Instrukce procesoru pentium.

Soubor hlavních instrukcí.

Pavlů Zdeněk

01.06.2016

Tento dokument jsem vyrobil zhruba před 12 roky. Neobsahuje veškeré instrukce. Při výrobě jsem čerpal z materiálů „Intel Corporation“.

Aritmetické instrukce

ADD (ADDition) : ADD dst, src
součet
Součet operandů je uložen do levého (cílového) operandu.
příznaky: mění podle výsledku AF,CF,OF,PF,SF,ZF

ADC (ADdition with Carry) : ADC dst, src
součet s přenosem
Součet operandů s přičtením příznaku přenosu (CF) je uložen do levého (cílového) operandu.
příznaky: mění podle výsledku AF,CF,OF,PF,SF,ZF

INC (INCrement) : INC src
inkrementace
Přičtení jedničky k operandu.
příznaky: mění podle výsledku AF,OF,PF,SF,ZF

SUB (SUBtraction) : SUB dst, src
odčítání
Pravý operand (src) je odečten od levého (dst) a výsledek rozdílu je uložen do levého (cílového) operandu.
příznaky: mění podle výsledku AF,CF,OF,PF,SF,ZF

SBB (SuBtraction with Borrow) : SBB dst, src
odčítání s výpůjčkou
Pravý operand (src) je odečten od levého (dst) a výsledek rozdílu je uložen do levého (cílového) operandu.Je-li nastaven příznak CF je navíc odečtena jednička (výpůjčka).
příznaky: mění podle výsledku AF,CF,OF,PF,SF,ZF

DEC (DECrement) : DEC src
dekrementace
Odečtení jedničky od operandu.
příznaky: mění podle výsledku AF,OF,PF,SF,ZF

CMP (CoMPare) : CMP dst, src
porovnání
Pravý operand (src) je odečten od levého (dst) bez ukládání výsledku rozdílu.
příznaky: mění podle výsledku AF,CF,OF,PF,SF,ZF

NEG (NEGate) : NEG src
neguj
Změna znaménka operandu podle dvojkového doplňku.
příznaky: mění podle výsledku AF,CF,OF,PF,SF,ZF

MUL (MULtiplication unsigned) : MUL dst, src
násobení bez znaménka
Násobení celých čísel bez znaménka.
příznaky: mění podle výsledku AF,CF,OF,PF,SF,ZF

IMUL (Integer MULtiplication signed) : IMUL dst, src
celočíselné násobení se znaménkem
Násobení celých čísel se znaménkem v dvojkovém doplňkovém kódu.

příznaky: mění podle výsledku AF,CF,OF,PF,SF,ZF

DIV (DIVision unsigned) : DIV dst, src

dělení bez znaménka

Dělení celých čísel bez znaménka.

příznaky: mění podle výsledku AF,CF,OF,PF,SF,ZF

IDIV (Integer DIVision signed) : IDIV dst, src

celočíselné dělení se znaménkem

Dělení celých čísel se znaménkem v dvojkovém doplňkovém kódu.

příznaky: mění podle výsledku AF,CF,OF,PF,SF,ZF

CBW (Convert Byte to Word) : CBW

převeď byte do slova

Znaménkové rozšíření nižší slabiky střadače do slabiky vyšší.

formát: 1001 1000 (98)

funkce: Je-li (AL)<0x80 pak 0x00 =>AH jinak 0xff =>AH

příznaky: nemění

CWD (Convert Word to Doubleword) : CWD

převeď slovo na dvojslovo

Znaménkové rozšíření střadače (AX) do registru DX.

formát: 1001 1001 (99)

funkce: Je-li (AX)<0x8000 pak 0x0000 =>DX jinak 0xffff =>DX

příznaky: nemění

CWDE (Convert Word to Doubleword) : CWDE

převeď slovo na dvojslovo

Znaménkové rozšíření střadače (AX) do 31 až 16-bitu registru EAX.

formát: 0110 0110 1001 1000 (66 98)

funkce: Je-li (AX)<0x8000 pak 0x0000 =>EAX jinak 0xffff =>EAX

příznaky: nemění

CDQ (Convert Doubleword to Quadword) : CDQ

převeď dvojslovo na čtyřslovo

Znaménkové rozšíření střadače (EAX) do páru registrů EDX:EAX.

formát: 0110 0110 1001 1001 (66 99)

funkce: Je-li (EAX)<0x80000000 pak 0x00000000 =>EDX jinak 0xffffffff =>EDX

příznaky: nemění

Bitové instrukce

SET (SET byte) : SET dst

nastav byte

Skupina SET byla poprvé zavedena u procesorů 386. Instrukce pracují s jednotlivými bity nebo řetězci bitů. Nastavují cílový operand (dst) na hodnotu 1 při splnění podmínky, jinak operand vynulují.

- SETA - nastav, je-li nad (CF=0 nebo ZF=0)
- SETAE - nastav, je-li nad nebo rovno (CF=0)
- SETB - nastav, je-li pod (CF=1)
- SETBE - nastav, je-li pod nebo rovno (CF=1 a ZF=1)
- SETE - nastav, je-li rovno (ZF=1)
- SETG - nastav, je-li větší (ZF=0 a SF=OF)
- SETGE - nastav, je-li větší nebo rovno (SF=OF)
- SETL - nastav, je-li menší (SF<>OF)
- SETLE - nastav, je-li menší nebo rovno (SF<>OF nebo ZF=1)
- SETNA - nastav, není-li nad (CF=1 a ZF=1)
- SETNAE - nastav, není-li nad nebo rovno(CF=1)
- SETNB - nastav, není-li menší (CF=0)
- SETNBE - nastav, není-li menší nebo rovno (CF=0 nebo ZF=0)
- SETNE - nastav, není-li rovno (ZF=0)
- SETNG - nastav, není-li větší (ZF=1 nebo SF<>OF)
- SETNGE - nastav, není-li větší nebo rovno (SF<>OF)
- SETNL - nastav, není-li menší (SF=OF)
- SETNLE - nastav, není-li menší nebo rovno (SF=OF a ZF=0)
- SETNO - nastav, není-li přeplnění (OF=0)
- SETNP - nastav, není-li parita (PF=0)
- SETNS - nastav, není-li znaménko (SF=0)
- SETNZ - nastav, není-li nula (ZF=0)
- SETO - nastav, je-li přeplnění (OF=1)
- SETP - nastav, je-li parita (PF=1)
- SETS - nastav, je-li záporné (SF=1)
- SETZ - nastav, je-li nula (ZF=1)

příznaky: nemění

BSF (Bit Scan Forward) : BSF dst, src

hledání bitu kupředu

Hledání prvního nenulového bitu (od D0) ve zdrojovém operandu.Při nalezení je ZF nastaven na jedničku a číslo bitu navráceno v cílovém operandu,jinak je ZF vynulován. Cílovým operandem musí být registr.

formát: 0000 1111 1011 1100 mod reg r/m [displ [disph]] příznaky: nastavuje ZF. Hodnoty OF,SF,AF,PF,CF nejsou po provedení definovány.

