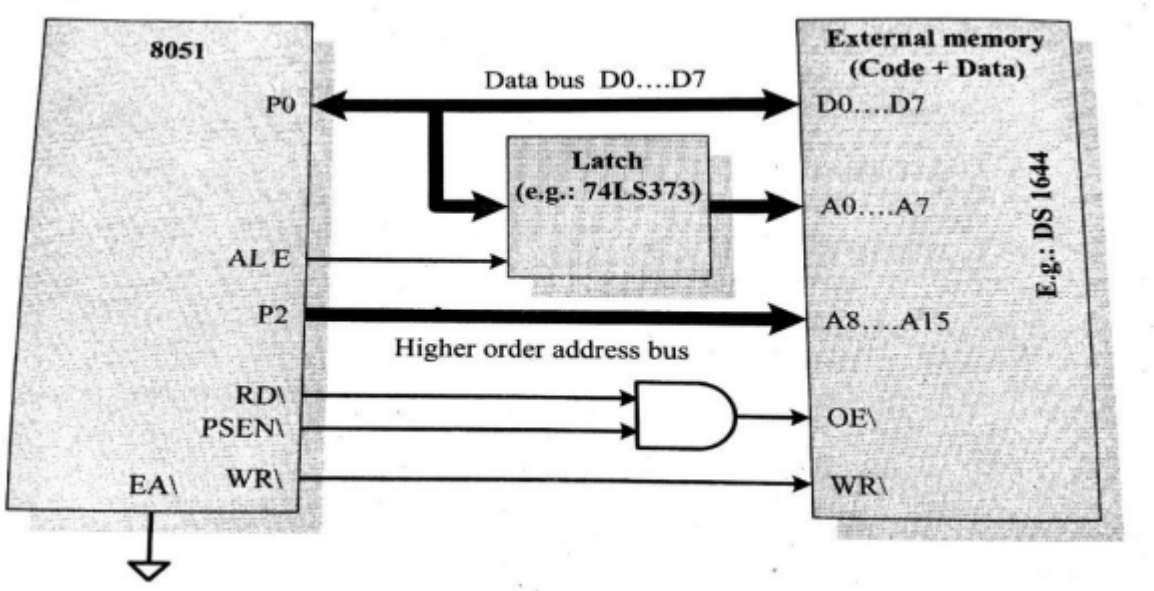
INTRODUCTION TO EMBEDDED SYSTEMS

QUESTION BANK FOR SEMESTER-II

1. **What is an Embedded System? Explain the different applications of Embedded System.**

2. **Differentiate between RISC and CISC processors.**
3. **Explain the various purposes of embedded system.**
4. **What are the characteristics of an Embedded System?**
5. **What is Read-Write Memory? Explain the categories of Read-Write Memory.**
6. **Explain Automotive Communication Buses.**
7. **What is Direct Memory Access (DMA)?**
8. **Explain the working of embedded system with respect to Washing Machine.**
9. **Differentiate between Microprocessor and Microcontroller.**
10. **Write 8051 C program to generate delay using timer register.**
11. **Explain the following:**
    a. RXD. b. TXD. c. $\overline{\text{INT}}$0 and $\overline{\text{INT}}$1. d. T0 and T1. e. $\overline{\text{PSEN}}$
12. **Write 8051 C program to convert packed BCD 0x29 to ASCII and display the bytes on P1 and P2.**
13. **Explain the Von-Neumann Memory Model for 8051.**



