EMBEDDED SYSTEM

ELECTRICAL ENGINEERING DEPARTMENT

VEER SURENDRA SAI UNIVERSITY OF TECHNOLOGY

Version 1.0 , Dated 17-04-2015

Prepared By

Er B.K.Rana

Assistant Prof ., Electrical Engg.

Scheme

Ref 1:Introduction to embedded system:by SHIBU K.V., TMH Publication

Ref 2: Embedded Systems – Architecture , Programming and Design – by RAJKAMAL , TMH Publication

	Course Item	Ref 1	Ref 2	Other
1.	MODULE _I Introduction to Embedded System : What is embedded system	Page 4 Sec 1.1		
2.	Embedded systems Vs General Computing system	Page 4 Sec 1.2		
3.	History of embedded systems , classification of embedded system	Page 5,6 Sec 1.3 , Sec 1,4		
4.	Major application area of embedded sys	Page 7 Sec 1.5		
5.	Purpose of embeded system	Page 8 Sec 1.6		
6.	Typical Embedded sys: Core of embedded system	Page 15 Chap 2		
7.	Memory	Page 28 Sec 2.2		
8.	Sensors	Page 35 Sec 2.3		
9.	Actuators	Page 35 Sec 2.3		
10.	Communication Interface	Page 45 Sec 2.4		
11.	Embedded Firmware	Page 59 Sec 2.5		
12.	Other system component	Page 60 Sec 2.6		
13.	PCB and Passive components	Page 64 Sec 2.7		
14.	MODULE _II Hardware Software co-	Page 204 Chap 7		

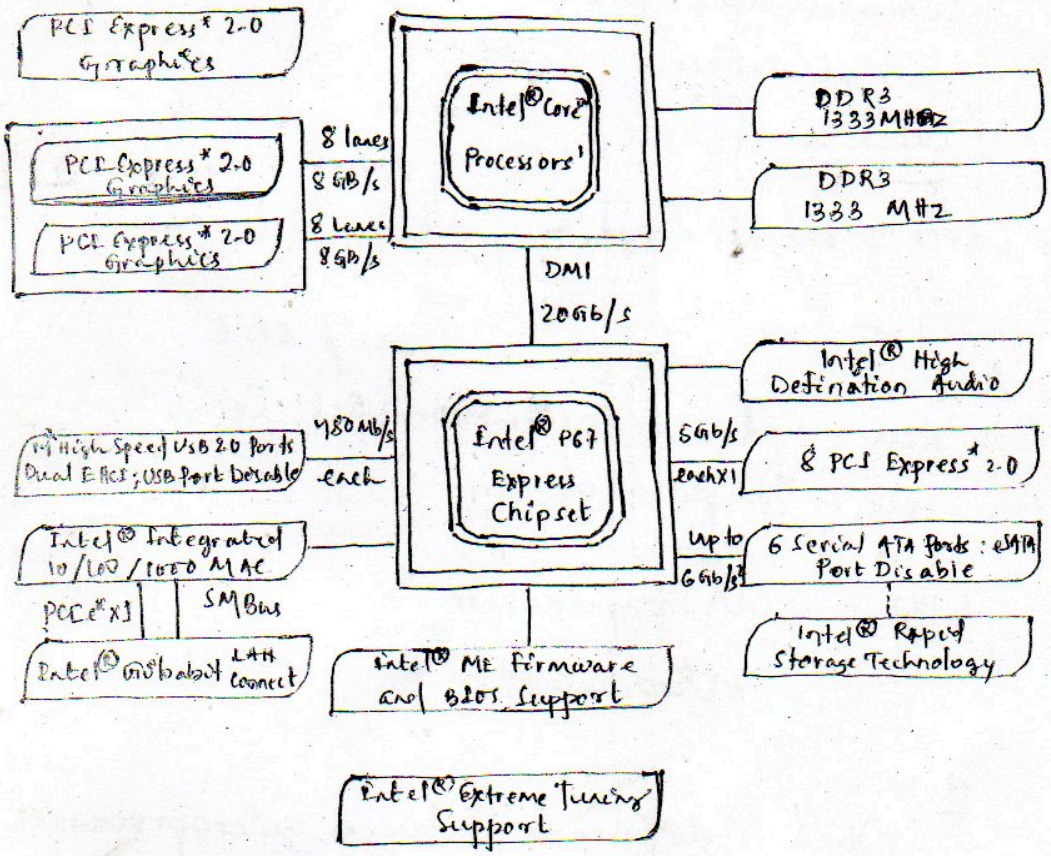
	design and programming model			
15.	Fundamental Issues in Hardware –Software co design	Page 205 Sec 7.1		
16.	Comutational model Embedded System	Page 207 Sec 7.2		
17.	Introduction to unified modeling language (UML)	Page 214 Sec 7.3	Page 295 Sec 6.5	
18.	Hardware software tradeoff	Page 219 Sec 7.4		
19.	Embeded Hardware design and development : Analog design components	Page 228 Sec 8.1		
20.	Digital Electronics component	Page 230 Sec 8.2		
21.	VLSI and Integrated Circuit design	Page 245 Sec 8.3.1		
22.	Embedded Firmware design and development	Page 302 Chap 9	Page 25 Sec 1.4.8	
23.	Embeddded Firmware design approach	Page 303 Sec 9.1		
24.	Embedded firmware development language	Page 313 Sec 9.2.2		
25.	Programming in Embedded C	Page 318 Sec 9.3	Page 235,237 Sec 5.1,5.2	
26.	MODULE _III			
27.	Real Time operating system (RTOS) based embedded system design	Page 381 Chap 10	Page 350 Chap 8	
28.	Operating system basic	Page 382 Sec 10.1		
29.	Types of operating systems	Page 386 Sec10.2		
30.	Task process and threads	Page 390 Sec 10.3		
31.	Multiprocessing and multi tasking	Page 402 Sec 10.4		
32.	Task scheduling	Page 404 Sec 10.5		
33.	Thread and process scheduling	Page 422 Sec 10.6		
34.	Putting task communication	Page 426 Sec 10.7	Page 303 Chap 7	
35.	Task synchronization	Page 442 Sec 10.8		
36.	Device drivers	Page 476	Page 2.20	

		Sec 10.9	Sec 4.9	
37.	How to choose RTOS	Page 478 Sec 10.10		
38.	MODULE _ IV			
39.	Embedded system Development environment (IDE)	Page 556 Chap 13 Sec 13.1		
40.	Types of files generated on cross compilation	Page 588 Sec13.2		
41.	Dissembler / Decompiler	Page 597 Sec 13.3		
42.	Simulators	Page 598 Sec 13.4		
43.	Emulator	Page 599 Sec 13.4		
44.	Debugging	Page 599 Sec 13.4		
45.	Target Hardware debugging	Page 607 Sec 13.5		
46.	Design Case studies	Page 669 Appendix II	Page 43 Sec 1.10	
47.	Digital Clock	Page 669 Appendix II- 1	Page 44 Sec 1.10.3	
48.	Batterry operated smart card Reader	Page 699 Appendix II-2		
49.	Automated meter reading sysytem (AMR)	Page 701 Appendix II-3		
50.	Digital Camera	Page 703 Appendix II-4	Page 48 Sec 1.10.4	

Assignments

1. Draw IBM PC architecture and describe all components
2. What is difference between Microprocessor and Micro Controller .
3. Describe a CPU which does floating point calculation.
4. Describe Math coprocessor
5. Describe Graphics processor
6. Write 5 general purpose microprocessor and 5 micro controller .
7. Write a page on DSP processor
8. What is PLD and describe comercial types .
9. Write key words of C language . Give 1 line description for each.
10. Differentiate Task , process , thread .
11. Name five commercial operating system and 5 real time operating system.
12. Draw RESET curcuit for a CPU .
13. Draw all diagrams used in High level language UML .
14. What is Network Operating System (NOS)? Give it's seven layer OSI architecture .
15.
16.
17.

