Y-chart

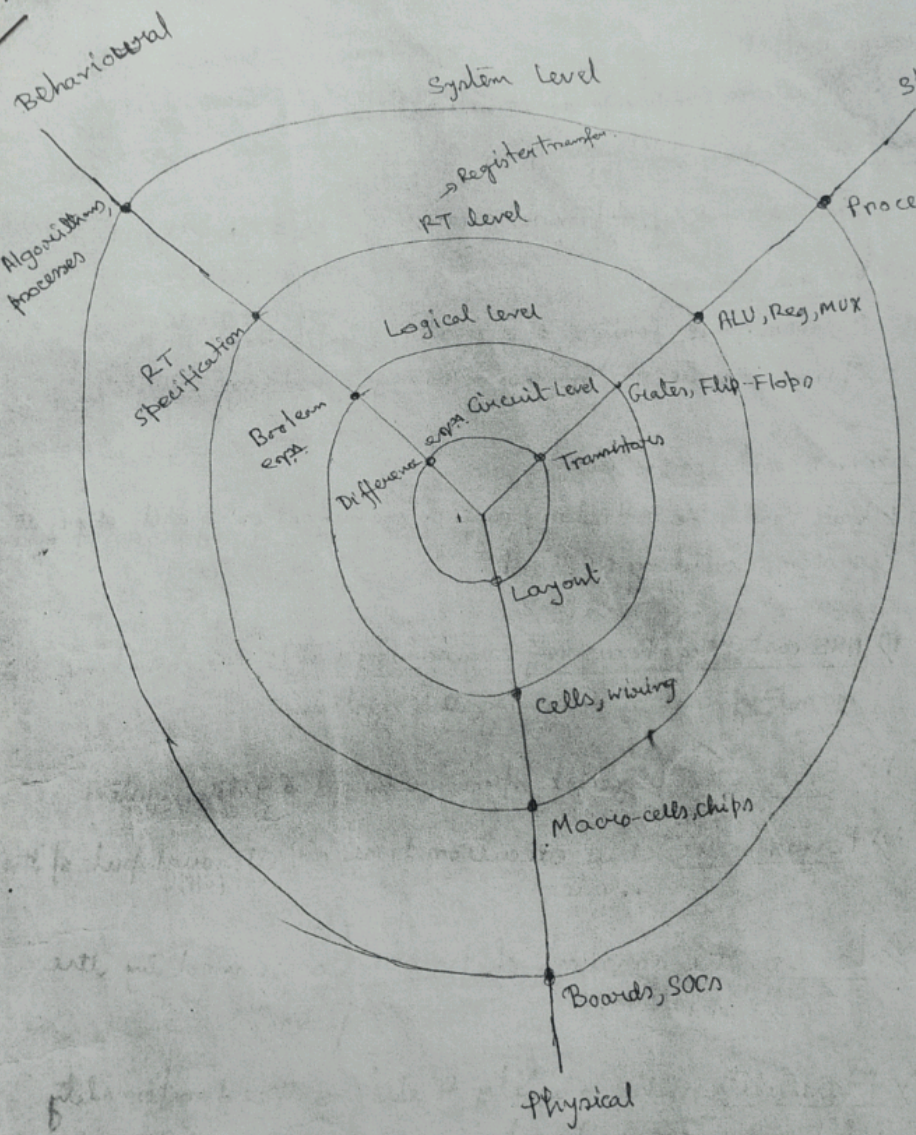


Fig:- Design Levels of Embedd ~~level~~ system.
(Y-chart)

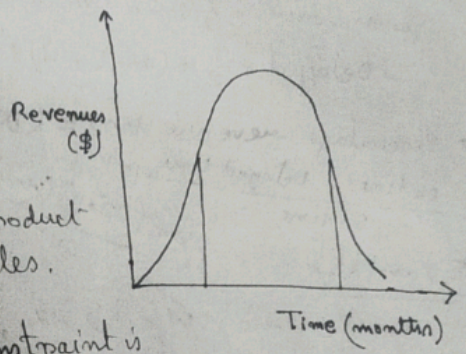
Design Challenges:- (optimizing design issues)

1. Design goal:
   - Construct an implementation with desired functionali
2. Key design challenge:
   - Simultaneously optimize numerous metrics (issues)
3. Design metric: (issues)
   - A measurable feature of a system's implementation
   - Optimizing design ~~metrics~~ issues is a key challenge.

4. Common metrics:- (issues)
   i) Unit cost: the monetary cost of manufacturing each copy of system, excluding NRE cost.
   ii) NRE cost (Non-Recurring Engineering cost): the one-time monetary cost of designing the system.
   iii) Size: the physical space required by the system
   iv) Performance: the execution time or throughput of system. (or)
   v) Power: the amount of power consumed by the system.
   vi) Flexibility: The ability to change the functionality of the system without incurring heavy NRE cost
   vii) Time-to-prototype: The time needed to build a working version of the system.
   viii) Time-to-market: The time required to develop a system to the point that it can be released and sold to customers.
   ix) Maintainability: the ability to modify the system after its initial release.

