UNIUNEA EUROPEANĂ
GUVERNUL ROMÂNIEI
Instrumente Structurale
2007-2013

Platformă de e-learning și curriculă e-content pentru învățământul superior tehnic

Programare în limbaj de asamblare

14. Tabela vectorilor de întrerupere.

Tabela vectorilor de întrerupere

Această tabelă reprezintă legătura dintre codul întreruperii şi procedura definită pentru a servi întreruperea asociată cu acel cod. Câțiva din vectorii cei mai prioritari sunt rezervați de INTEL pentru funcții dedicate. Utilizatorul nu poate folosi nici unul din primii 32 de vectori. În modul real, această tabelă ocupă 1 Ko de memorie (pentru fiecare tip de întrerupere din cele 256 posibile este rezervat un pointer, adică un dublu cuvânt), care conține adresa procedurilor de tratare a întreruperilor. Adresa este memorată astfel: cuvântul cel mai semnificativ conține adresa de bază a segmentului, iar cuvântul mai puțin semnificativ conține offsetul, acestea fiind memorate la adrese succesive în ordinea următoare: cuvântul mai puțin semnificativ (offsetul) şi apoi cuvântul mai semnificativ (selectorul de segment).

În tabelă sunt prezentați vectorii de întrerupere pentru modul real.

Excepția de eroare la împărțire (nivel 0) apare în cazul în care câtul este prea mare (mai mare decât destinația) sau dacă se încearcă împărțirea la zero, utilizând o instrucțiune DIV sau IDIV. Se salvează în stivă adresa instrucțiunii (CS:IP) care a generat eroarea; registrele DX şi AX rămân nemodificate.

Întreruperea „single step“ (nivel 1) apare după fiecare instrucțiune dacă TF=1; evident că TF este pus pe 0 după intrarea în această procedură pentru a preveni recursivitatea infinită. Valoarea salvată în stivă pentru CS:IP va face referire la următoarea instrucțiune.

Întreruperea NMI (nivel 2), apare dacă se primeşte un semnal extern pe pinul NMI; în mod normal, NMI este utilizat pentru a implementa proceduri de cădere de tensiune sau repornire. Valoarea salvată în stivă pentru CS:IP va face referire la primul octet al instrucțiunii întrerupte. Dacă NMI apare în timp ce se execută o servire curentă NMI, prezența ei va fi salvată pentru o servire ulterioară, după execuția primei instrucțiuni IRET.

Întreruperea de suspendare (nivel 3) apare în urma execuției acestei instrucțiuni, INT 3. Această instrucțiune este folosită pentru implementarea de programe de depanare, deoarece necesită un singur octet de cod, şi poate substitui foarte uşor orice prim octet de instrucțiune. Se va salva în stivă adresa următoarei instrucțiuni de executat.

Excepția de depăşire detectată de instrucțiunea INTO (4) apare la execuția acestei instrucțiuni dacă indicatorul OF=1; instrucțiunea va salva în stivă adresa următoarei instrucțiuni.

Excepția de depăşire de domeniu detectată de instrucțiunea BOUND (5) este cauzată de execuția instrucțiunii BOUND dacă indexul vectorului specificat este invalid în raport cu limitele date ale vectorului. Un program poate utiliza instrucțiunea BOUND pentru a verifica dacă indexul cu semn al unui vector se încadrează în limitele definite într-un bloc de memorie. Valoarea salvată în stivă pentru CS:IP va face referire la primul octet al acestei instrucțiuni.

Excepția generată de cod de operație invalid (6) se generează dacă se încearcă execuția unui cod de operație invalid. Excepția este detectată numai la încercarea de execuție a unui cod inexistent, nu şi la citirea anticipată a acestui cod în coada de instrucțiuni. În modul real, majoritatea instrucțiunilor corespunzătoare modului protejat sunt considerate invalide şi nu trebuie utilizate. De asemenea, această excepție poate să apară dacă adresa specificată de anumite instrucțiuni (LES, LDS, BOUND, LIDT) specifică un registru în loc de o locație de memorie. Această excepție poate să mai apară şi dacă au fost utilizate prefixe redundante în fața instrucțiunii, astfel încât lungimea totală a instrucțiunii depăşeşte 8 octeți, la 286, sau 16 octeți la 386/486. În stivă se va salva adresa primului octet al instrucțiunii invalide.