Architecture of PC



1. Compatible with 2nd Generation Intel® Core™ processor family
2. All SATA ports capable of 3 Gb/s 2 ports capable of 6 Gb/s

Computing System

23.07.2014

- Hardware
  - CPU
  - Memory
  - I/O peripheral
  - Interrupt controller.
  - Timer.
  - Display.
- Software

Operating System

Communication System.

device driver.

clock speed.

CPU is characterised by address, data, Speed.

RISC / CISC.

RISC - Reduced Instruction Set Computer.

CISC - Complicated " " "

CPU - Microprocessor

Microcontroller.

H.W.

What is difference between microprocessor and micro controller. (2 page)

CPU operated with floating point calculation.

Math Coprocessor.

Graphics Coprocessor.

Commercial CPUs

INTEL

8051
8085 Pentium
8086
80186
80286
80386
80486

Motorola

: 68000
68010
68020
68030
68040

~~Spark~~
Power PC

Sparc

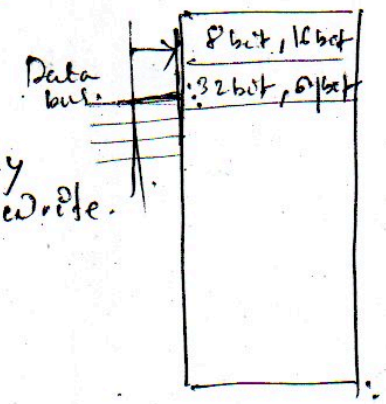
Slims of each processor.
- Data Size
- Address Size
- Speed.

Processor Comparison.

Speed	Data size	address.	RISC / CISC

Memory

EPROM — Read only
RAM. — Read / write.



Address bus

0-7

0-15

0-31 ...

Q. What are the Commercial EPROM chip and RAM chip available in the Market?

www.wheelos.com

24.07.2014

Hardware

CPU

Memory

I/O device

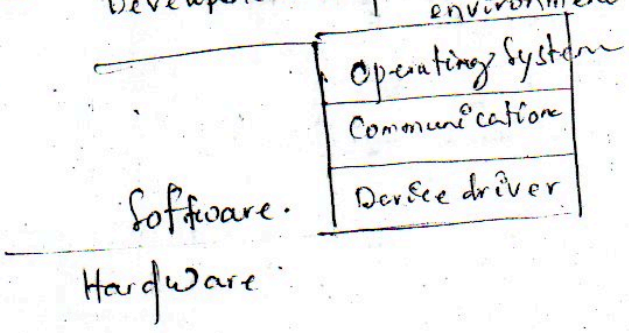
Interrupt Controller — Parallel 8255, Serial 8251, 8259

Timer

Bus Controller — 8253, 8254

findout chip no. for bus Controller.

Architecture of General Computing environment



Commercial operating systems

Windows
Unix
Lynux
MAC
RTOS ...

Embedded Equivonment

- General, Tomer Characteristics.

RTOS - Real time operating system.

Soft Hard,

& RMX
MAC-OS II
Vx works ..
lynux.

ATM machine
Designed based on windows operating system.

DIFFERENTIATE:

Section Page 1-2, 4, 5	General Computing System	Embedded System
	(1) A system which is combination of a genetre hardware and general purpose operating system for executing variety of application.	(1) A system which is combination of special purpose ~~operating system~~ hardware and embedded os for executing a specific set of application.

GPOS.

2. Contains general purpose operating system (GPOS)	2. May or May not have operating system OS is called Embedded OS.
3. Upgradability and altering program by user.	3. For upgrading and altering program you have to go to manufacturer.
4. Performance is the deciding factor in selection of the system. Always faster Memory Time	4. Power Consumption, Memory, Time. } Performance issues.
5. Response requirements are not time concealed.	5. For certain type of embedded system like mission critical system, The response time requirement is highly critical.
6. Need not be deterministic in execution behaviour.	6. Execution behaviour is deterministic for certain embedded system called Hard real time system.

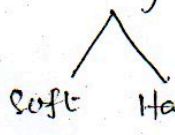
25.07.2014

Classification of Embedded System

4 types of classification.

(1) Based on Generation
(2) Complexity and performance requirement
(3) Based on deterministic behaviour.
(4) Based on time triggering.

Real time System



(1) Based on generation.

(1) First generation

8 bit microprocessor type.

8085,

8 & 4 bit microcontroller. (Search internet)

8051

No OS required

Assembly language programming.

(2) Second Generation

16 bit microprocessor

8086, 80186

8 bit or 16 bit microcontroller.

then.

what is 16 bit microcontroller.

Instruction set is complex
compared to generation 1.

Embedded operating System.

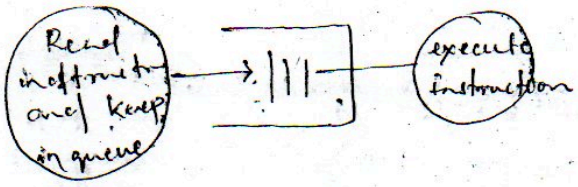
Application

Supervision Control And Data Acquisition
SCADA system

(3) Third Generation:

32 bit microprocessor.
16 bit micro controller.
DSP - Digital Signal Processing.
ASIC - Application Specific Integrator Circuit.

More complex instruction.
Instruction pipelining.



Dedicated
Embedded
and realtime operating System.
like iRMX is available.
Intel Realtime Multitasking operating system.

Application:

- Robotics
- Media.
- Industrial Process Controller
- Communication System.

(4) Fourth Generation:

SOC - System on chip.

Operating System specific Hardware.
and communication specific hardware
are integrated with cpu.
Multipole.
High performance Real time operating System.

Application

- Smart phone devices,
- Mobile internet devices.
- Industrial process Control.

Classification based on complexity and performance

(1) Small Scale embedded system.

(a) - Simple
- No performance requirement.
- 4 bit, 8bit, 16 bit.
- Example - Electronic toy application.
- small

(2) Medium Scale embedded system
- low-cost, 16-bit, 32bit processor
- small RTOS, performance.
- Industrial distribution.
- On computer Controlled Network

Example chip - MOTOROLA

(3) Large-scale embedded system
- 32-bit / 64 bit processor.
- time performance is missing critical.
- High performance RTOS.
- RISC.

Application of embedded system:

{ Shakti
Narendra
Amrit
Selvam

Page 7/8. Section 1.5.