BSR (Bit Scan Reverse) : BSR dst, src

hledání bitu pozpátku

Hledání prvního nenulového bitu od 15-tého bitu (od D31 32-bit) ve zdrojovém operandu.Při nalezení je ZF nastaven na jedničku a číslo bitu navráceno v cílovém operandu,jinak je ZF vynulován. Cílovým operandem musí být registr.

formát: 0000 1111 1011 1101 mod reg r/m [displ [disph]] příznaky: nastavuje ZF. Hodnoty OF,SF,AF,PF,CF nejsou po provedení definovány.

Instrukce pro práci s řetězci dat

REP (REPeat string operation) : REP

opakuj řetězcovou operaci

Následující řetězcová operace (MOVS, STOS) bude prováděna opakovaně. Počet opakovaní udává hodnota registru CX (ECX), která je po každém provedení řetězcové instrukce snížena o jedničku. Opakování je prováděno dokud je CX různé od nuly.

formát: 1111 0010 (f2)

funkce: L : (CX)-1=>CX; Je-li(CX)>0, pak proveď řetězcovou operaci

příznaky: podle řetězcové operace

REPE (REPeat string operation to Equal) : REPE

opakuj řetězcovou operaci do shody

Následující řetězcová operace (CMPS, SCAS) bude prováděna opakovaně. Počet opakovaní udává hodnota registru CX (ECX), která je po každém provedení řetězcové instrukce snížena o jedničku. Opakování je prováděno dokud je CX různé od nuly. Opakování bude předčasně ukončeno v případě, že řetězcová operace vynuluje příznak ZF.

formát: 1111 0010 (f2)

funkce: L : (CX)-1=>CX; Je-li(CX)>0 a zároveň (ZF)=1, pak proveď řetězcovou operaci

příznaky: podle řetězcové operace

REPNE (REPeat string operation to Not Equal) : REPNE

opakuj řetězcovou operaci dokud je shoda

Následující řetězcová operace (CMPS, SCAS) bude prováděna opakovaně. Počet opakovaní udává hodnota registru CX (ECX), která je po každém provedení řetězcové instrukce snížena o jedničku. Opakování je prováděno dokud je CX různé od nuly. Opakování bude předčasně ukončeno v případě, že řetězcová operace nastaví příznak ZF.

formát: 1111 0011 (f3)

funkce: L : (CX)-1=>CX; Je-li(CX)>0 a zároveň (ZF)=0, pak proveď řetězcovou operaci

příznaky: podle řetězcové operace

MOVS (MOVe String) : MOVS src, dst

přesuň řetězec

Přesun slabiky (MOVSB), slova (MOVSW) nebo dvojslova (MOVSD) ze zdrojového řetězce (DS:SI) do cílového (ES:DI). Podle příznaku směru (DF) budou po každém přesunu změněny registry SI (ESI 32-bit) a DI (EDI 32-bit) o velikost delta nahoru

(DF=0) nebo dolů (DF=1). Delta je buď 1 byte (MOVSB), 2 byte (MOVSW) nebo 4 byte (MOVSD).
příznaky: nemění

CMPS (CoMPare String) : CMPS src, dst
porovnej řetězec
Odečítá slabiky (CMPSB), slova (CMPSW) nebo dvojslova (CMPSD) zdrojového řetězce (DS:SI) s cílovím (ES:DI). Podle příznaku směru (DF) budou po každém porovnání změněny registry SI (ESI 32-bit) a DI (EDI 32-bit) o velikost delta nahoru (DF=0) nebo dolů (DF=1). Delta je buď 1 byte (CMPSB), 2 byte (CMPSW) nebo 4 byte (CMPSD).
příznaky: mění AF,CF,OF,PF,SF,ZF

SCAS (SCAn String) : SCAS src
prohledej řetězec
Odečítá slabiku (SCASB), slovo (SCASW) nebo dvojslovo (SCASD) zdrojového řetězce (DS:SI) se střadačem (AX). Podle příznaku směru (DF) bude po každém porovnání změněn registr SI (ESI 32-bit) o velikost delta nahoru (DF=0) nebo dolů (DF=1). Delta je buď 1 byte (SCASB), 2 byte (SCASW) nebo 4 byte (SCASD).
příznaky: mění AF,CF,OF,PF,SF,ZF

LODS (LOaD String) : LODS src
načti řetězec
Přesouvá slabiku (LODSB), slovo (LODSW) nebo dvojslovo (LODSD) zdrojového řetězce (DS:SI) do střadače (AX). Podle příznaku směru (DF) bude po každém přesunu změněn registr SI (ESI 32-bit) o velikost delta nahoru (DF=0) nebo dolů (DF=1). Delta je buď 1 byte (LODSB), 2 byte (LODSW) nebo 4 byte (LODSD). Opakování (REP) nemá příliš využití.
příznaky: nemění

STOS (STOre String) : STOS dst
ulož řetězec
Přesouvá slabiku (STOSB), slovo (STOSW) nebo dvojslovo (STOSD) ze střadače (AL/AX/EAX) do cílového řetězce (ES:DI). Podle příznaku směru (DF) bude po každém přesunu změněn registr DI (ESI 32-bit) o velikost delta nahoru (DF=0) nebo dolů (DF=1). Delta je buď 1 byte (STOSB), 2 byte (STOSW) nebo 4 byte (STOSD). Naplnění bloku konstantou (nejčastěji nulou).
příznaky: nemění

Instrukce přesunu dat

MOV (MOVe) : MOV dst, src
přesuň
Přesun dat ze střadače do paměti.
formát: 1010 001W addr
funkce: podle W:

