UNIT : IV PROTECTION

Page No.
Date

1) Need of protection :

→

- Problem may occur in multitasking operating systems or multi-user systems when two or more users attempt to read and change the content of a memory location at the same time.
The section of program where the value of a variable is being read and changed must be protected from access by other tasks until the operation is computed.

- Another region that requires protection is Operating System code. The incorrect address in a user program may cause program to write over the critical sections of operating system corrupting the OS codes and data areas.
The system then 'locks-up' and the only way to get control again is to reboot the system. In a multitasking system this is intolerable, so several methods are used to protect the OS.

Page No.
Date

eg - Border

① Explain need of protection mechanism in 80386?

→ - Promblem may occor ~~when~~ in multitasking operating system or multi user systems when two or more user attempts to read and change content of memory location at some time.

- The main purpose of protection Feature of 80386 is to have multi tasking with each task having its Protected memory. Protectection ensure that multiple task cannot acess the code and data of other task.

- In case of multitasking there is chance or possibility that a task can accessed code and data of another task hence to avoid this protection is implemented.

2) Explore five aspects of protection applied in segmentation.

→

- The protection is implumented internally in memory management system, both in segment as well as page translation mechanism.

- Protection in 80386Dx has five aspects:-

(i) Type checking:

- segmentation can incorporate type checking mechanisms to ensure that data accessed within a segment is of correct type.
- As mentioned, W (writeable), R (Read), A (accessed), etc bits from type field specifies the usage of segment and restrict segment for particular use only.

(ii) Limit checking:

- 80386Dx uses limit field of segment descriptor to prevent programs from addressin outside the segments.
- for all type of segments, the value of the limit is one less than size (expressed in bytes) of the segments.
- It ensures that accessed to data within segment do not exceeds predefined boundaries thereby preventing buffer / stack overflows

(iii) Restrictions of Addressable Domain:
- Segmentation can restrict the addressable domain of each segment, limiting the memory location that can be accessed within segment.
- This helps prevent unauthorized access to sensitive data or system resources by ensuring that segments can access only allotted memory location.

(iv) Restrictions of Procedure Entry points:
- segmentation can restrict the entry points to procedure or functions within each segment.
- By controlling which procedures can be invoked from specific segments, ensuring that only authorized code executions paths are allowed within each segment.

(v) Restrictions of Instruction set:
- segmentation can also restrict the set of instructions that can be executed within each segment.
- This involves specifying which CPU instruction are allowed to be executed within a segment based on intended functionality and security requirements of segment.

* Rules of protection :-

1] Access control :-
- Each ring has it own set of permission that access to system resource such as memory , IO port's

2] privileage instruction
- It canatain instruction cans only be executed to high level (Ring 0 & sometimes Ring 1) while lower privilage level from executing them.

3] Memory protection
- Each process allocated its ow virtual space and it enforces memory protection to prevent unorthorized access to other processes memory.

4] Exception handling :
- processor raises exception for illeagal or privilaged instruction exception in lower privilage level's ensuring that the system resemains stable & secure.

3) Write a short note on CPL, DPL, RPL, EPL, IOPL.

→

Descriptor privilege Level (DPL):

- Descriptor contains field called the Descriptor Privilege Level (DPL). It is least privilege level at which a task may access that descriptor and the segment associated with that descriptor
- It is contained in access right byte of descriptor of the segment.

Current Privilege Level (CPL):

- The 80386DX stores the descriptors in the internal cache for currently executing segments. Previlge levels of such descriptors are referred to as Current Privilege Level (CPL).
- This privilge level is also called as Task privilege level.
- It specifies privilge level of currently executing task.

Request Privilge Level (RPL):

- Selector contains field called the Requester's Privilge level (RPL). The RPL is intended to represent privilge level of proudure that originates a selector.
- RPL is the two least significant bits of selector.

Page No.
Date

Effective privilege Level (EPL):

- When access to a new memory is desired, an Effective Privilege Level (EPL) is computed. This is the least privilege of CPL and RPL.
- EPL is defined as

EPL = max{ RPL, CPL } (numerically)
∴ thus task become least privilge.

- for example,
  if RPL = 2
  CPL = 1
  then EPL = 2 → task become least privilge.

Input/output privilge Level (IOPL):

- In IOPL mechanism, for execution of IN, INS, OUT, OUTS, CLI and STI instructions, the CPL of procedure or task must be the same or a lower number than the number represented by the IOPL bits. (CPL ≤ IOPL)

- Any attempt by less privileged procedure to use these instructions results in general protection exception. These instruction are called sensitive instructions because they are sensitive to IOPL.

- Thus each task can have different IOPL, task can change IOPL. But only procedure executing at PL0 can change it otherewise it remains unaltered and do not result in any exception.