1. Consumer Electronics - Camcoders
Cameras,

2. Household appliances - TV. DVD, ~~Washing~~
Washing Machine; fridge.
Microwave.

3. Home automation
and security system.
- air conditioner
- alarms.
- close Circuit Camera. (CCTV).

4. Automotive Industry
- Antilock breaking system (ABS)
- Engine Control.
- Ignition System.
- Automotive Navigation.

5. Telecom.
- Cellular phone and switches.

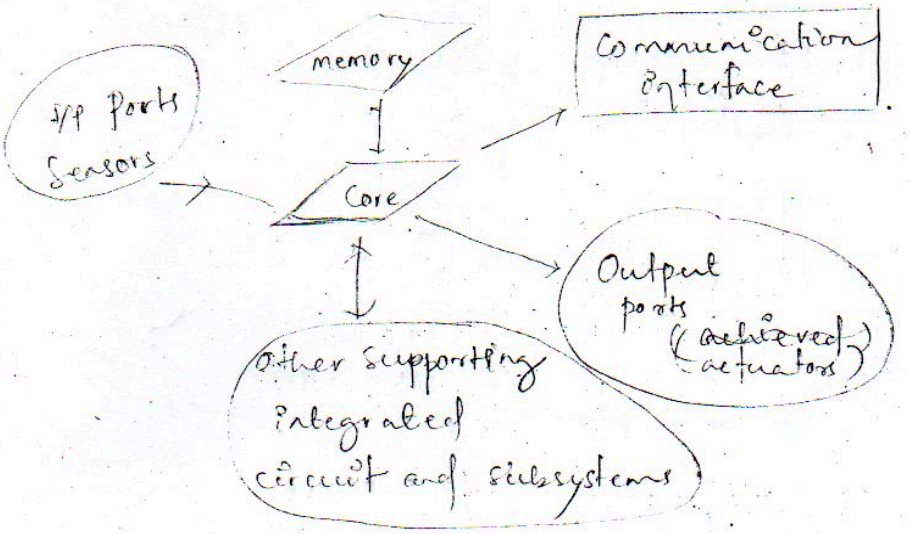
6. - computer peripherals.
Printer, Scanners, fax machines.

7. Computer Networking System.
- Routers, Switches, Hub, Gateways, ~~fire~~
fire alarms.

8. Measurement and instrumentation.
   - Digital CRO
   - Logic analyser
   - PLC
9. Health Care.
   - Scanner, EEG, ECG
10. Banking and R
   - ATM, currency counter.
11. Card Reader.
   - Barcode
   - Smartcard Reader.

@ 01.08.2014

Real world of Embedded System



Core of embedded System

1. General purpose and domain specific processor

a) Microprocessor

b) Microcontroller

c) DSP processors

4-iv

(a) Write a page on DSP processors.

Application Specific :

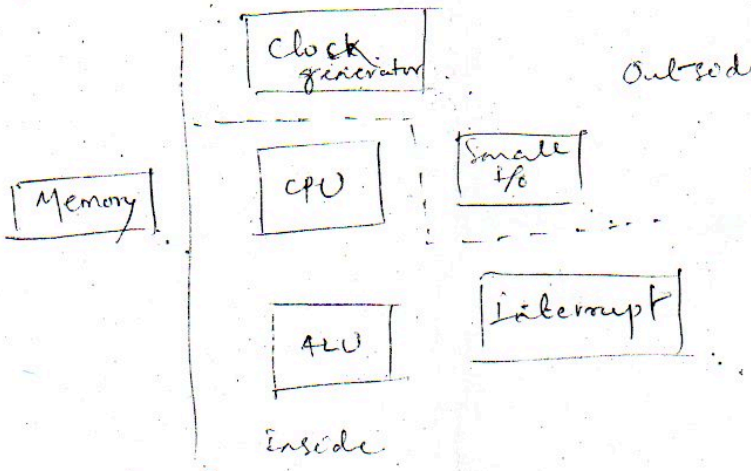
(b) Integrated Circuits (ASIC)

(c) Programmable Logic Device (PLD)

(d) Component Commercial of the Self Components (COTS)

Write in brief about COTS.

Microprocessor



Assignment:

(a) Draw Architecture of 8086

(b) Draw Architecture of 8051.

~~Geral~~ General purpose processor: (GPP)

Application Specific Instruction Set processor (ASIP)

Pentium-4

H.W

Readout

Microcontroller.

It contains

On a single chip:
1. CPU
2. ALU
3. RAM
4. ROM
5. I/O sub system
6. General purpose register arrays.

Examples:

Intel 8031
8048
8051

~~Hoo~~ Home Work:

Difference between Microprocessor micro processor

See 2.1.1 page 21

Digital Signal Processing Processors

(1) Powerful special purpose 8/16/32/64 bit processors.

(2) Computational engine is high purpose.

(3) It is 2-3 times faster than GPP.

H.W.

find out a Commercial DSP processor.

06.02.2014

Risc and Cisc processor

Risc : reduced instruction set Computer.

Cisc : Complicated instruction set Computer.

ADD :

Multiply using ADD.

$a \times y$ x is added y times.

Instead MUL. Complicated instruction

30-40 instructions.

speed is high

user has power to design Complicated instruction.

Page No. 22

Home Work: See 2.1.1.6 Differentiate between Risc and Cisc.

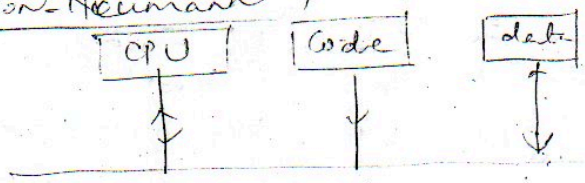
Architecture

- Havard Model
- Von - Neumann model of processor.

Havard
- data - code read/write

code read only.

Von-Neumann



(Instruction Bus) Bus 1 Bus 2 (Data Bus)

Havard Model

While executing code cpu can fetch next instruction and this is pipelined.

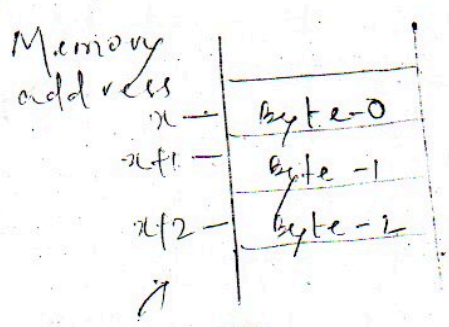
Comparision:

Havard Architecture	Von-Neumann Architecture
① Separate buses for instruction and data fetching	① Single shared bus for instruction and data fetching.

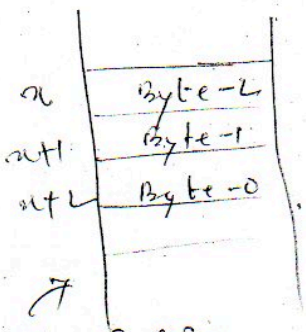
(2) Easier to Pipeline instruction and faster	(2) low performance
(3) Comparatively high Cost ~~and~~ Cheaper.	(3) Cheaper.
(4) No memory alignment problem	(4) Allows self modifying codes.
(5) Since data and Code Memory are separate no corruption of memory possible.	(5) Since data and Code are on same chip corruption happens.

Big-Endian and Little-Endian processor.

byte 2 - byte 1 - byte 0.



Little Endian.



Big-Endian.

07.08.2014

Firmware & Tools for Embedded system Development

See 1.4.8

Table 1.2

Rajkamal

1. Editors: for writing C, C++, Assembly Language Code in computer/pc.

   Allows entry, addition, deletion, insert, appending previously written lines or files.