- W = 0 : (AL)=>addr
- W = 1 : (AX)=>addr přesun z AX (16) nebo (EAX)=>addr EAX (32) podle módu procesoru (16/32 bitů)

příznaky: nemění

MOV (MOVe) : MOV dst, src

přesuň

Přesun dat z paměti do střadače.

formát: 1010 000W addr

funkce: podle W:

- W = 0 : (addr)=>AL, přesun do AL
- W = 1 : (addr)=>AX, přesun do AX (16) nebo (addr)=>EAX, přesun do EAX (32) podle módu procesoru (16/32 bitů)

příznaky: nemění

MOV (MOVe) : MOV dst, src

přesuň

Přesun dat z registru do paměti nebo registru, nebo z paměti do registru (výběr podle D).

formát: 1000 10DW mod reg r/m [displ[disph]]

funkce: podle D:

- D = 0 : (REG)=>EA (mod, r/m)
- D = 1 : (EA)(mod, r/m)=>REG

příznaky: nemění

MOV (MOVe) : MOV dst, src

přesuň

Přesun přímého operandu do registru.

formát: 1011 W reg datal [datah]

funkce: data=>REG (podle W se rozhodne mezi 16/32 bitovou operací)

příznaky: nemění

MOV (MOVe) : MOV dst, src

přesuň

Přesun přímého operandu do paměti nebo registru.

formát: 1100 011W mod 000 r/m [displ[disph]] datal [datah]

funkce: data=>EA (mod r/m) (podle W se rozhodne mezi 16/32 bitovou operací)

příznaky: nemění

PUSH (PUSH word to stack) : PUSH src

ulož slovo do zásobníku

Uložení univerzálního registru [E](AX, BX, CX, DX, SI, DI, BP, SP) .

formát: 0101 0 reg

funkce: mód:

- 16 bitů : (SP)-2=>(SP); (REG)=>(SP)
- 32 bitů : (ESP)-4=>(ESP); (REG)=>(ESP)

příznaky: nemění

POP (POP word from stack) : POP dst

vyber slovo ze zásobníku

Uložení do univerzálního registru [E](AX, BX, CX, DX).

formát: 0101 1 reg

funkce: mód:

- 16 bitů : (SP)=>(REG); (SP)+2=>(SP)
- 32 bitů : (ESP)=>(REG); (ESP)+4=>(ESP);

příznaky: nemění

PUSHA (PUSH to stack All registers) : PUSHA

ulož do zásobníku všechny registry

Uložení celé sady mikroprocesoru 80286+.

formát: 0110 0000 (60)

funkce: jako "PUSH" na AX, BX, CX, DX, SI, DI, BP, SP

příznaky: nemění

POPA (POP from stack All registers) : POPA

vyber ze zásobníku všechny registry

Vyzvednutí celé sady uložené PUSHA.

formát: 0110 0001 (61)

funkce: jako "POP" na AX, BX, CX, DX, SI, DI, BP, SP

příznaky: nemění

LAHF (Load status Flags into AH register) :

naplň registr AH z registru příznaků

Naplnění registru AH hodnotami příznaků z registru příznaků ve tvaru registru příznaků mikroprocesoru Intel 8080.

formát: 1001 1111 (9f)

funkce: (SF):(ZF):0:(AF):0:(PF):(0):(CF) => AH

příznaky: nemění

SAHF (Store AH to Flags) :

ulož registr AH do registru příznaků

Hodnoty určitých příznaků jsou přepsány hodnotami bitů v AH.

formát: 1001 1110 (9e)

funkce: (AH0)=>CF; (AH2)=>PF; (AH4)=>AF; (AH6)=>ZF; (AH7)=>SF;

příznaky: mění (AF, CF, PF, SF, ZF)

Instrukce rotací a posuvů

SAL (Shift Arithmetic Left) : SAL src, count

aritmetický posun doleva

Posun operandu n-krát doleva. Každým posunem je nejvyšší bit operandu přesunut do příznaku CF. Do nejnižšího bitu jsou doplňovány nuly. Je-li počet posunů roven jedné bude příznak OF nastaven, pokud se nový nejvyšší bit neshoduje s příznakem CF (předchozí nejvyšší bit), jinak bude OF vynulován. Při jiném počtu posunů není hodnota příznaku OF definována.

příznaky: mění CF,OF,PF,SF,ZF a AF není po provedení definován

SHL (SHift logical Left) : SHL src, count

logický posun doleva

Posun operandu n-krát doleva. Každým posunem je nejvyšší bit operandu přesunut do příznaku CF. Do nejnižšího bitu jsou doplňovány nuly. Je-li počet posunů roven jedné bude příznak OF nastaven, pokud se nový nejvyšší bit neshoduje s příznakem CF (předchozí nejvyšší bit), jinak bude OF vynulován. Při jiném počtu posunů není hodnota příznaku OF definována.

příznaky: mění CF,OF,PF,SF,ZF a AF není po provedení definován

SAR (Shift Arithmetic Right) : SAR src, count

aritmetický posun doprava

Posun operandu n-krát doprava. Každým posunem je nejnižší bit operandu přesunut do příznaku CF. V nejvyšším bitu se při každém posunu zachovává jeho hodnota. Příznak OF je vynulován.

příznaky: mění CF,PF,SF,ZF a AF není po provedení definován, OF je vynulován

SHR (SHift logical Right) : SHR src, count

logický posun doprava

Posun operandu n-krát doprava. Každým posunem je nejnižší bit operandu přesunut do příznaku CF. Do nejvyššího bitu bude při každém posunu doplněna nula. Je-li počet posunů roven jedné bude příznak OF nastaven, pokud se nový nejvyšší bit neshoduje s příznakem CF (předchozí nejvyšší bit), jinak bude OF vynulován. Při jiném počtu posunů není hodnota příznaku OF definována.

příznaky: mění CF,OF,PF,SF,ZF a AF není po provedení definován

RCL (Rotate Left through Carry) : RCL src, count

rotuj doleva přes příznak CF

Rotace operandu n-krát doleva přes příznak CF. Při každé rotaci je nejnižší bit doplněn hodnotout příznaku CF a hodnota nejvyššího bitu zapsána do CF. Příznak OF je nastaven, je-li počet rotací roven jedné a dva nejvyšší bity operandu (před rotací) jsou různé, při shodě je OF vynulován. Při jiném počtu opakování není hodnota OF definována.