Time-to-market: (a demanding design metric)

1. Time required to develop a product to the point it can be sold to customers.

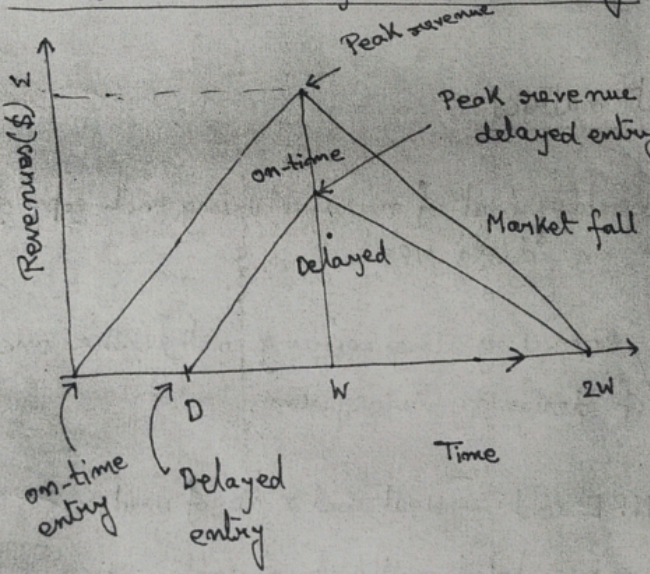


2. Market window:
   - Period during which the product would have highest sales.

3. Average time-to-market constraint is about 8 months.

4. Delays can be costly.

Losses due to delayed market entry



- Simplified revenue model
  - Product life = 2W, Peak at W.
  - Time of market entry defines a triangle, representing market penetration.
  - Triangle area equals revenue.

- Loss
  - The difference between the on-time and delayed

- Area = 1/2 * base * height
  - on time = $\frac{1}{2} * 2W * W$
  - Delayed = $\frac{1}{2} * (W - D + W) * (W - D)$

$= 2W^2 - 3WD + D^2$

$= [(W-D)^2 + W^2 - WD] * \frac{1}{2}$

$= [W^2 - 2WD + D^2 + W^2 - WD] * \frac{1}{2}$

$= (2W^2 - 3WD + D^2) * \frac{1}{2}$

- Percentage revenue loss = $(D(3W - D)/2W^2) * 100\%$

$= \frac{\text{On time} - \text{Delayed time}}{\text{On time}} \times 100\%$

$\frac{2W^2 - (2W*D)(W-D)}{2W^2}$ $\quad \frac{2W^2 - (2W^2 - 2WD + DW + D^2)}{2W^2}$

$\frac{2W^2 - (2W^2 - 3DW + D^2)}{2W^2}$

Example:-

Life time $2W = 52$ wks, delay $D = 4$ wks

$(4(3*26 - 4)/2*26^2) * 100 = 22\%$.

Life time $2W = 52$ wks, delay $D = 10$ wks.

$(10*(3\times26 - 10)/2\times26^2) = 50\%$.

So delays are costly.

NRE and unit cost matrics :-

Unit cost: the monetary cost of manufacturing each copy of the system, excluding NRE cost.

NRE cost (Non-Recuring Engineering cost): The one-time monetary cost of designing the system.

Total cost = NRE cost + unit cost x no. of units

Per-product cost = total cost / no. of units

= (NRE cost / no. of units) + Unit cost

- Example:-

NRE = $2000, unit = $100

For 10 units

- total cost = $2000 + 10 * $1000 = $3000

Per product cost = $2000/10 + $100 = $300

A. Karmakar

Software Issue-Programming Languages

S/W issues: i) Programming Languages
ii) Time criticality of the Applications
iii) Real time operating systems (RTOS)

- Assembly Level Language
- Procedural Language - 'C'
- Object Oriented Language - 'C++', Embedded JAVA
- Hardware Design Language - VHDL, VeriLog, Handel-C

Software Issue-Time Criticality

- Data Arrival Rate
- Execution Time Required
- Respond to an external event within a deterministic time
- Task(Context) Switching

Software Issue-

Real Time Operating Systems(RTOS).

❑ Real Time system

a) A Real Time System is that they must receive and respond to a set of external stimuli within rigid and critical time constraints referred to as deadlines

b) Tasks have to be completed under the constraint of the deadline.

Software Issue-(Contd.)

Real Time Operating Systems(RTOS)

- Hard vs. soft real time
  - Hard real time - drop dead time - Ex: shuttle control, flight control
  - Soft real time - tasks can go beyond the deadlines but with consequences, like the failure to establish network connection

Applications of RTOS :-

1. Air Traffic Control System (ATC)
2. Radar Signal Processing System.
3. Missile control system.
4. Space Craft control system.
5. Microprocessor Based Patient care system.