   Turbo C editor

   Microsoft C editor

   Windows Environment

   VI editor.) Unix platform.

2. Interpreter:

   for expression - by expression or line by line translations of written code to machine executable code.

3. Compiler:

   Edited High Level Language Code is Compiled using Compiler tools and Error or mistakes, are generated if no error then High level code file is converted to machine code file or object file.

4. Linker:

Application Contains more than one file. All files will be compiled. Generated objects file will be linked to form executable file by linker.

5. Assembler: For translating assembly mnemonics to binary opcode (or instructions), that is into an executable file and binary file.

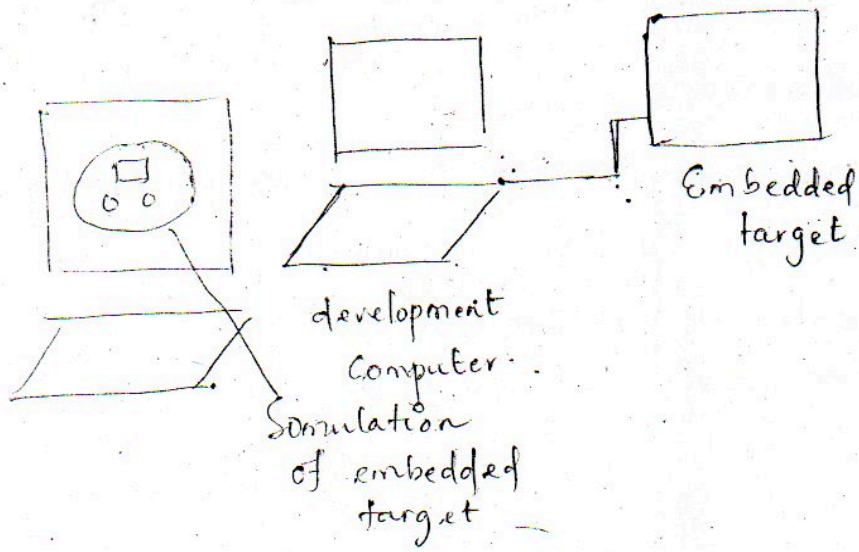
6. Cross Compiler / Assembler:

Cross Compiler / assembler

Convert source code to binary code using PC / any other computer for different microprocessors.

7. Simulators:

To simulate all functions of embedded system circuits including that, or additional memory and peripherals. It is independent of a particular target system.



8. Source Code Engineering:

(%user for Communicate Comp.)

For source code comprehension navigation and browsing.
Editing, debugging & configuring, disambling and enabeling feature.
Tools available in the market

- CVS
- Clearance
- logiscope.

08.08.2014

9. RTOS. Real Time Operating system

Layer 3	Application
Layer 2	API
Layer 1	RTOS
Layer 0	Hardware

This will be taught elaborately in module-III.

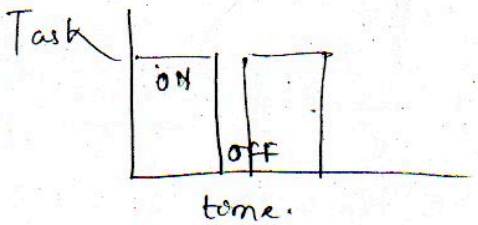
. API, Application Interface.

10. The Stethoscope.

For dynamically tracking the changes in programming variable or parameters.
It demostrate the sequence of multiple processes, and real time history.

(11) Trace scope :-

To help in tracing the changes in modules and tasks with time in X-axis.



(12) Integrated Development Environment: (IDE)

This development software and hardware environment that Consist of simulator, editor, compiler assemblers, RTOS debugger, stethoscope RTOS emmulator, tracer, logic analyzer and application code burner in Eprom and Flash.

(13) Prototyper :-

- Prototype is a small version of the actual large system. This is build to see behaviour of in less time.

(14) Locator

This uses cross assembler output and a memory location map. and provide a locator program output as a hex file. It is the final stage of software design of embedded system.

Table 1.3

Application Vs. firmware.
module
Requirement.

Module-II
Integrated Circuit VLSI.

Section-2.5 Embedded firmware.

Design issues of Embedded System

Table 1.8
Page-39
Raj Kamal.

① Power Dissipation

Power Consumed by the system.

for many small embedded system, battery operated system are used Eg. mobile, digital, camera, etc.

Consumption of the battery should be low.

② Performance

Time performance → Execution time should be minimised.

Memory performance. → low memory will be used.

Compiler
has optimization option for this purpose.

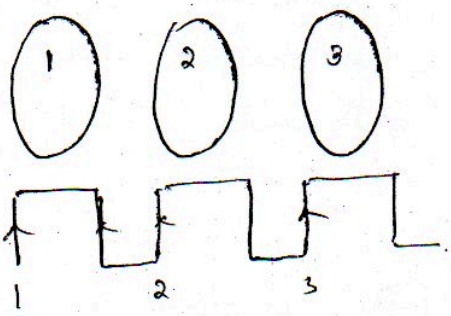
③ Process Deadlines.

The embedded application will have N number of process.

Process is a piece of code that is ~~st~~ scheduled by

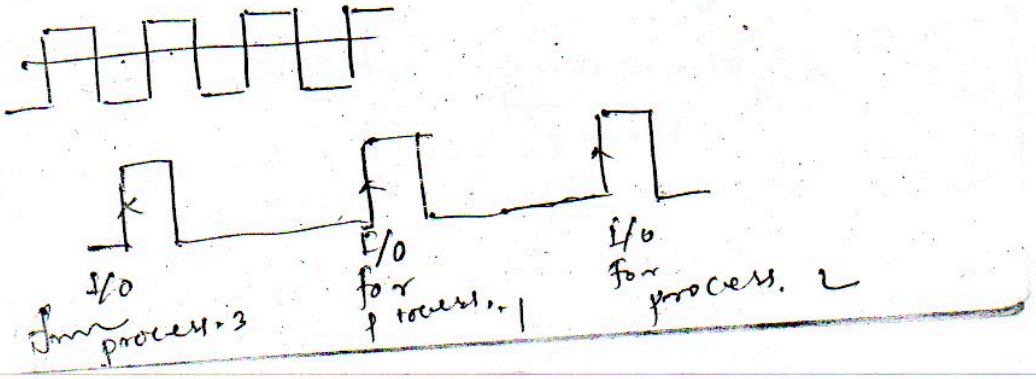
~~Created~~	created	function	called
Process	run	or	return.
Task	suspended	Procedure	
	exit		

Cost:



Process will Start.

Round robin Scheduling.



Interrupt based scheduling When I/O starts running the process.

Process has priority

Higher the priority first it will run.

There are no. of process in a system for example. keypad input processing.

Graphic display refresh.

Audio signal processing and video signal processing. These are deadlines, within which each of them may be required to finished computation and give result.

(4) User interface

(a) Keypad

GUI — Graphical User Interface.

VUI — Visual user interface.

(5) Size

Size of the system is measured in terms of

(1) Physical space requirements.

(2) RAM. in KB.

(3) Internal flash memory in terms of MB or GB.