příznaky: mění CF,OF

ROL (ROtate Left) : ROL src, count

rotuj doleva

Rotace operandu n-krát doleva. Při každé rotaci je nejnižší bit doplněn hodnotout nejvyššího bitu a hodnota nejvyššího bitu zapsána do CF. Příznak OF je nastaven, je-li

počet rotací roven jedné a dva nejvyšší bity operandu (před rotací) jsou různé, při shodě je OF vynulován. Při jiném počtu opakování není hodnota OF definována.
příznaky: mění CF,OF

RCR (Rotate Right through Carry) : RCR src, count

rotuj doprava přes příznak CF

Rotace operandu n-krát doprava přes příznak CF. Při každé rotaci je nejvyšší bit doplněn hodnotout příznaku CF a hodnota nejnižšího bitu zapsána do CF. Příznak OF je nastaven, je-li počet rotací roven jedné a dva nejvyšší bity operandu (před rotací) jsou různé, při shodě je OF vynulován. Při jiném počtu opakování není hodnota OF definována.
příznaky: mění CF,OF

ROR (ROtate Right) : ROR src, count

rotuj doprava

Rotace operandu n-krát doprava. Při každé rotaci je nejvyšší bit doplněn hodnotout nejnižšího bitu a hodnota nejnižšího bitu zapsána do CF. Příznak OF je nastaven, je-li počet rotací roven jedné a dva nejvyšší bity operandu (před rotací) jsou různé, při shodě je OF vynulován. Při jiném počtu opakování není hodnota OF definována.
příznaky: mění CF,OF

Instrukce skoků, volání a návratů

JMP (JuMP) : JMP dst

skok

Instrukce nepodmíněného skoku předá řízení na adresu operandu.
příznaky: nemění

Jcc (Jump if condition is met) : Jcc dst

podmíněný skok

Instrukce podmíněného skoku předá řízení na adresu operandu jen pokud je splněna podmínka. Ta je určena některými bity registru příznaků nastavených předchozí instrukcí měnící příznaky.

- JA - skok, je-li nad (CF=0 nebo ZF=0)
- JAE - skok, je-li nad nebo rovno (CF=0)
- JB - skok, je-li pod (CF=1)
- JBE - skok, je-li pod nebo rovno (CF=1 a ZF=1)
- JC - skok, je-li přenos (CF=1)
- JE - skok, je-li rovno (ZF=1)
- JG - skok, je-li větší (ZF=0 a SF=OF)
- JGE - skok, je-li větší nebo rovno (SF=OF)
- JL - skok, je-li menší (SF<>OF)
- JLE - skok, je-li menší nebo rovno (SF<>OF nebo ZF=1)
- JNA - skok, není-li nad (CF=1 a ZF=1)
- JNAE - skok, není-li nad nebo rovno(CF=1)
- JNB - skok, není-li menší (CF=0)
- JNBE - skok, není-li menší nebo rovno (CF=0 nebo ZF=0)
- JNC - skok, není-li přenos (CF=0)

- JNE - skok, není-li rovno (ZF=0)
- JNG - skok, není-li větší (ZF=1 nebo SF<>OF)
- JNGE - skok, není-li větší nebo rovno (SF<>OF)
- JNL - skok, není-li menší (SF=OF)
- JNLE - skok, není-li menší nebo rovno (SF=OF a ZF=0)
- JNO - skok, není-li přeplnění (OF=0)
- JNP - skok, není-li parita (PF=0)
- JNS - skok, není-li znaménko (SF=0)
- JNZ - skok, není-li nula (ZF=0)
- JO - skok, je-li přeplnění (OF=1)
- JP - skok, je-li parita (PF=1)
- JS - skok, je-li záporné (SF=1)
- JZ - skok, je-li nula (ZF=1)

příznaky: nemění

JCXZ (Jump if CX is Zero) : JCXZ dst

skok, je-li registr CX nulový

Instrukce podmíněného skoku předá řízení na adresu operandu pokud je obsah registru CX (čítače) nulový.

příznaky: nemění

JECXZ (Jump if ECX is Zero) : JECXZ dst

skok, je-li registr ECX nulový

Instrukce podmíněného skoku předá řízení na adresu operandu pokud je obsah registru ECX (čítače) nulový.

příznaky: nemění

LOOP (LOOP) : LOOP dst

cyklus

Instrukce sníží obsah čítače CX (ECX v 32-bitovém režimu) o jedničku a předá řízení na místo operandu je-li obsah čítače nenulový.

příznaky: nemění

CALL (CALL a procedure) : CALL dst

volání podprogramu

Předání řízení podprogramu s uložením návratové hodnoty do zásobníku.

příznaky: nemění

RET (RETurn from procedure) : RET [n]

návrat z podprogramu

Navrácení řízení na návratovou adresu uloženou v zásobníku předchozíinstrukcí CALL. Volitelné uvedení celočíselné konstanty [n] vymažen slov ze zásobníku (parametry podprogramu).

příznaky: nemění

Logické instrukce

AND (AND) : AND dst, src
    logický součin
    Výsledek logického součinu dvou operandů je uložen do levého (dst) operandu.
    příznaky: nuluje CF,OF; mění PF,SF,ZF; po provedení není definován AF
OR (OR) : OR dst, src
    logický součet
    Výsledek logického součtu dvou operandů je uložen do levého (dst) operandu.
    příznaky: nuluje CF,OF; mění PF,SF,ZF; po provedení není definován AF
XOR (XOR) : XOR dst, src
    výhradní logický součet
    Výsledek výhradního logického součtu dvou operandů je uložen do levého (dst) operandu.
    příznaky: nuluje CF,OF; mění PF,SF,ZF; po provedení není definován AF
NOT (NOT) : NOT src
    negace
    Výsledek negace (záměna jednotlivých bitů) je uložen zpět do operandu.
    příznaky: nemění
TEST (TEST) : TEST dst, src
    logické porovnání
    Logického součin dvou operandů bez uložení výsledku.
    příznaky: nuluje CF,OF; mění PF,SF,ZF; po provedení není definován AF

