2. The actual emulation device

3. A pin adapter that gives the emulator's target connector the same "package" and pin out as the microcontroller to be emulated.

- An emulator is a piece of hardware that looks like a processor, has memory like a processor, and executes instructions like a processor but it is not a processor.
- The advantage is that we can probe points of the circuit that are not accessible inside a chip.
- It is a combination of hardware and software.

DEBUGGERS

What is Debugging?

- Debugging in embedded application is the process of diagnosing the firmware execution, monitoring the target processor's registers and memory while the firmware is running and checking the signals from various buses of embedded hardware.
- Classified as
    - Hardware Debugging
    - Firmware Debugging
- Hardware Debugging:
    - Deals with monitoring of various bus signals and checking the status lines of target hardware.
- Firmware Debugging:
    - Deals with examining the firmware execution, execution flow, changes to various CPU registers and status registers on execution of the firmware to ensure that the firmware is running as per the design

Why is Debugging required?

- Firmware Debugging is performed to figure out the bug or the error in the firmware which creates the unexpected behavior.

<u>Debugging Techniques</u>

1. Incremental EEPROM Burning Technique:

- Most primitive technique
- Code is separated into different functional code units.
- Code is burned into EEPROM in incremental order.
- LED or BUZZER provided to indicate correct functioning
- Time Consuming but is a onetime process.

2. Inline Breakpoint Based Firmware Debugging

- An inline debug code is inserted within the firmware at a point where we want to ensure correct execution
- Debug code is ***printf()*** function.
- View debug code generated data on the "Hyper Terminal"
    - Configure serial communication setting of the Hyper Terminal connection to the same as that of serial communication setting configured in the firmware (Baudrate, Polarity, Stop Bit, Flow Control).
    - Connect target board's serial port to development PC's COM port using RS232 cable.

3. Monitor Program Based Firmware Debugging

- Monitor Program which acts as supervisor is developed.
- It controls the downloading of user code into code memory, inspect and modifies register/memory locations, allow single stepping of source code etc.
- It always listen to serial port of target device and according to command received it performs command specific actions.
- The first step in any monitor program development is determining a set of commands for performing various operations like firmware downloading, memory register inspection/modification, single stepping, etc. Once the commands for each operation is fixed write the code for performing the actions corresponding to these commands.
- The commands may be received through any of the external interface of the target processor (e.g. RS-232C serial interface/parallel interface/USB, etc.).
- The monitor program should query this interface to get commands or should handle the command reception if the data reception is implemented through interrupts. On receiving a command, examine it and perform the action corresponding to it.
- The entire code handling the command reception and corresponding action implementation is known as the "monitor program".
- After the successful completion of the monitor program development, it is compiled and burned into the FLASH memory or ROM of the target board. The code memory containing the monitor program is known as the Monitor ROM

The Monitor program contains the following set of minimal features:

- Command set interface to establish communication with the dubbing application
- Firmware download option to code memory

- Examine and modify processor registers and working memory(RAM)
- Single step program execution
- Set breakup points in firmware execution
- Send debug info to debug application running on host machine

- The monitor program usually resides at the reset vector (code memory 0000H) of the target processor
- The actual code memory is downloaded into a RAM chip which is interfaced to the processor in the Von Neumann architecture model.
- The von Neumann architecture model is achieved by ANDing the PSEN and RD signals of the target processor (In case of 8051) and connecting the output of AND Gate to the Output Enable (RD) pin of RAM chip. WR signal of the target processor is interfaced to the WR signal of the Von Neumann RAM.
- An address decoder circuit maps the address range allocated to the monitor ROM and activate the Chip Select (CS) of the ROM if the address is within the range specified for the Monitor ROM.
- A user program is normally loaded at location 0x4000 or 0x8000. The address decoder Circuit ensures the enabling of the RAM chip (CS) when the address range is outside that allocated to the ROM monitor.

<u>The major drawbacks of monitor based debugging system are</u>

- The entire memory map is converted into a Von Neumann model and it is shared between the monitor ROM, monitor program data memory, monitor program trace buffer, user written firmware and external user memory. For 8037, the original Harvard architecture supports 64K code memory and 64K external data memory (Total 128K memory map). Going for a monitor based debugging shrinks the total available memory to 64K Von Neumann memory and it needs to accommodate all kinds of memory requirement
- The communication link between the debug application running on Development PC and monitor program residing in the target system is achieved through a serial link and usually the controller's On-chip UART is used for establishing this link. Hence one serial port of the target processor becomes dedicated for the monitor application and it cannot be used for any other device interfacing. Wastage of serial port! It is a serious issue in controllers or processors with single UART

4. In Circuit Emulator Based Firmware Debugging

Functional Units

- Emulation Device
- Emulation Memory
- Emulator Control Logic
- Device Adaptors

Emulation Device: replica of target CPU

- Standard Chip
- Programmable Logic Device(PLD)

Emulation Memory: Replacement for EEPROM of target device

- It is the RAM incorporated in the emulator device
- Acts as Trace Buffer. Trace buffer is a memory pool holding the instructions executed/registers modified/related data by the processor while debugging. The common features of trace buffer memory and trace buffer data viewing are listed below. Trace buffer records cache bus cycle in frames. Trace data can be viewed in the debugger application as Assembly/Source code. Trace buffering can be done on the basis of a Trace trigger (Event). Trace buffer can also record signals from target board other than CPU signals. Trace data is a very useful information in firmware debugging

Device adaptors

- Act as an interface between the target board and emulator POD.
- Normally pin-to-pin compatible sockets which can be inserted/plugged into the target board for routing the various signals from the pins assigned for the target processor.
- The device adaptor is usually connected to the emulator POD using ribbon cables. The adaptor type varies depending on the target processor chip package.