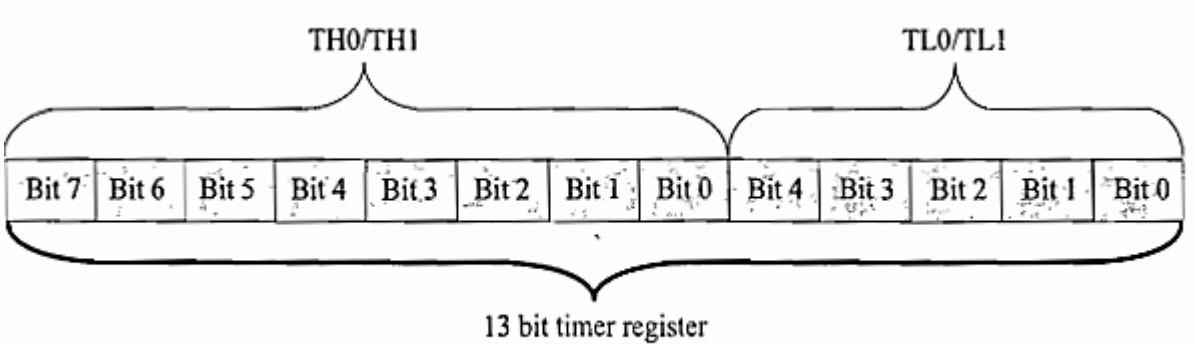
- The code memory and data memory of 8051 can be combined together to get benefits of Von-Neumann model.
- A single memory chip with read/write option can be used for this purpose.
- The program memory can be allocated to the lower memory space starting from 0000H and data memory cane be assigned to some other specific area after the code memory.

- For program memory fetching and data memory read operations combine the PSEN\ and RD\ signals using AND gate and connect it to the Output enable(OE\) signal of the memory chip.
- Drawback:
    - Accidental corruption of program memory.
    - Reduction in total memory space. In separate program and data memory model the total available memory is 128KB(64KB program memory + 64KB data memory) whereas in combine only 64KB memory is available.

14. What are the factors to be considered in selecting a controller?

- Speed of Operation :It is one of the most important factor that needs to be considered while selecting microcontroller. The number of clocks required per instruction cycle and maximum operating clock frequency supported by processor greatly affects the speed of operation of the controller.
- Code Memory Space: The target processor/controller application is written in C or any other high level language. Does the controller support sufficient code memory space to hold the compiled hex code?
- Data Memory Space: Does the controller support sufficient internal data memory(ON chip RAM) to hold run time variables and data structures?
- Development Support: In this it is essential to check that the controller manufacture provides cost effective development tools. It also needs to be checked that manufacturer provide sample product for prototyping and sample development stuffs to alleviate the development pains. Does the controller support third party development tools? Does the manufacturer provide technical support if necessary?
- Availability: This is another important factor that should be considered for selecting the process. Since the product is entirely dependent on the controller, the product development time and time to market the product solely depends on its availability. Technically it is referred as Lead Time. Lead time is the time elapsed between the purchase order approval and the supply of the product.
- Power Consumption: Power consumption of the controller should be minimum. It is critical factor since high power requirement leads to bulky power supply design. The high power dissipation also demands for cooling fans and it will make the overall system messy and expensive. Controller should support idle and power down modes of operation to reduce power consumption.
- Cost: It is the biggest deciding factor in selecting a controller. The cost should be within the reachable limit of the end user and the targeted user should not be high tech.

15.Explain Timer/Counter in Mode 0.



26 **Timer (Counter) Register for Mode 0**

- Timer/Counter-0 and Timer/Counter-1 in mode 0 acts as a 13bit timer/counter (Fig. 5.26).
- The 13bit register is formed by all 8 bits of TH0 and the lower 5 bits of TL0 for Timer/Counter-0 (All 8 bits of TH1 and the lower 5 bits of TL1 for Timer/Counter-l).
- The timer/counter mode selection is done by the bit C/T of the register TMOD.
- Timer/Counter-0 & Timer/Counter-1 has separate selection bits.
- If Timer/Counter-0 is configured as timer and if the corresponding run control bit for Timer/Counter-0 (TR0 in Timer Control Register (TCON)) is set, registers TL0 & TH0 functions as a 13bit register and starts incrementing from its current value.
- The timer register is incremented by one on each machine cycle.
- When the 13bit Timer register rolls over from all 1s to all 0s (FFFFH to (0000H), the corresponding timer overflow flag TF0, present in TCON register is set.
- If the Timer 0 interrupt is in the enabled state and no other conditions block the Timer 0 interrupt, Timer 0 interrupt is generated and is vectored to its corresponding vector address 000BH.
- On vectoring the interrupt, the TFO is automatically cleared.
- Operation of Timer 1 in mode 0 is same.