Řídící instrukce

STC (SeT Carry flag) :
    nastav příznak přenosu
    nastavuje příznak CF
    formát: 1111 1001 (f9)
    funkce: CF = 1
    příznaky: mění (CF)
CLC (CLear Carry flag) :
    nuluj příznak přenosu
    nuluje příznak CF
    formát: 1111 1000 (f8)
    funkce: CF = 0
    příznaky: mění (CF)
CMC (CoMplement Carry flag) :
    invertuj příznak přenosu
    invertuje příznak CF
    formát: 1111 0101 (f5)
    funkce: CF = ~CF
    příznaky: mění (CF)
STD (SeT Direction flag) :
    nastav příznak směru

nastavuje příznak směru zpracování řetězců
formát: 1111 1101 (fd)
funkce: DF = 1
příznaky: mění (DF)

CLD (Clear Direction flag) :
nuluj příznak směru
nuluje příznak směru zpracování řetězců
formát: 1111 1100 (fc)
funkce: DF = 0
příznaky: mění (DF)

NOP (No OPeration) :
žádná operace
Zpoždění tří period hodin (8086 - 80386) nebo jedné periody (80486+). Ve skutečnosti jde o instrukci XCHG AX, AX.
formát: 1001 0000 (90)
funkce: žádná
příznaky: nemění

Aritmetické funkce a konstanty numerického koprocesoru

konstanty

Hodnoty konstant jsou ukládány na vrchol zásobníku.

- FLDZ - nula
- FLD1 - jedna
- FLDPI - Ludolfovo číslo (3.14159265357...)
- FLDLN2 - ln(2)
- FLDL2E - Eulerovo číslo (2.718281828459...)
- FLDL2T - logaritmus (10) o základu 2
- FLDLG2 - log(2)

funkce

Funkce pracuje s hodnotou na vrcholu zásobníku ST(0). Na vrchol zásobníku poté uloží i výsledek. Složitější funkce pracují i s druhou hodnotou zásobníku ST(1).

- FABS - absolutní hodnota
- FSQRT - druhá odmocnina (reálného čísla)
- FRNDINT - celá část reálného čísla
- FRINEAR - zaokrouhlení na celé číslo
- FRINT2 - zaokrouhlení na celé číslo se znaménkem
- FRICHOP - odříznutí reálné části
- FPREM - zbytek dělení registrem ST(1)
- FPREM1 - zbytek dělení registrem ST(1) se skrytým nejvyšším bitem

- FXTRACT - uloží mantisu ST(0) a exponent ST(1)
- FSCALE - vynásobení ST(0) s 2 na ST(1)
- FSIN - sin(ST0)
- FCOS - cos(ST0)
- FSINCOS - v ST0 cos(původní ST0) a ST1 sin(původní ST0)
- FPTAN - ST0 = tan(ST0), ST1 = 1
- FPATAN - ST0 = (arctan(ST1) / ST0)
- F2XM1 - (2 na ST0) - 1
- FYL2X - ST1 * log2(ST0)
- FYL2XP1 - ST1 * log2(ST0+1)

Aritmetické instrukce numerického koprocesoru

FADD (Floating ADDition) : FADD dst, src

součet reálných čísel

K cílovému operandu (registr numerického koprocesoru) přičte zdrojový operand (reálné číslo). Výsledek uloží na místo cílového operandu.

příznaky: mění registr SWR kromě pole TOP

FADDP (Floating ADDition and Pop) : FADDP dst, st

součet reálných čísel s odstraněním ze zásobníku

K cílovému operandu (registr numerického koprocesoru) přičte obsah vrcholu zásobníku registrů numerického koprocesoru. Výsledek uloží na místo cílového operandu a dekrementuje ukazatel vrcholu zásobníku numerického koprocesoru.

formát: 1101 1110 1100 0 ST(i) (de c)

funkce: (ST(i))+(ST)=>ST(i); (TOP)-1=>TOP

příznaky: mění registr SWR numerického koprocesoru

FIADD (Floating and Integer ADDition) : FIADD src

součet reálného a celého čísla

K obsahu vrcholu zásobníku registrů numerického koprocesoru přičte celočíselný operand (se znaménkem). Výsledek uloží na vrchol zásobníku numerického koprocesoru.

formát: 1101 1W10 mod 000 r/m [displ [disph]]

funkce:

- W=0, (ST)+data16=>ST;
- W=1, (ST)+data32=>ST;

příznaky: mění registr SWR kromě pole TOP

FSUB (Floating SUBtraction) : FSUB dst, src

odečti reálná čísla

Od cílového operandu (registr numerického koprocesoru) odečte zdrojový operand (reálné číslo). Výsledek uloží na místo cílového operandu.

příznaky: mění registr SWR kromě pole TOP

FSUBP (Floating SUBtraction and Pop) : FSUBP dst, st

rozdíl reálných čísel s odstraněním ze zásobníku

Od vrcholu zásobníku registrů numerického koprocesoru odečte operand (registr numerického koprocesoru). Výsledek uloží na místo operandu a dekrementuje ukazatel vrcholu zásobníku numerického koprocesoru.
formát: 1101 1110 1110 1 ST(i) (de e)
funkce: (ST)-(ST(i))=>ST(i); (TOP)-1=>TOP
příznaky: mění registr SWR numerického koprocesoru

FSUBR (Floating SUBtraction Reverse) : FSUB src, dst

opačně odečti reálná čísla
Od cílového operandu (reálné číslo) odečte zdrojový operand (registr numerického koprocesoru). Výsledek uloží na místo cílového operandu.
příznaky: mění registr SWR kromě pole TOP

FSUBRP (Floating SUBtraction Reverse and Pop) : FSUBRP st, dst

opačně odečti reálná čísla s odstraněním ze zásobníku
Od operandu (registr numerického koprocesoru) odečte obsah vrcholu zásobníku registrů numerického koprocesoru. Výsledek uloží na místo operandu a dekrementuje ukazatel vrcholu zásobníku numerického koprocesoru.
formát: 1101 1110 1110 0 ST(i) (de e)
funkce: (ST(i))-(ST)=>ST(i); (TOP)-1=>TOP
příznaky: mění registr SWR numerického koprocesoru

FISUB (Floating and Integer SUBtraction) : FISUB src

odečti celé číslo od reálného
Od obsahu vrcholu zásobníku registrů numerického koprocesoru odečte celočíselný operand (se znaménkem). Výsledek uloží na vrchol zásobníku numerického koprocesoru.
formát: 1101 1W10 mod 100 r/m [displ [disph]]
funkce:

- W=0, (ST)-data16=>ST;
- W=1, (ST)-data32=>ST;

příznaky: mění registr SWR kromě pole TOP

FISUBR (Floating and Integer SUBtraction Reverse) : FISUBR src

odečti reálné číslo od celého
Od operandu (celé číslo se znaménkem) odečte obsah vrcholu zásobníku registrů numerického koprocesoru. Výsledek uloží na vrchol zásobníku numerického koprocesoru.
formát: 1101 1W10 mod 101 r/m [displ [disph]]
funkce:

- W=0, data16-(ST)=>ST;
- W=1, data32-(ST)=>ST;

příznaky: mění registr SWR kromě pole TOP

FCHS (Floating CHange Sign) : FCHS

změna znaménka reálného čísla

Změní znaménko obsahu vrcholu zásobníku registrů numerického koprocesoru.
formát: 1101 1001 1110 0000 (d9 e0)
funkce: -(ST)=>ST;
příznaky: nemění

FTST (Floating TeST) : FTST

testuj reálné číslo
Na základě hodnoty vrcholu zásobníku numerického koprocesoru nastaví příznaky C0, C2 a C3 registru SWR numerického procesoru.
formát: 1101 1001 1110 0100 (d9 e4)
funkce:

- (ST)=Nan, 1=>C3; 1=>C2; 1=>C0;
- jinak, 0=>C2;
  - (ST)>0, 0=>C3; 0=>C0;
  - (ST)=0, 1=>C3; 0=>C0;
  - (ST)<0, 0=>C3; 1=>C0;

příznaky: mění bity C3,C2 a C0 registru SWR

FXAM (Floating eXAMine) : FXAM

zkus reálné číslo
Na základě hodnoty vrcholu zásobníku numerického koprocesoru nastaví příznaky C3 až C0 registru SWR numerického procesoru.
formát: 1101 1001 1110 0101 (d9 e5)příznaky: mění bity C3 až C0 registru SWR

FCOM (Floating COMpare) : FCOM src

porovnej reálná čísla
Nastaví bity registru SWR na základě porovnání hodnoty vrcholu zásobníku numerického koprocesoru s operandem. Generuje požadavek o přerušení v případě, že jeden z operandů má hodnotu NaN (Not a Number). (Instrukce FUCOM přerušení negeneruje).
příznaky: mění bity C3, C2 a C0 registru SWR

FCOMP (Floating COMpare and Pop) : FCOMP src

porovnej reálná čísla s odstraněním ze zásobníku
Nastaví bity registru SWR na základě porovnání hodnoty vrcholu zásobníku numerického koprocesoru s operandem. Generuje požadavek o přerušení v případě, že jeden z operandů má hodnotu NaN (Not a Number). (Instrukce FUCOMP přerušení negeneruje). Po porovnání dekrementuje ukazatel vrcholu zásobníku.
příznaky: mění bity C3, C2, C0 a TOP registru SWR

FCOMPP (Floating COMpare and Pop and Pop) : FCOMPP

porovnej reálná čísla s dvojnásobným odstraněním ze zásobníku
Nastaví bity registru SWR na základě porovnání hodnoty vrcholu zásobníku numerického koprocesoru s registrem ST1 numerického koprocesoru. Generuje požadavek o přerušení v případě, že jeden z operandů má hodnotu NaN (Not a Number). (Instrukce FUCOMPP přerušení negeneruje). Po porovnání dvakrát dekrementuje ukazatel vrcholu zásobníku.
příznaky: mění bity C3, C2, C0 a TOP registru SWR

FICOM (Floating and Integer COMpare) : FICOM src

porovnej reálné a celé číslo
Nastaví bity registru SWR na základě porovnání hodnoty vrcholu zásobníku numerického koprocesoru s operandem. Generuje požadavek o přerušení v případě, že jeden z operandů má hodnotu NaN (Not a Number).
příznaky: mění bity C3, C2 a C0 registru SWR

FICOMP (Floating and Integer COMpare and Pop) : FICOMP src
porovnej reálné a celé číslo s odstraněním ze zásobníku
Nastaví bity registru SWR na základě porovnání hodnoty vrcholu zásobníku numerického koprocesoru s operandem (celé číslo se znaménkem). Generuje požadavek o přerušení v případě, že jeden z operandů má hodnotu NaN (Not a Number). Po porovnání dekrementuje ukazatel vrcholu zásobníku.
příznaky: mění bity C3, C2 a C0 registru SWR

FCOMI (Floating COMpare setting Integer flags) : FCOMI st, src
porovnej reálná čísla s nastavením celočíselných příznaků
Nastaví příznakové bity na základě porovnání hodnoty vrcholu zásobníku numerického koprocesoru s operandem (registr zásobníku numerického koprocesoru). Generuje požadavek o přerušení v případě, že jeden z operandů má hodnotu NaN (Not a Number). (Instrukce FUCOMI přerušení negeneruje).
formát: 1101 1011 1111 0 ST(i) příznaky: mění příznaky CF, ZF, PF a SF

FCOMIP (Floating COMpare setting Integer flags and Pop) : FCOMIP st, src
porovnej reálná čísla s nastavením celočíselných příznaků s odstraněním ze zásobníku
Nastaví příznakové bity na základě porovnání hodnoty vrcholu zásobníku numerického koprocesoru s operandem (registr zásobníku numerického koprocesoru). Generuje požadavek o přerušení v případě, že jeden z operandů má hodnotu NaN (Not a Number). (Instrukce FUCOMIP přerušení negeneruje). Po porovnání dekrementuje ukazatel vrcholu zásobníku.
formát: 1101 1111 1111 0 ST(i) příznaky: mění příznaky CF, ZF, PF a SF

FMUL (Floating MULtiplication) : FMUL dst, src
vynásob reálná čísla
Vynásobí cílový operand (registr numerického koprocesoru) se zdrojovým operandem (reálné číslo). Výsledek uloží na místo cílového operandu.
příznaky: mění registr SWR kromě pole TOP

FMULP (Floating MULtiplication and Pop) : FMULP dst, st
vynásob reálná čísla s odstraněním ze zásobníku
Vynásobí cílový operand (registr numerického koprocesoru) s obsahem vrcholu zásobníku numerického koprocesoru. Výsledek uloží na místo cílového operandu a dekrementuje ukazatel vrcholu zásobníku numerického koprocesoru.
formát: 1101 1110 1100 1 ST(i)
funkce: (ST(i))*(ST)=>ST(i); (TOP)-1=>TOP
příznaky: mění registr SWR