Funcția	Număr întrerupere	Instrucțiuni legate de întrerupere	Returnează adresa instr. ce a generat excepția
excepție eroare împărțire	0	DIV, IDIV	da
întreruperea 'Single Step'	1	toate	-
întreruperea NMI	2	toate	-
întreruperea de breakpoint	3	INT 3	-
excepție depăşire detectată de instr. INTO	4	INTO	nu
excepție depăşire domeniu detectată de instr. BOUND	5	BOUND	da
excepție cod instrucțiune invalid	6	orice cod nedefinit	da
excepție extensie procesor nu este disponibilă	7	ESC sau WAIT	da
excepție limită tabela de întreruperi prea mică	8	LIDT	da
excepție depăşire segment extensie procesor	9	ESC	da
excepție depăşire segment	13	toate instr. cu referire la memorie	da
excepție eroare extensie procesor	16	ESC sau WAIT	-
întreruperi rezervate	10-12, 14, 15, 17-31		-
întreruperi definite de utilizator	32-255		-

Vectorii de întrerupere rezervați şi dedicați

Excepția pentru extensie procesor inexistentă (7) poate să apară la execuția unei instrucțiuni ESC dacă biții de stare din MSW arată că funcțiile extensiei procesor sunt emulate prin soft, sau la execuția instrucțiunilor ESC, WAIT, dacă MP=1 şi TS=1, adică în situația în care a avut loc o comutare de task, pentru a verifica în ce măsură contextul coprocesorului corespunde contextului curent. Se salvează în stivă adresa primului octet al acelei instrucțiuni.

Excepția pentru limita tabelei de întreruperi prea mică (8) va apărea dacă limita tabelei de întreruperi a fost modificată de instrucțiunea LIDT de la valoarea 3FFH. Valoarea salvată pentru CS:IP va face referire la primul octet al instrucțiunii ce a cauzat întreruperea, sau cea care era gata de execuție înaintea apariției unei întreruperi externe.

Excepția pentru depăşirea segmentului de către extensia procesor (9) va apărea dacă un operand din memorie solicitat de extensia procesor (prin instrucțiunea ESC) nu încape într-un segment; se va salva adresa instrucțiunii ce a cauzat întreruperea.

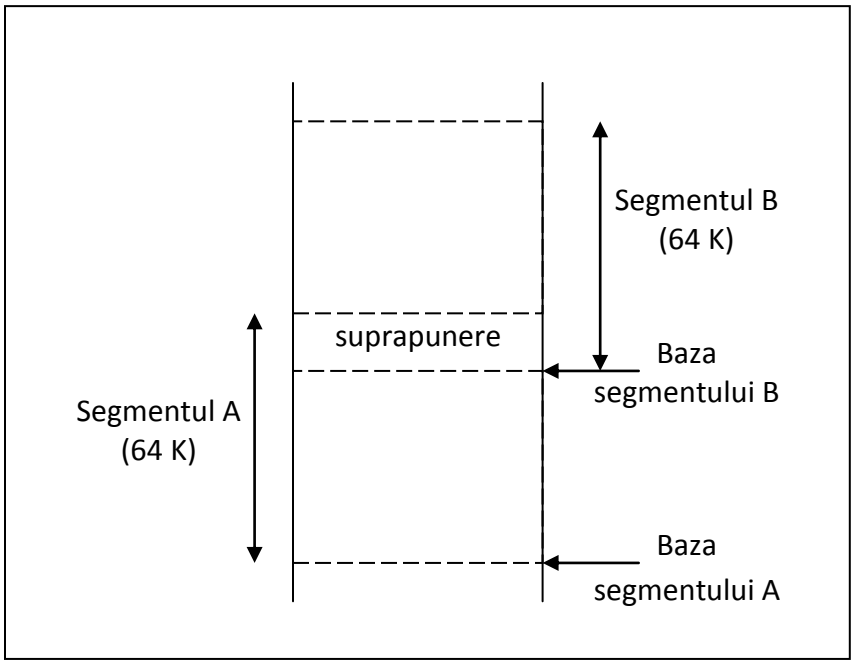
Excepția pentru depăşirea segmentului (13) apare dacă un operand nu încape într-un segment; se salvează adresa instrucțiunii ce a cauzat întreruperea.