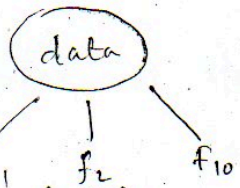
(4)

(6) Engineering Cost:

Initial Cost of developing, debugging.

Set 3 data function
Set 4 data function.
Set 5 data function.

Any data which is public is prone to corruption.



functions are clustered around data.

OOP characteristic

Encapsulation.

Data and function together defined into object.

Class is a template for an object.

oop class classname

```
{
  data
  function
}.
```

Object is programmed by class is object oriented programming language.

This is a protection.

Encapsulation

H.W Give example of 10 objects and define it in class of c++.

3 or 6 3.

For example - class = Geometric shape.

Triangle
Rectangle
Circle.

function ~~draw~~ draw
& Calculate area..
calculate perimeter.

(2) Object

Can inherit another object.

(6) Color.
Drawing pen.

Drawing pen will include the Color.
This is called inheritance.

Class Person,

Office member.

All office members are also person.

Page-234, chapter-5, Rajkamal.

16.10.2014

Home Work

See 8.3 VLSI and Integrated circuit design

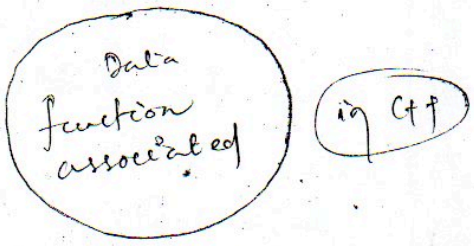
See 8.7 PCB layout design.

UML

Sect 7.3 introduction

Object Oriented Design

Data declaration
function declaration } seperately (in C)



Data and accessing function are kept together.
in a class. object is actual.

Data type are

Public — Usable anywhere
Protected — Used in inheritance.
Private — only internal.

Inheritance is something that permit inclusion of a object in another object.

Polygon
↓ inheritance
Rectangle.

: C++, Java are object oriented programming (OOP)

- Object Oriented Design (OOD)

UML - Unified Modelling Language.

UML is an OOD

- Modelled program will be depicted in figure.

www.omg.org,

Tutorial of UML: 3.0 or 2.0.

Training in C++. (in internet)

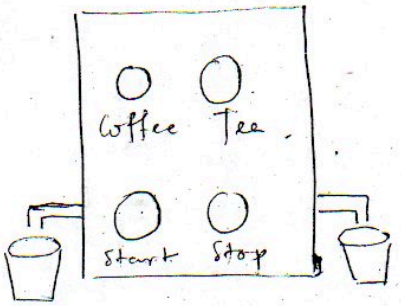
- Home work

① Give examples of 10 classes.

② Discuss of feature of embedded C.

Page No - 150, table.
Class representation table.

Example: Coffee/Tea Vending Machine.



6

Class. drink

{ int C, T, start, stop;

int c, T, start, stop;

if (c==1 && start==1)

cout

if Coffee_b = ON

if start_b = ON

then vend coffee.

if start = ON

stop is pressed

stop vending.

If coffee is ON

if $Tea_b is pressed

do Nothing.

If Tea_b = ON

if start_s = ON

then vend Tea.

if start = ON

stop is pressed

stop vending

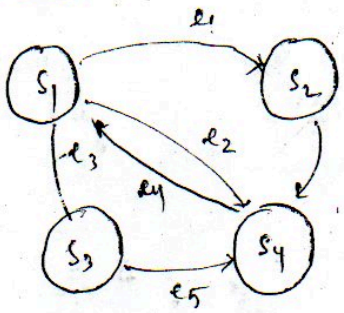
If Coffee+Tea is ON

if Coffee_b is pressed

do Nothing.

UML : State transition Diagram

Program has N number of states, M number of event.



finite state machine (FSM)

Based on the event the program will transit from current state to next state.

Millee model

Moore model.

~~Fcase~~

Switch (state)

Case of S1 switch (event)

case of

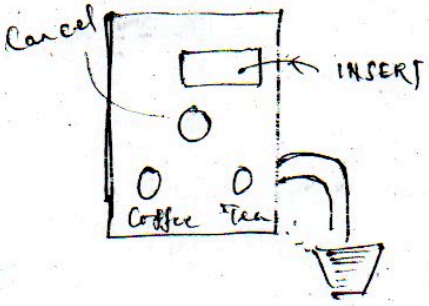
e1 : call f1()

e2 : call f2()

e3 : call f3()

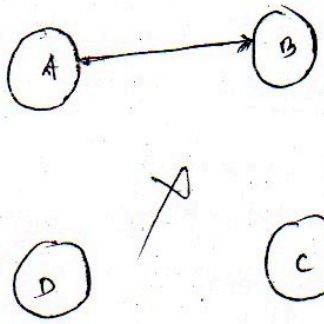
Study fig. 7.5 page 210.

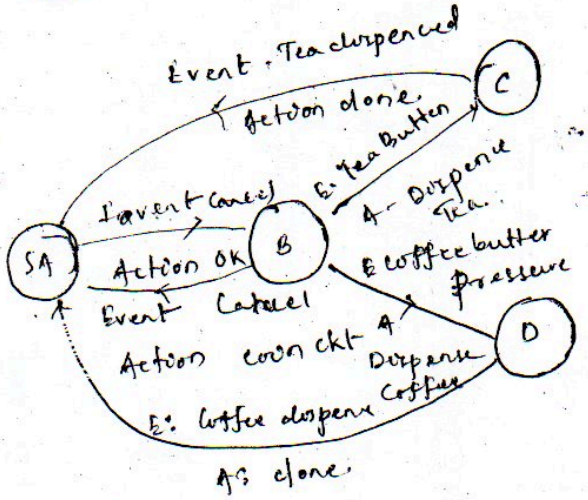
FSM for a coffee and Tea vending machine.



Machine has 4 states.

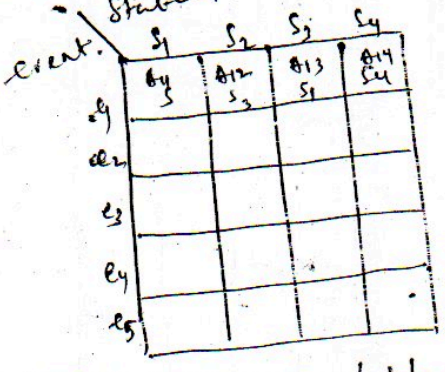
State-A wait for coin
State B coin inserted.
state C dispense tea.
State D Dispense coffee.





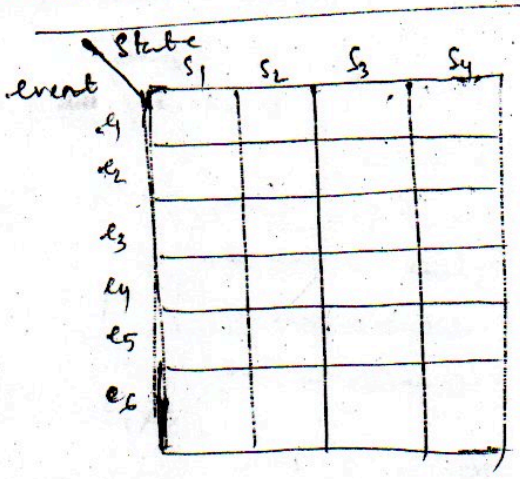
Coin operated telephone system: page 211, fig 7.6

FSM of coin operated



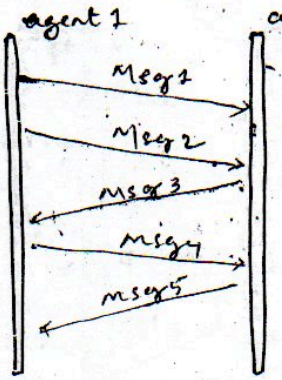
State transition table.