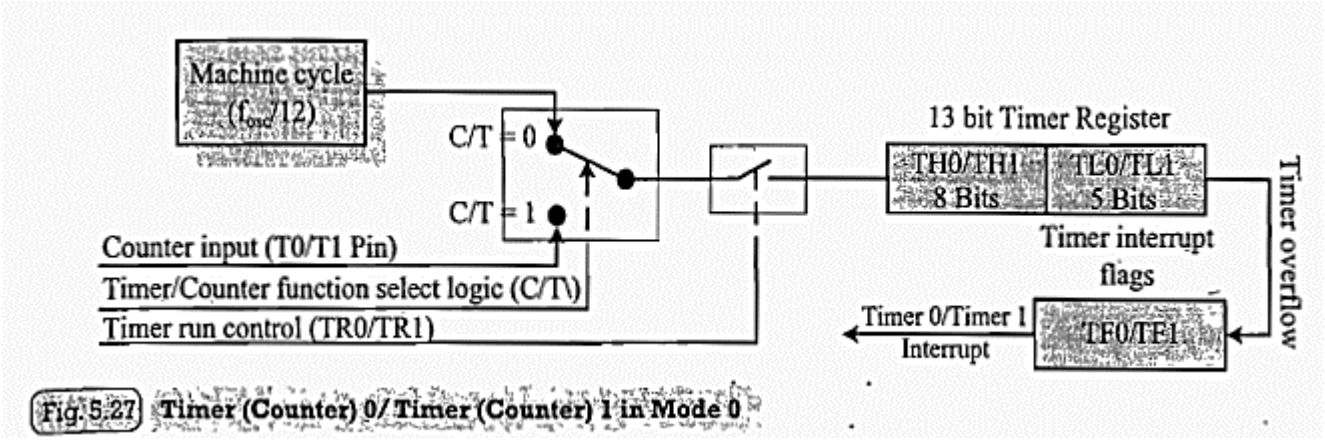
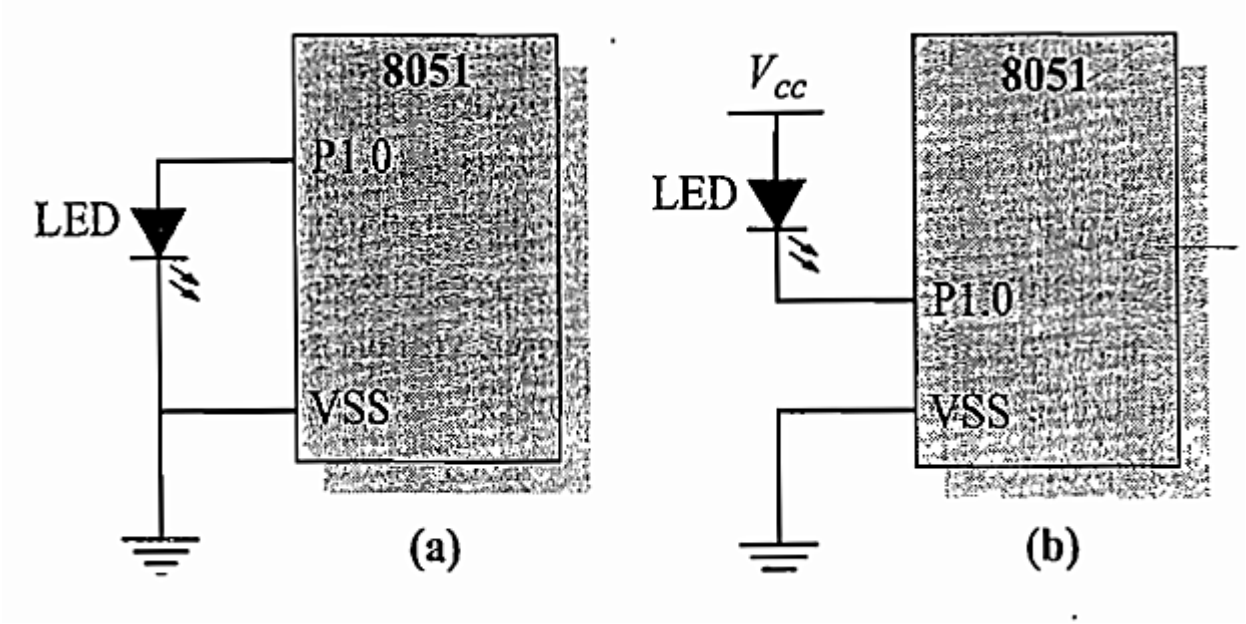