Întreruperea generată de eroare la extensia procesor (16) apare după execuția unei instrucțiuni aritmetice care cauzează o eroare, semnalată pe linia ERROR\, care este testată la execuția instrucțiunilor ESC, WAIT. Ea poate să apară numai când se execută instrucțiunea WAIT sau ESC următoare celei care a generat eroarea. Valoarea salvată în stivă pentru CS:IP va face referire la primul octet al instrucțiunii ESC sau WAIT executată ulterior. Adresa

instrucțiunii numerice eronate este salvată în extensia procesor.

Valoarea salvată în stivă pentru CS:IP va include şi toate prefixele care preced instrucțiunea respectivă.

În modul real nu se realizează verificări de limită sau de acces. Toate segmentele pot fi citite, scrise sau executate şi au limita FFFFH. Segmentele incomplet utilizate se pot suprapune, dar în modul protejat de adresare virtuală, programele care adresează un segment (B) dintr-un alt segment (A) devin incompatibile.



Accesarea unui segment din cadrul altui segment

Întreruperi în modul protejat

În modul protejat sunt detectate mult mai multe condiții de eroare care generează o întrerupere internă. Programul utilizat pentru servirea întreruperii poate fi executat în contextul task-ului care a generat întreruperea, sau poate fi un task separat; alegerea depinde de funcția de realizat şi de nivelul de izolare dorit.

Pentru acest mod este definită o tabelă de descriptori de întrerupere IDT – (Interrupt Descriptor Table), care defineşte procedurile pentru toate cele 256 de întreruperi. Această tabelă este referită de un registru intern al procesorului IDTR – Interrupt Descriptor Table Register – care conține adresa de bază a tabelei (24 biți) şi limita acesteia (16 biți). Registrul IDTR este încărcat de instrucțiunea LIDT, de codul executat pe nivelul 0 de privilegiu, de către rutina de inițializare a sistemului. Tabela poate fi plasată oriunde în memorie (figura următoare). Fiecare intrare în tabela IDT conține un descriptor de poartă (gate), format din 4 cuvinte (8 octeți), care conține referința la rutina respectivă. În tabela IDT sunt permise doar trei tipuri de porți: întrerupere (interrupt gate), capcană (trap gate) şi task (task gate). Primele două prelucrează întreruperile în cadrul aceluiaşi task, în timp ce a treia realizează o comutare de task. Porțile task din IDT sunt identice cu acelea din GDT (descriptorii sunt asemănători celor de sistem) şi operează în acelaşi mod. Diferența dintre primele două este că prima pune indicatorul IF pe zero

(dezactivează întreruperile), în timp ce a doua lasă indicatorul nemodificat. IDT nu necesită toate cele 256 de intrări; registrul limită (16 biți) un număr de intrări mai redus decât numărul total.

Intrările neutilizate vor avea zero în octetul de drepturi de acces. Accesul în afara limitelor sau la un descriptor eronat va genera o eroare de protecție generală (GP), cu un cod de identificare a erorii (a vectorului IDT invalid). Întreruperile 0÷31 sunt rezervate de INTEL, astfel că limita IDT trebuie să fie cel puțin 255, pentru numărul minim de intrări.