Sd- State transition table for previous Ques:

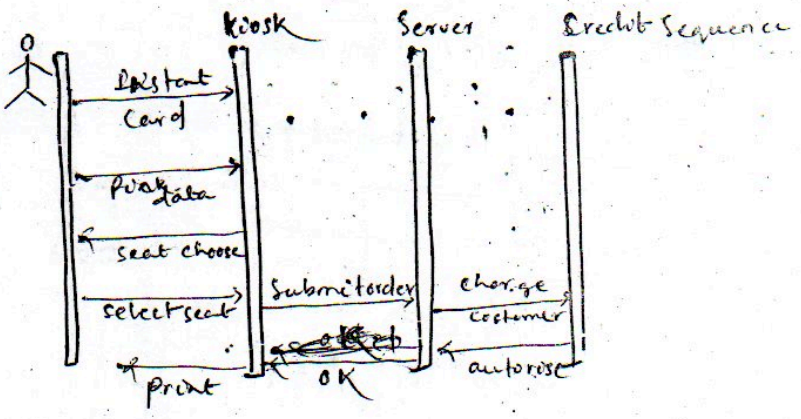


Message Sequence Diagram (MSD)

Two agents can communicate with each other by two ways message which are sequenced.

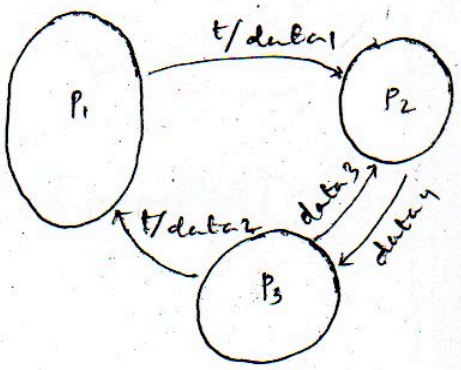


Railway ticket bus reservation system,



Data flow diagram : (DFD)

Process will receive and send data.



Data will flow one process to another process on to system.

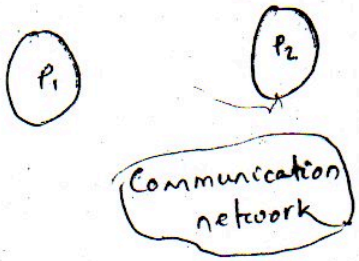
Send data

Receive data

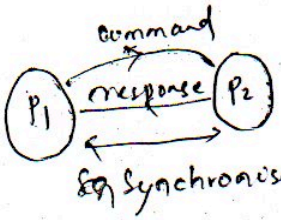
process data

and take action.

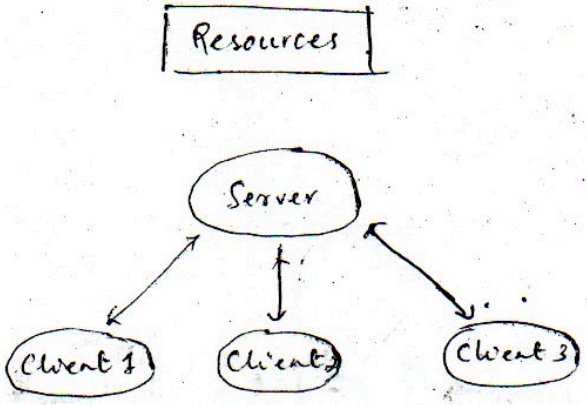
Collaboration diagram



P1, P2 may be on one complete or different complet.



for example
- client server



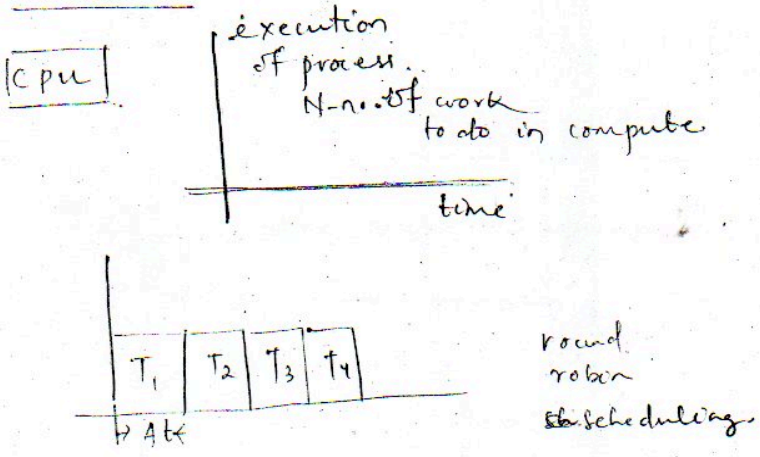
Read Chapter 6 / Page 273 (Rajkamal.)

Chapter-9 Embedded firmware Design and Development

Two approaches.

1. Super loop model
2. Operating system (OS) based approach.

Process / Task



ISR

→ ① Interrupt

② Process communication

③ pro

→ Numbers (0, 1, -2, 3)

Pre-emptive Scheduling.

When high priority task/process is ready.

Low priority task will be stopped, high priority task will resume

In super loop, there no OS.

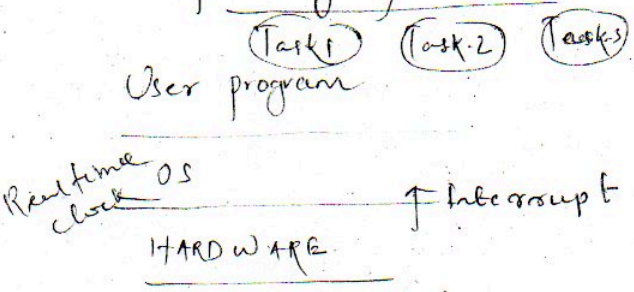
Linear coding of the program is done.

```
Task1()
Task2()
Task3()

while(1)
{
  Task1()
  Task2()
  Task3()
  Task4()
}
```

Ref sect. Gill Shibu K.V.