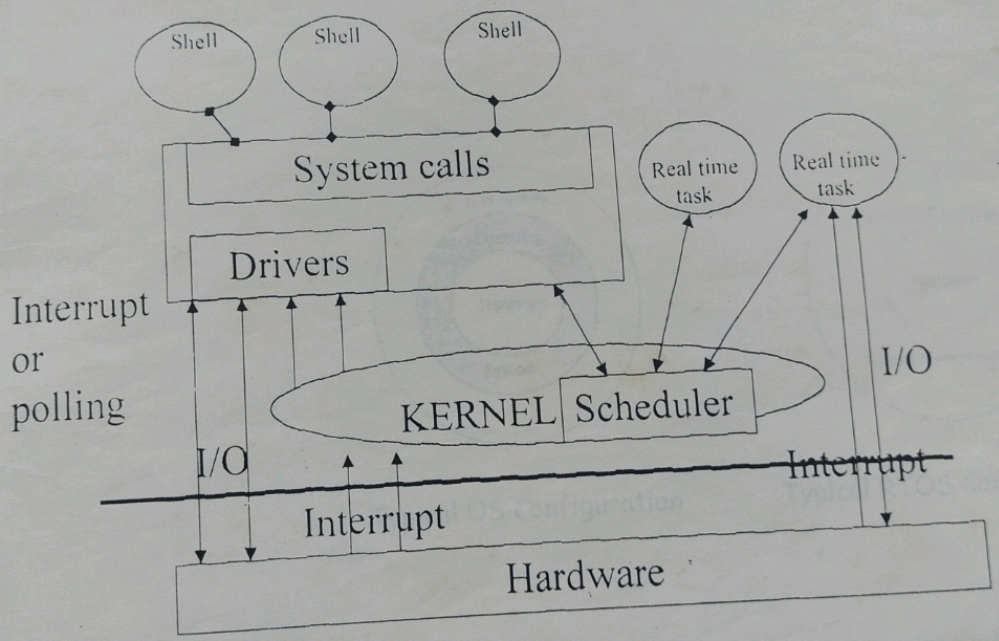
Software Issue-(Contd.)

Real Time Operating Systems(RTOS)

- **Characteristics of Real Time Operating Systems**
  - Handles Time-Critical Tasks
  - Handles Multiple Threads based on **preemptible priority .**
  - **Must support predictable thread synchronisation(semaphores,message passing etc).**
  - Handles Nested Interrupts
  - Assigns of Priority to the Threads
  - Facilitates Hardware Control.
  - **Behaviour of the OS must be predictable**

Structure Of RTOS

- RTOS from different vendors: VxWorks, pSOS, RTLinux, Ecos
  (Seller)



Software Testing Techniques

- "White Box" or "Code-Based" Testing
- "Black Box" or "Functional" Testing

Testing(contd.)

"White Box" or "Code-Based" Testing

- Examines the internal design of a program
- Looks for software failures based on the structure and organization of the source code.
- Typical requires that the tester has a detailed knowledge of the software structure and its intended role.
- Code-based testing exercises the software uniformly, ensures that the components work together, and that the interfaces work correctly.

Testing(contd.)

"White Box" or "Code-Based" Testing

- For embedded systems, code-based testing is not required to be done on the target hardware. It is often more practical to do white box testing on the development environment using hardware simulation.

Testing(contd.)

"Black Box" or "Functional" Testing

Looks for software failures based on the intended use and features of the software. Black box tests are designed without knowledge of the program's internal structure.

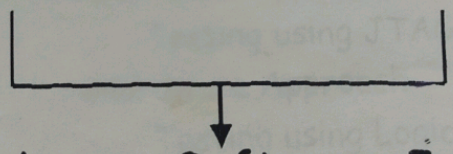
The greatest strength of black box testing comes in its independence from the code, by looking at the system from the real-world perspective.

Strongly correlate to the requirements, the quality of the requirements will affect the resultant tests.

Testing(contd.)

Testing Phases:

Software	Hardware
■ Module testing	Board Level Testing
■ Functional testing using simulation tools	-



Hardware-Software Integrated Testing

Testing(contd.)

Hardware Testing Techniques

- Bare Board Testing
- Functional Testing after mounting the components
  - -Software Approach
    Testing using JTAG
  - -Hardware Approach
    Testing using Logic State Analyzer, CRO etc.

Testing(contd.)

Testing Tools

Software Tools

- Simulator
- Performance Analysis Tools
- GUI Testers (Graphical User Interface)
- Debugger
- JTAG based debugger ?

Hardware Tools

- Logic State Analyzer
- CRO
- In Circuit Emulator ?