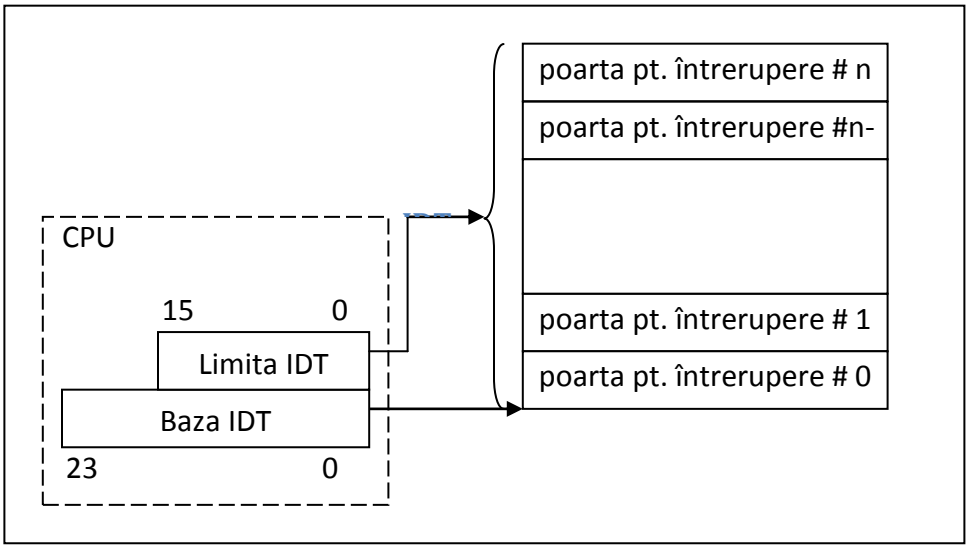
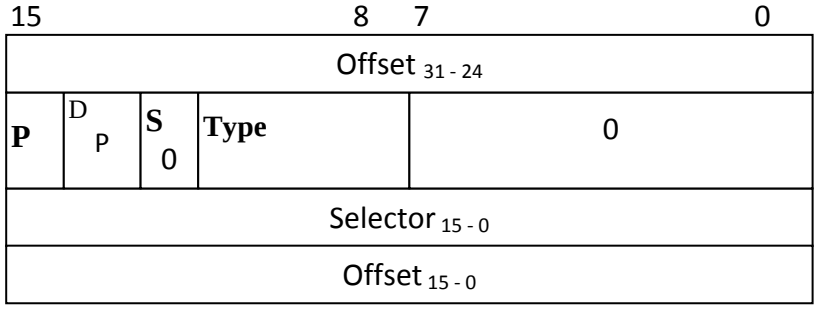


Tabela de descriptori de întrerupere (IDT)

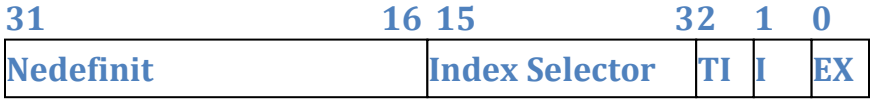
O poartă task poate fi invocată de o întrerupere/excepție; starea maşinii se salvează în TSS-ul curent şi se încarcă o nouă stare din TSS-ul asociat porții task. Astfel, o întrerupere poate avea propriul său spațiu de adresă (LDT). O comutare de task durează mai mult decât un transfer de poartă, dar oferă avantajul separării față de task-ul curent. Descriptorul pentru aceste porți are forma din figură:



Structura unui descriptor de portă întrerupere/capcană

Caracteristicile porților de întrerupere/capcană sunt asemănătoare celor de apel (call). Ele pot face referire la segmente de cod la anumite niveluri de privilegiu, sau conform regulilor de privilegiu; spre deosebire de porțile de apel, acestea nu conțin câmpul „contor“. Revenirea din întrerupere se face cu instrucțiunea IRET, care reface registrul indicatori şi segmentul de stivă. Dacă NT a fost 1 (în registrul indicatori) se va realiza o comutare de task la TSS-ul original. La

apariția unei excepții de eroare se salvează în stivă, după indicatori şi CS:EIP, şi un cod de eroare (de un cuvânt). Programatorul trebuie să extragă orice cod de eroare (generat de poartă şi pus în stivă) din stivă, înainte de revenirea din procedura de întrerupere. Unele excepții determină un transfer al controlului prin IDT; altele determină depunerea unui cod de eroare în stivă, cum ar fi: 8 (Double Fault), 10 (TSS invalid), 11 (NP), 12 (eroare stivă), 13 (GP), 14 (PG) şi 17 (AC). Codul de eroare este 0 sau de următoarea formă:



Codul de eroare returnat de excepții

Indexul selectorului şi TI sunt luați din selectorul segmentului asociat cu excepția. În locul câmpului RPL se află codul de eroare, de 2 biți, cu următoarea semnificație:

- dacă I=1, selectorul face referire la un index din tabela IDT, iar bitul TI este ignorat;
- dacă I=0, bitul TI indică dacă selectorul este din GDT (TI=0) sau din LDT (TI=1);
- dacă EX=1, eroarea a fost determinată de un eveniment din afara programului în execuție.

Pe lângă cele trei tipuri de porți, mecanismul de protecție mai suportă şi un al patrulea tip de poartă, cea de apel (call gate). Porțile de apel sunt invocate printr-o instrucțiune de apel de rutină standard. Ele sunt utilizate pentru a realiza comunicația internivel, permițând astfel unei aplicații ce rulează pe nivelul de privilegiu 3 să poată apela serviciile sistemului de operare, care rulează pe nivelul 0, de protecție.