Fig. 5.27 **Timer (Counter) 0/ Timer (Counter) 1 in Mode 0**

16. What is Sink current and Source current?



(a) Current Sourcing, (b) Current Sinking

- Source Current: The term source current refers to how much current the 8051 port pin can supply to drive an externally connected device. The device can be an LED, a buzzer or a TTL logic device. For TTL family of 8051 devices the source current is defined in terns of TTL logic. TTL logic has two logic levels namely logic I (High) and logic 0 (Low).
- Sink Current: It refers to the maximum current that the 8051 port pin can absorb through a device which is connected to an external supply. The device can be an LED, a buzzer or a TTL logic device.

17. Explain the Types of Operating Systems.

- General Purpose Operating System (GPOS): The operating systems, which are deployed in general computing systems, are referred as General Purpose Operating Systems (GPOS). The kernel of such an OS is more generalised and it contains all kinds of services required for executing generic applications. General-purpose operating systems are often quite nondeterministic in behaviour. Their services can inject random delays into application software and may cause slow responsiveness of an application at unexpected times. GPOS are usually deployed in computing systems where deterministic behaviour is not an important criterion. Personal Computer/Desktop system is a typical example for a system where GPOSs are deployed. Windows etc. are examples for General Purpose Operating Systems.
- Real-Time Operating System (RTOS): 'Real-time' implies deterministic timing behaviour. Deterministic timing behaviour in RTOS context means the OS services consumes only known and expected amounts of time regardless the number of services. A Real-Tine System or RTOS implements policies and rules concerning time-critical of a system's resources. The RTOS decides which applications should run

in which order and how much time needs to be allocated for each application. Predictable performance is the hallmark of a well-designed RTOS. This is best achieved by the consistent application of policies and rules. Policies guide the design of an RTOS. Rules implement those policies and resolve policy conflicts. Windows CE, QNX, VxWorks MicroC/OS-II, etc. are examples of Real-Time Operating Systems (RTOS).

18. What are the objectives of EDLC(Embedded Product Development Life Cycle)?

- Ensuring High Quality for Products:The primary definition of quality in any embedded product development is the Return on Investment (ROI) achieved by the product. The expenses incurred for developing the product may fall in any of the categories; initial investment, developer recruiting, training, or any other infrastructure requirement related. There will be some budgetary and cost allocation given to the development of the product and it will be allocated by some higher officials based on the assumption that the product will produce a good return or a return justifying the investment. The budget allocation might have done after studying the market trends and requirements of the product, competition, etc. EDLC must ensure that the development of the product has taken account of all the qualitative attributes of the embedded system.
- Risk Minimisation and Defect Prevention through Management: There are projects in embedded product development which requires or 'tight' project management. If the product development project is a simple one, a senior developer itself can take charge of the management activity and no need for a skilled project manager to look after this with dedicated effort throughout the development process, but there should be an overall supervision from a skilled project management team for ensuring that the development process is going in the right direction. Projects which are complex and requires timeliness should have a dedicated and skilled project management part and hence they are
  said to be "tightly" bounded to project management. 'Project management is essential for predictability, co-ordination and risk minimisation'. Whenever a product development request comes, an estimate on the duration of the development and deployment activity should be given to the end user/client.
- Increased Productivity: Productivity is a measure of efficiency as well as Return on Investment (ROI). One aspect of productivity covers how many resources are utilised to build the product, how much investment required, how much time is taken for developing the product, etc. For example, the productivity of a system is said to be doubled if a product developed by a team of 'X' members in a period Of 'X' days is developed by another team of 'X/2' members in a period of 'X' days or by a team of 'X' members in a period of 'X/2' days. This productivity measurement is based on total manpower efficiency. Productivity in terms of Returns is said to be increased, if the