FIMUL (Floating and Integer MULtiplication) : FIMUL src
vynásob reálné a celé číslo
Obsahu vrcholu zásobníku registrů numerického koprocesoru vynásobí celočíselným operandem (se znaménkem). Výsledek uloží na vrchol zásobníku numerického koprocesoru.
formát: 1101 1W10 mod 001 r/m [displ [disph]]
funkce:

- W=0, (ST)*data16=>ST;
- W=1, (ST)*data32=>ST;

příznaky: mění registr SWR kromě pole TOP

FDIV (Floating DIVision) : FDIV dst, src

vyděl reálná čísla

Vydělí cílový operand (registr numerického koprocesoru) se zdrojovým operandem (reálné číslo). Výsledek uloží na místo cílového operandu.

příznaky: mění registr SWR kromě pole TOP

FDIVP (Floating DIVision and Pop) : FDIVP st, src

vyděl reálná čísla s odstraněním ze zásobníku

Vydělí vrchol zásobníku numerického koprocesoru operandem (registr numerického koprocesoru). Výsledek uloží na místo cílového operandu a dekrementuje ukazatel vrcholu zásobníku numerického koprocesoru.

formát: 1101 1110 1111 1 ST(i)

funkce: (ST)/(ST(i))=>ST(i); (TOP)-1=>TOP

příznaky: mění registr SWR

FDIVR (Floating DIVision Reverse) : FDIVR dst, src

opačně vyděl reálná čísla

Vydělí zdrojový operand (reálné číslo) registrem numerického koprocesoru. Výsledek uloží na místo cílového operandu.

příznaky: mění registr SWR kromě pole TOP

FDIVRP (Floating DIVision Reverse and Pop) : FDIVRP st, dst

opačně vyděl reálná čísla s odstraněním ze zásobníku

Vydělí cílový operand (registr numerického koprocesoru) obsahem vrcholu zásobníku numerického koprocesoru. Výsledek uloží na místo cílového operandu a dekrementuje ukazatel vrcholu zásobníku numerického koprocesoru.

formát: 1101 1110 1111 0 ST(i)

funkce: (ST(i))/(ST)=>ST(i); (TOP)-1=>TOP

příznaky: mění registr SWR

FIDIV (Floating and Integer DIVision) : FIDIV src

vyděl reálné číslo celým číslem

Obsahu vrcholu zásobníku registrů numerického koprocesoru vydělí celočíselným operandem (se znaménkem). Výsledek uloží na vrchol zásobníku numerického koprocesoru.

formát: 1101 1W10 mod 110 r/m [displ [disph]]

funkce:

- W=0, (ST)/data16=>ST;
- W=1, (ST)/data32=>ST;

příznaky: mění registr SWR kromě pole TOP

FIDIVR (Floating and Integer DIVision Reverse) : FIDIVR src

vyděl celé číslo reálným

Operand (celé číslo se znaménkem) vydělí obsahem vrcholu zásobníku registrů numerického koprocesoru. Výsledek uloží na vrchol zásobníku numerického koprocesoru.
formát: 1101 1W10 mod 111 r/m [displ [disph]]
funkce:

- W=0, data16/(ST)=>ST;
- W=1, data32/(ST)=>ST;

příznaky: mění registr SWR kromě pole TOP

Instrukce přesunu dat numerického koprocesoru

FLD (Floating LoaD) : FLD src
načti reálné číslo
Zdrojový operand (reálné číslo) uloží (přepíše) na vrchol zásobníku numerického koprocesoru.
příznaky: nemění

FILD (Floating Integer LoaD) : FILD src
načti celé číslo
Zdrojový operand (celé číslo se znaménkem) uloží (přepíše) na vrchol zásobníku numerického koprocesoru.
příznaky: nemění

FST (Floating STore) : FST dst
ulož reálné číslo
Do cílového operandu (reálné číslo) uloží obsah vrcholu zásobníku numerického koprocesoru.
příznaky: nemění

FSTP (Floating STore and Pop) : FSTP dst
ulož reálné číslo s odebráním ze zásobníku
Do cílového operandu (reálné číslo) uloží obsah vrcholu zásobníku numerického koprocesoru a dekrementuje ukazatel vrcholu zásobníku numerického koprocesoru.
příznaky: mění pole TOP registru SWR

FIST (Floating Integer STore) : FIST dst
ulož celé číslo
Do cílového operandu (celé číslo se znaménkem) uloží obsah vrcholu zásobníku numerického koprocesoru.
příznaky: nemění

FISTP (Floating Integer STore and Pop) : FISTP dst
ulož celé číslo s odebráním ze zásobníku
Do cílového operandu (celé číslo se znaménkem) uloží obsah vrcholu zásobníku numerického koprocesoru a dekrementuje ukazatel vrcholu zásobníku numerického koprocesoru.
příznaky: mění pole TOP registru SWR

FXCH (Floating eXCHange) : FXCH src
zaměň reálná čísla
Záměna obsahu vrcholu zásobníku numerického koprocesoru se zdrojovým operandem.
příznaky: mění pole bitů ST registru SWR
FLDCW (Floating LoaD Control Word) : FLDCW src
načti řídící slovo
Zdrojový operand bude uložen do registru CWR numerického koprocesoru.
formát: 1101 1001 mod 101 r/m [displ [disph]]
funkce: (EA)=>CWR
příznaky: nemění
FSTCW (Floating STore Control Word) : FSTCW dst
ulož řídící slovo
Cílový operand bude naplňen registrem CWR numerického koprocesoru.
formát: 1101 1001 mod 111 r/m [displ [disph]]
funkce: CWR=>(EA)
příznaky: nemění
FSTSW (Floating STore Status Word) : FSTSW dst
ulož stavové slovo
Cílový operand bude naplňen registrem SWR numerického koprocesoru.
příznaky: nemění
FSTENV (Floating STore ENVironment) : FSTENV dst
ulož prostředí
Uložení prostředí (registry) koprocesoru do paměti.
formát: 1101 1001 mod 110 r/m [displ [disph]] příznaky: nemění
FLDENV (Floating LoaD ENVironment) : FLDENV src
načtení prostředí
Obnovení prostředí (registry) koprocesoru z paměti.
formát: 1101 1001 mod 100 r/m [displ [disph]] příznaky: mění SWR
FSAVE (Floating SAVE status) : FSAVE dst
ulož stav
Uložení stavu koprocesoru do paměti. Generuje instrukci WAIT.
formát: 1101 1101 mod 110 r/m [displ [disph]] příznaky: nemění
FRSTOR (Floating ReSTORe status) : FRSTOR src
obnovení stavu
Obnovení stavu koprocesoru z paměti.
formát: 1101 1101 mod 100 r/m [displ [disph]] příznaky: mění SWR
FCMOVcc (Floating Conditional MOVe if ...) : FCMOVcc st, src
podmíněný přesun reálného čísla

Nové instrukce podmíněných přesunů poprvé v Pentium Pro.