Operating System based



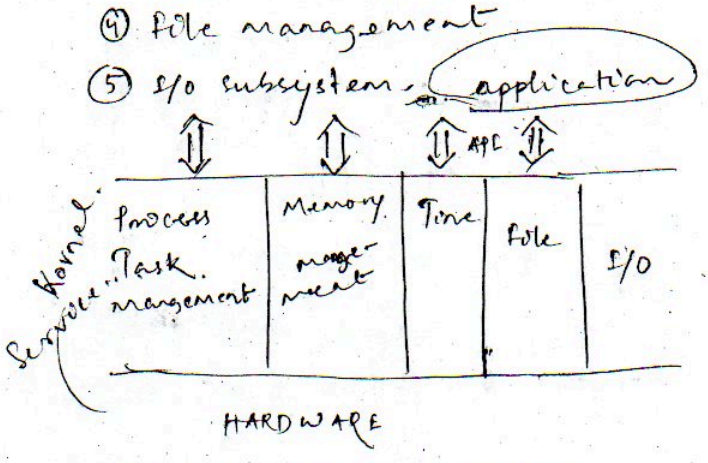
Task management
Time management
Interrupt management

Module - III

Real Time operating System

Functions of operating systems are

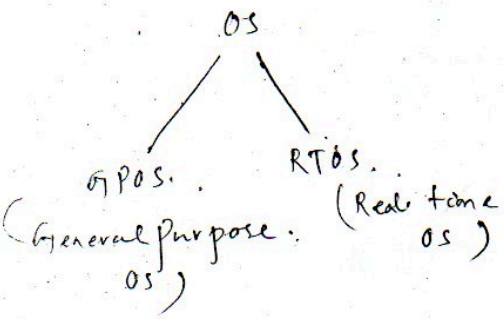
1. Process / Task management
2. Memory management
3. Time management
4. File management
5. I/O subsystem



One cannot access Kernel layer without API thus this layer is protected.

Large ~~system~~ service

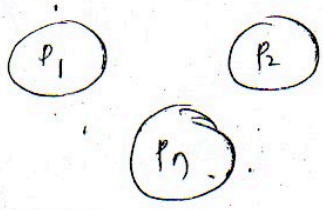
Signal service - MICRO KERNEL.



Ref -
Chapter - 10

Process / Task Management

Process / Task are code pieces running concurrently in application view and scheduled in kernel




```
While (1)
{
  Process..
Process start
{ while (1)
  {
   process..
process end
}
```

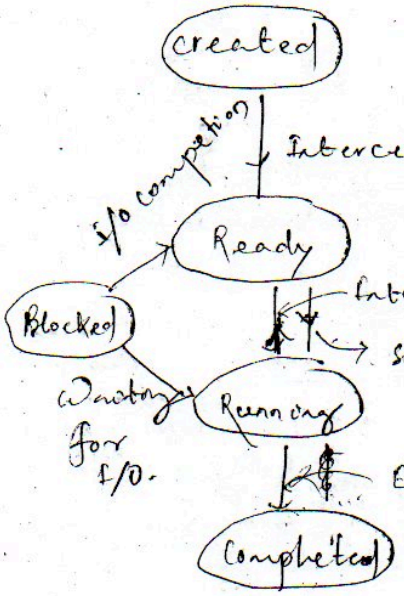
States of process.

Created.
Ready to run
Running
completed.
Blocked.

State transition diagram of process.

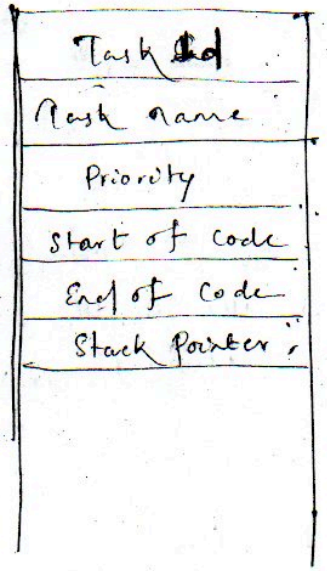
fig 10.6.

State Transition diagram.



Each process or task has a process / Task control block (PCB/TCB)

It is a piece of data used in management.



28.10.2014

Process and Task management:

PCB/ TCB.

create task (task id, TCB address, Name priority, code address)

Code address.

task id — integer

TCB address — memory pointer.

Name — String

Priority — Integer (0–15)

Code address — memory pointer to code.

run — task (task id)

It will put the task in ~~ready to run~~ READY TO RUN state

Suspend - task (task id).

It will put the task in ~~to~~ SUSPENDED or BLOCKED state.

delete task (task id)

This will delete task from memory.

```
Task-function
{
while (1)
   {
    task--
   }
}
```

Thread:-

It is a small task subjected to round robin scheduling.

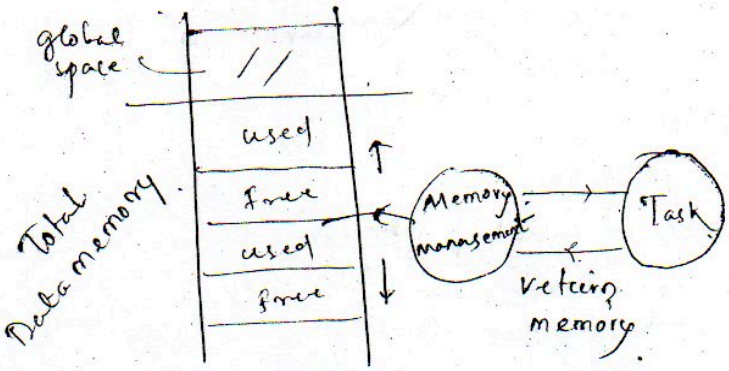
create - thread

run - thread

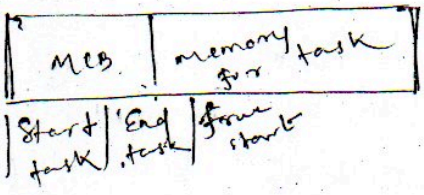
Stop - thread.

Context-lock () — Task scheduling is blocked.
Context-unlock () ↓ Task scheduling.

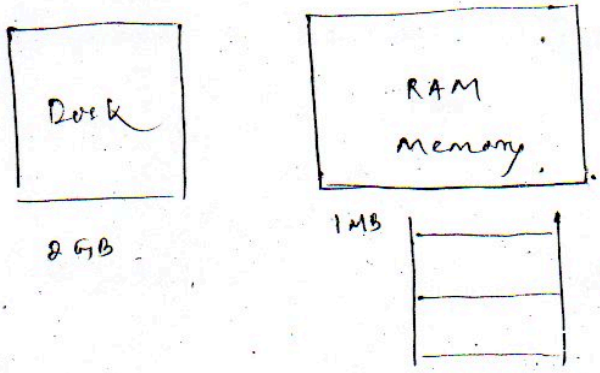
Memory Management



Garbage Management



create_memory
delete_memory
free - memory.



(2) Swapping virtual memory management

Virtual memory = 2 GB

Timer management

Create-timer (timer-id, timer_value

Stop-timer()

run-timer()

Sleep() - Sleep definite

or

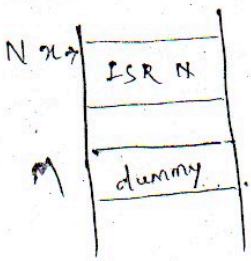
Sleep indefinite.

Code()
Sleep(10)

Interrupt management:

Install-ISR (ISR-address, N)

Delete-ISR (Dummy code/address N)

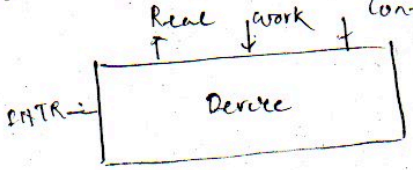


Device Management:

29.10.2014

I/O - Subsystem / Device management

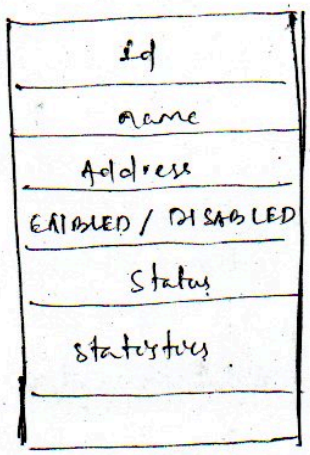
A generalised I/O device



We have to do device management from operating system like PCB, we will a data block for device called DCB.

agement

DCB



open-device()
close-device()
enable-device()
delete-device()
control-device()

read
write.
in
out

Initialize
control.
configure
Send-command
get-status.

given a device
This cell should be
coded to use in OS. and application
known as device driver.