4) Explain how segment and page level protection function in combination.

→

- When paging is enabled, the 80386 first evaluates the segment protection, then evaluates the page protection. If processor detects a protection violation at either the segment or page level, the requested operation cannot proceed; a protection exception occurs instead

- It is possible to define a large data segments which has some sub-units that are read only and other subunits that are read-write. In this case, the page directory (or page table) entries for the read only subunits would have the U/S and R/W bits set to X0, indicating no write rights for all pages described by that directory entry.

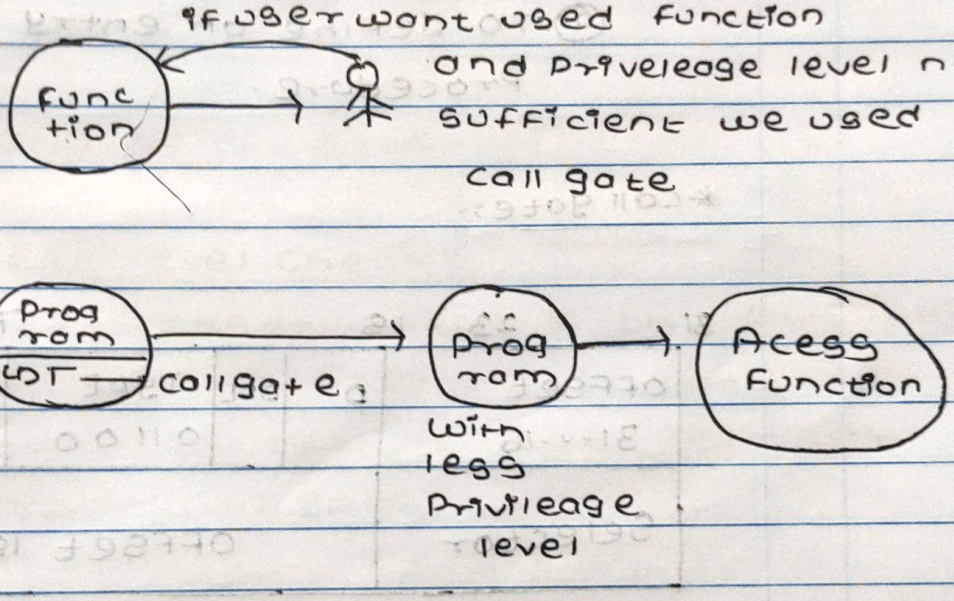
Do not write your name or seat no. below this line

Q. No.						Q. No.					

प्र. क्र.
Q. No.

Q4] a] What is call gate ? Explain how it is used in calling function with higher privilege level.

→



1. Privilege level check
2. Code segment loading
3. Stack setup
4. Control transfer

*Call gate :-

- A call gate is special type of descriptor that allow program running at certain level to call a procedure or function located in a different privilege level.

Do not write your name or seat no. below this line

Q. No.						Q. No.					

प्र. क.
Q. No.

- Call gate acts as an interface between code segment at different Privilege level.
- Call has mainly two primary function:-
  1. To specify privilege level of entry
  2. To define an entry point of Procedure.

*Call gate:-

31 ... 23	15	7 ... 0			
Offset 31...16	P	DPL	TYPE 01100	000	(WC) Dword count
Selector	Offset 15...0				

Call gate contain two things:

1] Selector:-
- Points to descriptor for segment where Procedure is actually loaded

2] Offset:-
Offset of called Procedure on its segment.

3] DL:-
It specify the privilege level to acess code segment.

Do not write your name or seat no. below this line

Q. No.						Q. No.					

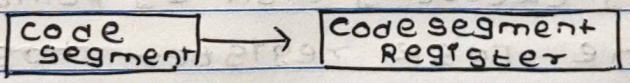
प्र. क.
Q. No.

4] Descriptor Type :-
It specify the ~~de~~ type of descriptor wherer the gate is call gate or task gate.

* Working of call gate :-

① Privilege level check :-
- The Processor check if the privileage level of calling program or task is sufficient to execute code segment specified in call gate.

② Code segment loading :-
-After checking privileage level the Processor loads the codesegment specified in call gate into code segment register (CS).



③ Stack setup :-
The processor setup the stack segment (SS) and stack pointer (SP) values in call gate.

Do not write your name or seat no. below this line

Q. No.						Q. No.					

प्र. क्र.
Q. No.

(4) Control transfer :-

- Finally the Processor transfers control to ~~other~~ offset specified in call gate.

2] Explain role of TASK register in multitasking and instruction used to modify and read TR.

→

- multitasking is ability of computer to run more than one program or task at some time.

-

*Role of TASK Register in multitas king :-

- The TASK Register (TR) identify curr enty executing task by pointing to TSS.
- The TASK register has both visible portion which can read and change by instruction and invisible portion which cannot read by on instruction.
- The selector in visible portion is used to ~~select~~ specify TSS descriptor in GDT.