- The above mentioned emulators are almost dedicated ones, meaning they are built for emulating a specific target processor and have little or less support for emulating the derivatives of the target processor for which the emulator is built. This type of emulators usually combines the entire emulation control logic and emulation device in a single board. They are known as Debug Board Modules (DBMs).
- An alternative method of emulator design supports emulation of a variety of target processors. Here the emulator hardware is partitioned into two
  - Base Terminal

- Probe Card.

- The Base terminal contains all the emulator hardware and emulation control logic except the emulation chip (Target board CPUs replica). The base terminal is connected to the Development PC for establishing communication with the debug application. The emulation chip is mounted on a separate PCB and it is connected to the base terminal through a ribbon cable.
- The Probe Card board contains the device adaptor sockets to plug the board into the target development board. The board containing the emulation chip is known as the Probe Card. For emulating different target CPUs the Probe Card will be different and the base terminal remains the same.

5. On Chip Firmware Debugging

- Today almost all processors incorporate built in debug modules called On Chip Debug (OCD) support.
- Though OCD adds silicon complexity and cost factor, from a developer perspective it is a very good feature supporting fast and efficient firmware debugging.
- The On Chip Debug facilities integrated to the processor/controller are chip vendor dependent and most of them are proprietary technologies like Background Debug Mode. Some vendors add 'on chip software debug support through JTAG (Joint Test Action Group) port.
- Processors/controllers with OCD support incorporate a dedicated debug module to the existing architecture
- Usually the on-chip debugger provides the means to set simple breakpoints, query the internal state of the chip and single step through code.
- OCD module implements dedicated registers for controlling debugging.
- An On Chip Debugger can be enabled by setting the OCD enable bit
- BDM and JTAG are the two commonly used interfaces to communicate between the Debug application running on Development PC and OCD module of target CPU.
- Background Debug Mode (BDM) interface is a proprietary On Chip Debug solution from Motorola
- BDM defines the communication interface between the chip resident debug core and host PC where the BDM compatible remote debugger is running.
- BDM makes use of 10 or 26 pin connector to connect to the target board.

- Serial data in (DSI), Şerial data out (DS0) and Serial clock (DSLR) are the three major signal lines used in BDM.
- DSI sends debug commands serially to the target processor from the remote debugger application
- DSO sends the debug response to the debugger from the processor
- Synchronisation of serial transmission is done by the serial clock DSCLK generated by the debugger application. Debugging is controlled by BDM specific debug commands. The debug commands are usually 17-bit wide. 16 bits are used for representing the command and I bit for status/control.
- Chips with JTAG debug interface contain a built-in JTAG port for communicating with the remote debugger application.

  The signal lines of JTAG protocol are explained below
  - Test Data In (TDI): It is used for sending debug commands serially from remote debugger to the target processor
  - Test Data Out (TDO): Transmit debug response to the remote debugger from target CPU.
  - Test Clock (TCK): Synchronizes the serial data transfer
  - Test Mode Select (TMS): Sets the mode of testing.
  - Test Reset (TRST): It is an optional signal line used for resetting the target CPU.

TARGET HARDWARE DEBUGGING

Firmware is bug free and everything is intact with the board. Still embedded product need not function as per the expected behavior in the first attempt. This is due to hardware related reasons like :

- Dry soldering of components
- Missing connections in PCB due to un-noticed error in PCB layout design
- Misplaced components
- Signal corruption due to noise

Hardware Debugging

- The monitoring of various signals of the target board like address/data lines, checking the inter connection among various components, circuit continuity checking etc.
- Not similar to firmware debugging

Hardware Debugging Tools

- Magnifying Glass
- Multimeter
- Digital CRO
- Logic Analyser
- Function Generator

Magnifying Glass

- It is a powerful visual inspection tool.
- Used for examining the target board for :
    - Dry soldering components
    - Missing components
    - Improper placement of components
    - Improper soldering
    - Tracks PCB damage
    - Short of track

Multimeter

- Used for measuring various electrical quantities like Voltage, Currents, Resistance, Capacitance, Continuity checking, Transistor checking, Cathode and Anode identification of diode etc.
- Primary debugging tool for physical contact based hardware debugging.
- It is mainly used in embedded hardware debugging for:
    - Checking the circuit continuity between different points on the board
    - Measuring the supply voltage
    - Checking the signal value, polarity etc.
    - Both analog and digital version are available

Digital Cathode Ray Oscilloscope

Used for

- Waveform capturing and analysis
- Measurement of signal strength
- Analysing interference noise in the power supply line and other signal lines
- Connecting point under observation on the target board to the channels of the oscilloscope, waveform can be captured and analysed for expected behavior.

- Digital CRO are available for high frequency support and incorporates modern technique for recording waveform over a period of time, capturing the waves on the basis of a configurable event for target board.

Logic Analyser

- It is similar to CRO
- Used to capture digital data (logic 1 or 0) from digital circuitry, whereas CRO is used to capture all kind of waves.
- The total no of logical signals that can be captured using CRO is limited to the no of channels. Logical analyser contain special connectors and chips which can be attached to the target board for capturing the digital data.
- In target board debugging applications, a logic analyser captures the states of various port pins, address bus and data bus of the target processor/controller, etc.
- Logic analysers give an exact reflection of what happens when a particular line of firmware is running.

Function Generator

- Function generator produce various periodic waveforms like sine wave,square wave, saw-tooth wave etc. with different frequency and amplitude.
- It is not a debugging tool, it is an input stimulator tool.
- The target board requires periodic waveform with particular frequency as input to some part of the board.