product is capable of yielding maximum returns with reduced investment. Saving manpower effort will definitely result in increased productivity. Usage of automated tools, wherever possible, is recommended for this. The initial investment on tools may be an additional burden in terms of money, but it will definitely save efforts in the next project also. It is a one-time investment. "Pay once use many time". Another important factor which can help in increased productivity is "reusable effort". Some of the works required for the current product development may have some common features which you built for some of the other product development in the projects you executed before. Identify those efforts and design the new product in such a way that it can directly be plugged into the new product without any additional effort. This will definitely increase the productivity by reducing the development effort. Another advised method for increasing the productivity is by using resources with specific skill sets which matches the exact requirement of the entire or part of the product (e.g. Resource with expertise in Bluetooth technology for developing a Bluetooth interface for the product). This reduces the learning time taken by a resource, who does not have prior expertise in the particular feature or domain.

19. What are simulators?

- Simulators simulate the target hardware and the firmware execution can be inspected using simulators.
- The features of simulator based debugging are listed below.
  l. Purely software based
  2. Doesn't require a real target system
  3. Very primitive (Lack of featured 1/0 support. Everything is a simulated one)
  4. Lack of Real-time behaviour.
- Advantages of Simulator Based Debugging:
  - No Need for Original Target Board: Simulator based debugging technique is purely software oriented. IDE's software support simulates the CPU of the target board. User only needs to know about the memory map of various devices within the target board and the firmware should be written on the basis of it. Since the real hardware is not required, firmware development can start well in advance immediately after the device interface and memory maps are finalised. This saves development time.
  - Simulate I/0 Peripherals: Simulator provides the option to simulate various I/O peripherals. Using simulator's I/O support you can edit the values for I/O registers and can be used as the input/output value in the firmware execution. Hence it eliminates the need for connecting I/O devices for debugging the firmware.
  - Simulates Abnormal Conditions: With simulator's simulation support you can input any desired value for any parameter during debugging the firmware and

can observe the control flow of firmware. It really helps the developer in simulating abnormal operational environment for firmware and helps the firmware developer to study the behaviour of the firmware under abnormal input conditions.

- Limitations of Simulator based Debugging:
  - Deviation from Real Behaviour: Simulation-based firmware debugging is always carried out in a development environment where the developer may not be able to debug the firmware under all possible combinations of input. Under certain operating conditions we may get some particular result and it need not be the same when the firmware runs in a production environment.
  - Lack of real timeliness: The major limitation of simulator based debugging is that it is not real-time in behaviour. The debugging is developer driven and it is no way capable of creating a real time behaviour. Moreover in a real application the I/O condition may be varying or unpredictable. Simulation goes for simulating those conditions for known values.

20. Explain the types of files generated on Cross Compilation.

- List Files (.lst) – Listing file is generated during the cross-compilation process. It contains an information about the cross compilation process like cross compiler details, formatted source text (‘C’ code), assembly code generated from the source file, symbol tables, errors and warnings detected during the cross-compilation process.
- Preprocessor Output file - It contains preprocessor output for preprocessor instructions used in the source file. This file is used for verifying the operation of Macros and preprocessor directive.
- Object file (.obj file) - Cross-compiling each source module converts the Embedded C/Assembly instructions and other directives present in the module to an object (.obj file).
- Map file (.map) - Also called as Linker List file. Map file contains information about the link/locate process and is composed of a number of sections.
- Hex File (.hex) - It is a binary executable file created from the source code. The file created by linker/locater is converted into processor understandable binary code. The tool used for converting and object file into a hex file is known as object to Hex converter. Hex file have specific format and it varies for different processor and controller. Two commonly used hex file format are Intel Hex & Motorola Hex. Both Intel and Motorola hex file format represent data in the form of ASCII codes.