* Explore the role of various field's in page level protection.

- Page-level protection :-
  when the processor is executing at supervisor level all pages are both readable and writable. when the processor is executing at user level only page's that belong to user level and are marked for read/write acess are writable page that belong to supervisor level are neigther readable nor writable from user level.

31 — 12	11 — 7						0
Page Frame Address 31--12	AVAIL	0	D	A	0	U/S	R/W	P

protection field's of page table entries.

- Two kind's of protection are related to pages :
  1. Restriction of addressable domain
  2. Type checking.

1] Restriction Addressable domain
- The concept of privilege for pages is implemented by assigning each page to one of two level's.

1. supervisor level (U/S=0):
   - for the operating system and other systems software and related data.

2. user level (U/S=1):-
   - for applications procedures and data.

- The current level (U or S) is related to CPL If CPL is 0, 1 or 2 the processor is executing at supervisor level
- If CPL is 3 the processor is executing at user level.
- when if processor is executing at supervisor level all page's are addressable
- but when the processor is executing at user level only pages that belong to the user level are addressable.

2] Type checking :-
- At the level of page addressing two type's are defined.
  1. Read-only access (R/W=0)
  2. Read/write access (R/W=1)

- when the processor is execution at supervisor level all page's are both readable and writable.
- when the processor is executing at user level only page's that belong to user level are marked for

read/write access are writable.
- page's that belong to supervisor level are neither readable nor writable from user level.

Page No.
Date

protech

eg- Border

① Explain need of protection mechanism in 80386?

→ - Problem may occur ~~when~~ in multitasking operating system or multi user systems when two or more user attempts to read and change content of memory location at some time.

- The main purpose of protection Feature of 80386 is to have multi tasking with each task having its Protected memory. protection ensure that multiple task cannot acess the code and data of other task.

- In case of multitasking there is chance or possibility that a task can accessed code and data of another task hence to avoid this Protection is implemented.

② Define Function of type checking and limit checking?

→

* Type checking :-

• Whenever a selector is loaded into segment Register, the corresponding target Descriptor is copied From Descriptor table and looded into descriptor cache.

- The Descriptor indicate type of segment that is compared with segment Register in which selector is store.
- Acess is granted if type match if not Interrupt is occurs.

Type Field of descriptor specifies
① Type of descriptor
② Intended usage of segment.

- The W, R, C (Conforming), A (Accessed), E (Expanded Down) bits From type Field specify the usage of segment.

* Limit checking :-
- The 80386 DX uses limit field to Prevent Program from addressing outside the segment.
- The limit Field indicate size of segment which can vary From 1 byte to 4GB.
- The offset address which is desired to be accessed by an instruction should be within this limit then only access is granted.

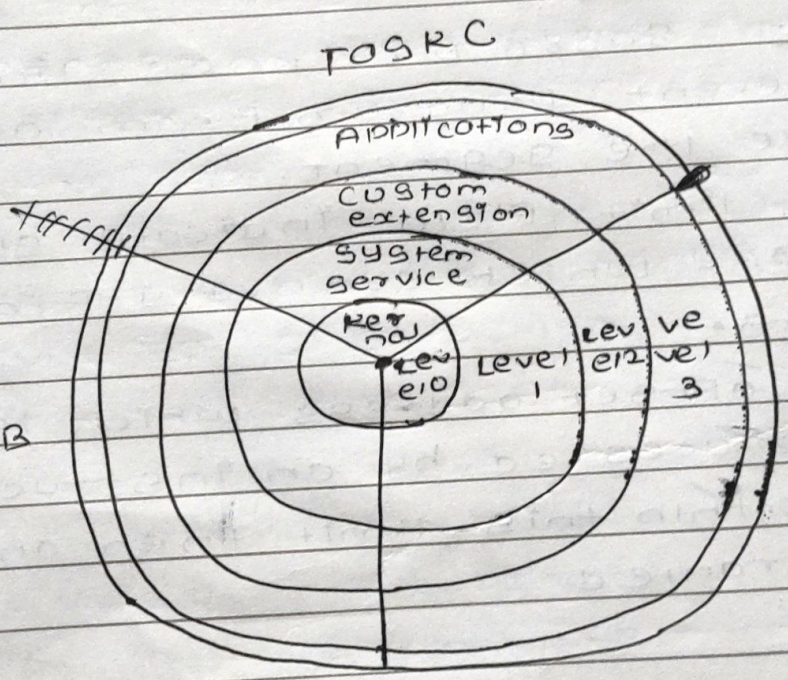
- If not general protection exception may occur :-
① For byte acess (8 bit), offset address must be L = limit
② For word acess (16 bit) offset address must be L = (limit - 1)

(5) For double word (16 bit) the offset address must be $2^{n}-3$

(3) Explain different privilege level?