Când se invocă o rutină printr-o poartă de apel, nivelul de privilegiu al rutinei ce se execută se modifică la nivelul segmentului de cod la care se referă poarta. Pentru a asigura integritatea datelor (parametrii şi adresa de revenire) salvate în stiva nivelului mai puțin privilegiat (care ar putea fi modificată de aplicație), o parte a stivei este copiată, prin poartă, într-un segment de stivă mai privilegiat (al rutinei care se execută).

Descriptorul unei porți de apel conține, pe lângă adresa rutinei (selector şi offset), un câmp (denumit contor) care indică numărul de cuvinte duble copiate din stiva mai puțin privilegiată în cea mai privilegiată. Fiind patru niveluri de privilegiu, fiecare aplicație trebuie să aibă tot atâtea segmente de stivă. Pointerul pentru stiva activă, dintre cele patru posibile, se află memorat în SS: (E)SP. Celelalte sunt memorate într-un obiect sistem, denumit segment de stare task (TSS – Task State Segment). Când un apel sau întrerupere, printr-o poartă, determină o modificare în nivelul de privilegiu, noile valori pentru SS: (E)SP sunt încărcate din TSS, din una din cele trei perechi disponibile : SS0: (E)SP0, SS1: (E)SP1, SS2: (E)SP2. Selectorul pentru TSS-ul curent este memorat într-un registru de task (TR).

Când apare o comutare de task, toate registrele task-ului executat (generale şi segment) sunt salvate în TSS-ul activ, apoi registru task (TR) este încărcat cu selectorul noului TSS (al task-ului care urmează să se execute) şi fiecare registru (general/segment) este încărcat cu valorile corespunzătoare din noul TSS. Noul TSS conține şi un câmp care face legătura cu task-ul anterior, astfel ca la terminarea sa să se poată comuta la task-ul anterior.

Întrucât modul virtual 8086 (V86) este parte a modului protejat, întreruperile sunt manipulate prin IDT în modul protejat standard.

Excepții specifice modului protejat

Întreruperea de depanare (1) poate apărea la îndeplinirea uneia dintre condițiile: întrerupere „single step“, breakpoint (suspendare) datorat unui registru de depanare (DR) sau la o comutare de task. La apariția unei întreruperi de depanare, bitul RF (Resume Flag, din registrul indicatori) este setat (1), mascând întreruperile de depanare suplimentare. După execuția completă a unei instrucțiuni, procesorul şterge bitul RF (0). Setarea bitului TF conduce la o întrerupere „single step“, vectorul IDT(1), care pune TF pe 0. Un program de depanare nu poate executa „single step“ o instrucțiune INT n, dar poate plasa un breakpoint fie la destinația porții referite de INT n, fie imediat după instrucțiunea INT. Un apel de poartă nu şterge TF, astfel că programul de depanare verifică toate instrucțiunile JMP/CALL de tip FAR, pentru a vedea dacă determină o modificare de privilegiu. Dacă este aşa, programatorii nu vor avea permisiunea la „single-step“ într-un cod mai privilegiat decât aplicația lor.

Începând de la procesorul 386, există şi 8 registre de depanare (DR0-DR7), care pot memora 4 adrese de breakpoint (suspendarea execuției programului curent la anumite adrese). Fiecare identifică o adresă liniară. Dacă procesorul adresează acele adrese, se generează întreruperea de depanare (1). Pentru a fixa puncte de întrerupere, se utilizează instrucțiunea MOV DRx,reg (reg este un registru general de 32 de biți). Primele 4 registre (DR0-DR3) sunt registre de adresă, în care se încarcă adresa liniară din descriptori. DR4 şi DR5 sunt rezervate Intel. DR6 este un registru de stare, care indică condițiile ce determină întreruperea; sunt câte 4 biți pentru fiecare dintre cele 4 registre de adresă, specificând care dintre ele a generat întreruperea:

- BT, întrerupere „single-step“, dacă TF=1;
- BT, comutare de task; bitul T al noului task este pus pe 1;
- BD este pus pe 1 dacă se utilizează hardware ICE (In-Circuit Emulator), adică registrele de depanare sunt rezervate pentru acest emulator (ICE). Dacă BD=1, orice încercare de a plasa o valoare în registrele de depanare va genera întreruperea 1.