Examples.

Printer driver in PC
This is loaded and used.

Chap-10
finished

The device 8255

Command

IN

Mode 0 :

Mode 1 :

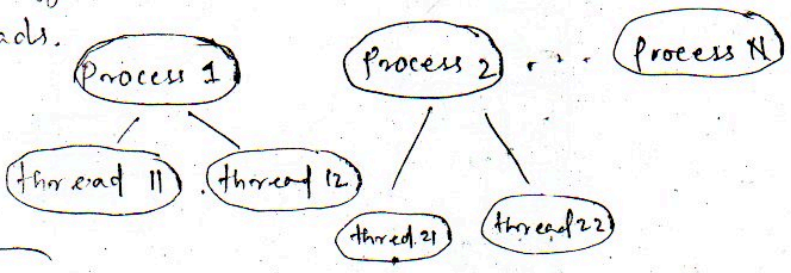
Mode 2 :

Inter process Communication

Ref
Chapter 7
Raj Kamal

In application where multiple process.

threads.



One should have Mechanism to talk between the process.

One should have mechanism to share data called DCB.

Data sharing

(1) Global data share
(2) Semaphone.
(3) Counting Semaphone.

Semaphone
• T/F = 0, 1

get - global - address. (name, ~~re~~address)

~~get~~ Create - Semaphone
get - Semaphone
delete - semaphone

In counting semaphone value of semaphone (0 - N)

Process - Communication

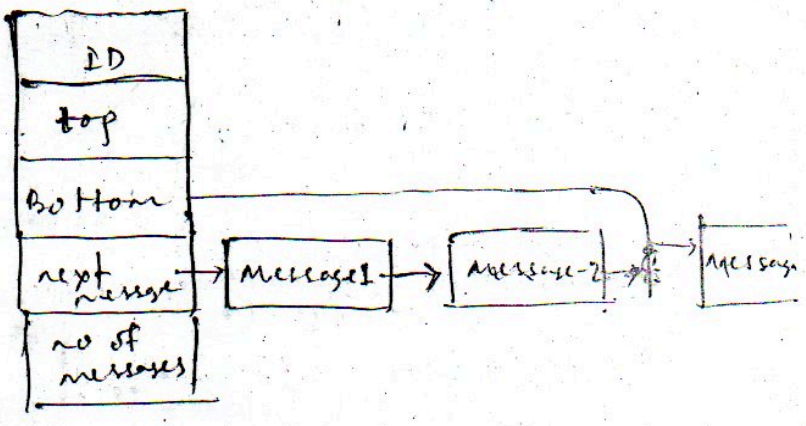
(1) Mail box
(2) Pipe
(3) Socket

Message Queue 30.10.2014

Read below it is linked

Must see for exam

Interprocess Communication



One can put a

(1) Message queue
(2) Mail box
(3) Pipe
(4) Socket
(5) RPC

(1) Message queue

1. It is called
2. Message in a queue having top and bottom indication
Message queue has an id

One can put a message in the linked list one pick a message from linked list.

Ref. Jain Rajkamal.

Create - Q Send - Q Queue - Q
Post - Q Flush - Q
Accept - Q Top - Q

A queue is., FIFO ~~first~~ first in first out.

Pend-Q: It will wait on a to read the message

flush-Q: delete all messages and statistics on the Q..

Top-Q: Return the pointer of top of the Q.

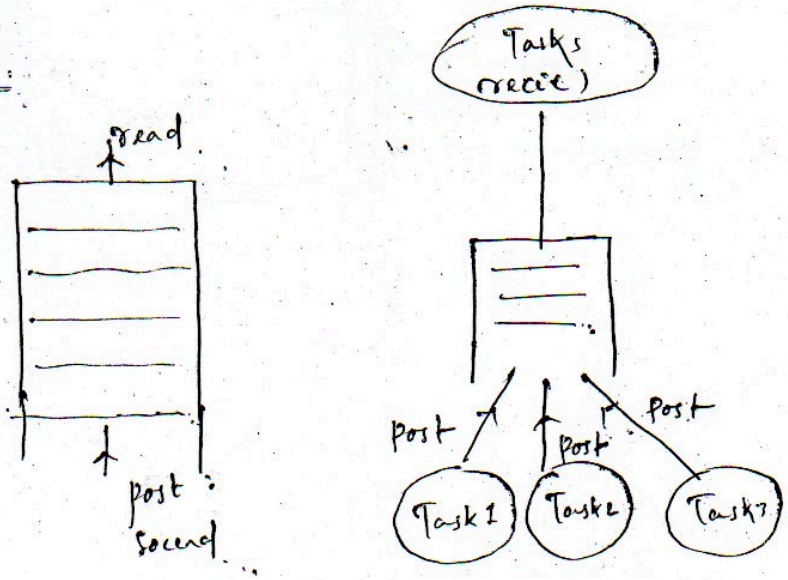
A particular message present in the Q or deleted.

7.19 Sample program Rajkamal

Query -Q

A particular message is deleted.

② Mail box.



create mail box

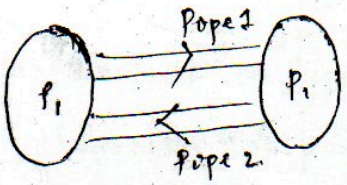
post — mbox

wait - mbox

Accept - mbox or read - mbox.

Query - mbox

Pipe

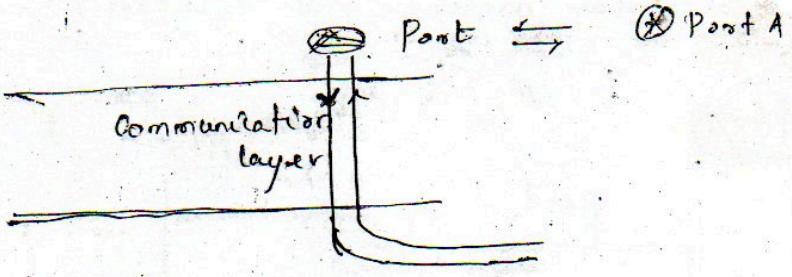


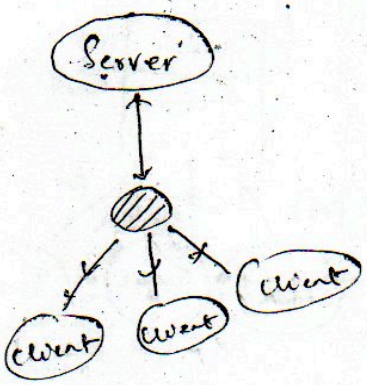
Typically used when large suitable for large data. file data.

Pipe - create
Pipe - open
Pipe - write
pipe - read
pipe - connect
pipe - error.

Ref. 7.14 Rajkamal.

Socket





ES-P1