→

- The 80386 has four level of protection which are optimized to support of a multitasking OS to isolate and protect user program from each other.
- Four protection level are numered from 0 to 3. whereas 0 is high privilege level and 3 is least privilege level.



① DPL :- (Descriptor Privilege level)
- Descriptor contain field called Descriptor privilege level.
- It is least privileged level at which task may acess that descriptor and segment associated with that descriptor.
- It is contain in acessed Right byte of descriptor of segment.

② CPL :- (Current Privilege level) :-
- The 80386DX stores the descriptor in internal cache for currently executing segment privilege level for such descriptor is called current privilege level.
- This is also known as task privilege level.
- It specify privilege level for currently executing task.
- A task CPL can only be changed by control transfer through gate descriptor.

③ Requestor privilege level (RPL) :-
- Selector contain the field called Requestor privilege level. The
- RPL is intended to represent privilege level of Procedure that originates a selector.
- RPL is two least significant bit of selector.

④ Effective Privilege level (CRPL):-
- when accessed to new memory segment is desired the EPL is computed.
- EPL is defined as:-
EPL = max(RPL, CPL)
- For example
IF RPL=2 and CPL=1, EPL=2

④ What is privileged Instruction? Explain its significance with examples?
→ There are 19 Privileged Instruction supported by 80386. privileged instruction are those that effect the segmentation and protection mechanism, alter interrupt Flag, or perform Peripheral I/O.

These are divided into two group:-
① Privileged Instruction
② IOPL-sensitive Instruction

* Privileged Instruction:-
- This instruction effect the system data structure are come under 1st group.
- The instruction under this group must be executed when CPL is 0; otherwise 80386 generate exception.

Q. ④ what is privileged instruction & Explain its significance with example &

→ ~~There~~ are

- the privileged level instruction are type of instruction use to give the the privilage to perticular item.
- there are 19 privileged instruction supported by 80386 privileged instruction ~~SUDO~~ are those that affect the segmentation and protection mechanism after interrupt flag or perfom peripha I/O.

- these are divided into two group :-
  ① Privileged Instruction
  ② IOPL - sensitive instruction.

* privileged Instruction.
- This instruction affect the system data structure are come under 1st group
- The instruction under this grouped most be executed when CPL is 0 ; otherwise 80386 generated exception.

* Instruction :
① HLT → Halt the processor
② CLTS → clear task switched flag
③ LGTS, LIDT, LLDT → Load GDT, IDT, LDT
④ LTR → Load task Register
⑤ LMSW → Load machine status word
⑥ mov CRn, REG / mov REG, CRn
- move to / from control Register

⑦ mov CRn, REG / mov REG, DRn
- move to / from debug register

⑧ mov TRn, REG / mov REG, TRn
- mov to / from test Register.

* IOPL sensitive instruction :-
- Here IOPL field in flag register defines right to use I/O related instruction Hence instruction from this group is called sensitive instruction.

* Instruction :	Action
- In, INS	Input data from I/O port
- Out, outs	output data to I/O port
- STL	Enable interrupt
- CLI	disable interrupt.

* priviledged level instruction

1. HLT (Halt)
   - The 'HLT' instruction halts the CPU until next external input is received
   - e.g : HLT

2. CLTS
   - clear Task-switched Flag
   - The 'CLTS' instruction clears the Task-switched (TS) flag in the control Register CR0
   - e.g : CLTS.

~~3. LDTR~~
   ~~- Local Descriptor Table Register.~~
   ~~- The LGT~~

3. LGDT
   - ~~Locald~~ Load Global Descriptor Table
   - The 'LGDT' instruction loads the base address and limit of the Global Descriptor table (GDT) from memory into the GDTR register.

4. LIDT
   - Load Interrupt Descriptor Table Register.
   - The 'LIDT' instruction loads the base address & limit of the interrupt Descriptor Table (IDT) from memory into the IDTR. register.
   - e.g : LIDT [address]

5. LTR (Load Task Register)
- The 'LTR' instruction loads the Task Register (TR) with a segment selector that point to a task segment (TSS)

6. LLDT
- Load Local Descriptor Table Register
- The LLDT instruction load Descriptor Table Register (LDTR) with a segment selector that point to a local Descriptor Table (LDT).

7. LMSW
- Load Machine status word
- The LMSW instruction loads the Machine status word modify certain bits of the CR0 Register.

8. IN
- (INput from port)
- The 'IN' instruction read a byte, word or double word from an I/O port into a CPU register.

9. Out
- (output to port)
- The 'out' instruction writes a byte word or double word from a CPU register to an I/O port.