Procedura de tratare a întreruperii de depanare trebuie să şteargă conținutul registrului DR6. Procesorul setează biții, dar biții pot fi şterşi numai prin program. DR7 este registrul de control depanare. Numai încărcarea unei adrese în DR0-DR3 nu va activa o întrerupere. Trebuie setați biții de activare din DR7 şi pusă condiția şi lungimea breakpoint-ului:

- câmpul lungime specifică, pe doi biți, tipul adresei: octet (orice adresă), cuvânt (acoperă cele două adrese ale unui cuvânt), dublu cuvânt (acoperă patru adrese);
- un alt câmp specifică, pe doi biți, tipul accesului la memorie, al breakpoint-ului pentru cod (execuție program), scriere sau scriere/citire memorie.

Mai există biții L_n, G_n, care permit ca punctele de suspendare să fie locale sau globale. Astfel, dacă L_n=1, o comutare de task şterge biții L_n. Sistemul marchează bitul T din TSS-ul task-ului utilizând breakpoint-uri locale, astfel că o întrerupere pe nivel 1 apare când task-ul este reactivat; atunci biții L_n pot fi şterşi.

Întrucât există doar 4 puncte de breakpoint (registre) active la un moment dat, este de preferat utilizarea registrelor de depanare pentru date şi întreruperea de breakpoint (INT 3) pentru instrucțiuni.

Excepția „Dublă Eroare“ (DF – Double Fault (8)) apare când la o singură instrucțiune se detectează două violări de protecție separate (de ex. o eroare de protecție pe nivelul trei este urmată de o eroare de segment „not present“ (NP) datorită absenței segmentului) sau, cu alte cuvinte, la apariția a două excepții succesive, ce nu pot fi serializate. Dacă apar şi alte violări de

protecție pe durata prelucrării excepției 8, atunci procesorul se blochează (shutdown), stare din care iese numai prin comandă pe linia NMI, caz în care rămâne în mod protejat, respectiv pe linia RESET, caz în care trece în modul real. La încărcarea registrului SS întreruperile hard şi de depanare sunt mascate pe durata instrucțiunii următoare încărcării lui SS. Astfel, se poate încărca registrul (E)SP fără riscul apariției unei întreruperi cu un pointer incorect la stivă. Poate să apară doar întreruperea 14 (PG), care va invoca, eventual, o stivă incorectă, conducând la o dublă eroare (8). Pentru a evita acest lucru se pot încărca ambele registre (SS şi ESP) utilizând o singură instrucțiune, LSS.

Excepția datorată unui „segment de stare task (TSS)“ invalid (10), care poate să apară dacă pe durata unei comutări de task noul TSS referit de poarta task este invalid: de exemplu, stiva definită este un segment în care nu se poate scrie sau selectoarele de segment sunt în afara limitelor.

Excepția „Not Present“ (NP), pe nivelul 11, apare la încercarea de a încărca un segment care nu e prezent, sau la utilizarea unui descriptor de segment care este marcat „not-present“. Această eroare permite repornirea programului, adică dacă rutina excepției face segmentul prezent şi returnează controlul, programul întrerupt îşi va relua execuția.

Excepția „eroare de stivă“ (12) apare la depăşirea superioară sau inferioară a stivei de către instrucțiunile ce lucrează cu stiva (POP, PUSH, ENTER, LEAVE) sau în urma încercării de a încărca registrul SS cu un descriptor care este marcat „not-present“ în timpul unei tranziții inter-task sau inter-nivel. O instrucțiune ce determină această eroare este restartabilă.

Eroarea de protecție generală (GP-General Protection), pe nivelul 13, apare în urma unei violări de protecție (transferul controlului la un segment care nu este executabil, scrierea într-un segment de cod), care nu este acoperită de celelalte întreruperi (violări limită, scriere în segmente de tip „read-only“), referire la un segment de date cu nivel de privilegiu mai mare decât cel curent, sau încărcări ale registrelor segment (SS, DS, ES, FS, GS) cu descriptori de sistem, cu segmente executabile sau care nu pot fi citite; în cazul acestei excepții se returnează, pe lângă adresa instrucțiunii respective, şi un cod de eroare.

La 386/486 mai pot să apară întreruperile 14 şi 17.

Întreruperea 14 (PG-Page Fault), „eroare de pagină“, apare când adresarea paginată este activată şi procesorul detectează, la translația adresei liniare în adresă fizică, o eroare: fie tabelele de pagină nu sunt prezente (bitul P=0), fie procedura curentă nu are drepturi de privilegiu la pagina respectivă.

Întreruperea 17 (AC-Alignment Check), „eroare de aliniere“, apare în cazul unei referințe la un operand din memorie care nu este aliniat, în cazul procesorului 486. Această eroare este activată de bitul 18 (AC) din registrul indicatori. Un astfel de acces se realizează dacă se face acces la un cuvânt de 16 biți memorat la o adresă impară sau la un dublu cuvânt, de 32 biți, care se găseşte la o adresă ce nu este multiplu de 4, sau la o dată de 64 biți (de ex. un real de tip „double“) a cărei adresă nu este divizibilă cu 8. Aceste transferuri sunt denumite transferuri nealiniate. Alinierea operandului şi dimensiunea sa determină numărul de cicluri de magistrală necesare pentru transfer. Dacă datele sunt aliniate, ele pot fi transferate pe durata minimă, adică un ciclu de magistrală, altfel fiind necesar un ciclu suplimentar pentru a finaliza transferul. Aceste erori de nealiniere sunt generate de programe ce rulează pe nivelul 3 de privilegiu. Bitul AC este ignorat la nivelurile superioare de privilegiu (0, 1, 2). Referințele la TSS se găsesc implicit pe nivelul 0, chiar dacă instrucțiunea care face referința este executată pe nivelul 3.

Toate aceste excepții sunt clasificate în următoarele tipuri: eroare, capcană şi abandon (abort), în funcție de modul în care sunt semnalate şi dacă se poate restarta instrucțiunea care a determinat excepția.

În prima categorie de excepții de eroare intră excepțiile care sunt semnalate „înainte“ de instrucțiunea ce determină excepția. Ele sunt detectate fie înainte să înceapă execuția instrucțiunii, fie pe durata execuției; eroarea este semnalată printr-o stare ce permite ca instrucțiunea să fie restartată, prin salvarea adresei instrucțiunii ce a determinat excepția. De exemplu, o astfel de excepție se generează când operandul unei instrucțiuni se află într-un segment marcat ca non-prezent. Un număr special de întrerupere este asociat cu fiecare condiție de eroare. Pointerul salvat în stivă după o astfel de excepție face referire la instrucțiunea ce a cauzat-o, astfel că sistemul de operare poate corecta eroarea şi relua execuția instrucțiunii. Astfel de excepții sunt: 0, 5, 6, 7, 10, 11, 12, 16.

Cea de-a doua categorie, excepții capcană, este semnalată la execuția instrucțiunii imediat următoare celei care a determinat excepția; în acest caz, adresa salvată face referire la instrucțiunea următoare celei ce a generat excepția. De exemplu instrucțiunea INTO. La execuția ei, procesorul verifică starea indicatorului OF; dacă OF=1 se generează întrerupere (INT 4). Toate întreruperile soft (INT n) sunt de acest tip. Pentru a executa o astfel de instrucțiune trebuie să existe un privilegiu de acces la descriptorul IDT pentru numărul întreruperii. De exemplu, dacă o aplicație pe nivelul 3 de privilegiu execută instrucțiunea INT 47, descriptorul la IDT(47) trebuie să aibă DPL=3, altfel apare o eroare de protecție. Acest mecanism împiedică aplicațiile de a executa instrucțiunile INT n, asociate cu întreruperi hard, deoarece porțile pentru acești vectori fac referire la codul sistemului de operare, care rulează la nivelul cel mai privilegiat (0).

Ultima categorie de excepții, abandon (abort), cuprinde acele excepții care nu permit localizarea precisă a instrucțiunii şi a contextului ce le-au generat, şi deci nu permite restartarea programului (8, 9, 13). Tot în această categorie intră şi erorile severe, hardware, sau detectarea unor valori ilegale în tabelele de sistem.

Întreruperile din intervalul 32-255 sunt disponibile pentru utilizare de către sistemul de operare. În acest spațiu se pot instala porți de întrerupere/capcană/task. Procedurile de întrerupere pot fi invocate prin soft (INT n) sau prin semnale hardware (pe linia INTR).