- FCMOVA - přesun, je-li nad (CF=0 nebo ZF=0)
- FCMOVAE - přesun, je-li nad nebo rovno (CF=0)
- FCMOVB - přesun, je-li pod (CF=1)
- FCMOVBE - přesun, je-li pod nebo rovno (CF=1 a ZF=1)
- FCMOVC - přesun, je-li přenos (CF=1)

- FCMOVE - přesun, je-li rovno (ZF=1)
- FCMOVNA - přesun, není-li nad (CF=1 a ZF=1)
- FCMOVNAE - přesun, není-li nad nebo rovno(CF=1)
- FCMOVNB - přesun, není-li menší (CF=0)
- FCMOVNBE - přesun, není-li menší nebo rovno (CF=0 nebo ZF=0)
- FCMOVNC - přesun, není-li přenos (CF=0)
- FCMOVNE - přesun, není-li rovno (ZF=0)
- FCMOVNP - přesun, není-li parita (PF=0)
- FCMOVNU - přesun, není-li neuspořádané (PF=0)
- FCMOVNZ - přesun, není-li nula (ZF=0)
- FCMOVP - přesun, je-li parita (PF=1)
- FCMOVPE - přesun, je-li sudá parita (PF=1)
- FCMOVPO - přesun, je-li lichá parita (PF=0)
- FCMOVU - přesun, je-li neuspořádané (PF=1)
- FCMOVZ - přesun, je-li nula (ZF=1)

příznaky: nemění

Řídící instrukce numerického koprocesoru

FFREE (Floating register FREE) : FFREE dst
uvolni registr plovoucí řádové čárky
Označí operand jako nepoužitý.
formát: 1101 1101 1100 0 ST(i)
funkce: 3=>TAG ST(i)
příznaky: nemění

FINCSTP (Floating INCrement STack Pointer) : FINCSTP
inkrementuj ukazatel zásobníku numerického koprocesoru
Inkrementace (navýšení) ukazatele vrcholu zásobníku numerického koprocesoru (pole TOP).
formát: 1101 1001 1111 0111 (d9 f7)
funkce: (TOP)+1=>TOP
příznaky: mění pole TOP registru SWR

FDECSTP (Floating DECrement STack Pointer) : FDECSTP
dekrementuj ukazatel zásobníku numerického koprocesoru
Dekrementace (snížení) ukazatele vrcholu zásobníku numerického koprocesoru (pole TOP).
formát: 1101 1001 1111 0110 (d9 f6)
funkce: (TOP)-1=>TOP
příznaky: mění pole TOP registru SWR

FINIT (Floating INITialize) : FINIT
inicializace numerického koprocesoru
Inicializace numerického koprocesoru před jeho použitím pro výpočty. Zároveň před sebou generuje instrukci WAIT. (Instrukce FNINIT instrukci WAIT negeneruje).

formát: 1101 1011 1110 0011 (db e3)
funkce: (SWR)and 0x4700=>SWR; 0x03f=>CWR; 0xffff=>TAG;
příznaky: mění registr SWR

FWAIT (Floating WAIT) : FWAIT

čekání na jednotku numerického koprocesoru
Neprovede žádnou činnost. Uvede procesor do stavu WAIT a čeká na signál TEST/BUSYod numerického koprocesoru.
formát: 1001 1011 (9b)příznaky: nemění

FNOP (Floating No Operation) : FNOP

žádná operace jednotky numerického koprocesoru
Neprovede žádnou činnost. Zavede zpoždění.
formát: 1101 1001 1101 0000 (d9 d0)
funkce: žádná
příznaky: nemění

INSTRUKCE MMX

Datové typy

Instrukce MMX využívají nové datové typy: komprimované bajty (packed byte, přípona instrukce **b**), komprimovaná slova (packed word, přípona **w**), komprimovaná dvojslova (packed doubleword, přípona **d**) a 64-bitová slova (quadword, přípona **q**).
Jedněch a těch samých 64-bitů může jedna MMX instrukce chápat jako 8 bajtů, jiná jako 4 slova atd. Datový typ je určen příponou instrukce.
Vstupní operand se může nacházet v MMX registru nebo v paměti
Výstupní operand musí být v MMX registru.

Cyklická aritmetika (wraparound arithmetic)

Pokud instrukce využívá cyklickou aritmetiku a výsledek operace je větší než povolený rozsah, pak jsou nejvyšší bity výsledku ”vytlačeny” a nejsou ve výsledku přítomny. Jinak řečeno, jestliže výsledek převyšuje maximální možnou hodnotu o n, bude výsledkem minimální hodnota + n –1.
Příklady:

```
7FFFh + 0002h = 8001h
(32767 + 2 = -32767 nebo 32769)
FFFFh + 0002h = 0001h
(65535 + 2 = 1 nebo -1 + 2 = 1)
```

Aritmetika s nasycením (saturation arithmetic)

Pokud instrukce využívá aritmetiku s nasycením a výsledek operace převyšuje maximální povolenou hodnotu číselného rozsahu (nebo je menší, než minimální hodnota), tak se za výsledek dosadí právě hraniční hodnota. (proběhne ”nasycení”).
Například, pokud je výsledek menší, než 8000h, pak se 16-bitové slovo se znaménkem bude po nasycení rovnat oněm 8000h. Pokud bude větší než 7FFFh, pak se bude toto číslo rovnat po nasycení 7FFFh.

Instrukce MMX probíhají ve stejné části procesoru, jako příkazy pro matematiku s desetinnými čísly. Proto při provádění všech instrukcí MMX (kromě EMMS) dochází

k narušení dat v registrech pro práci s desetinnými čísly. Instrukce `EMMS` zabezpečuje přechod procesoru od režimu instrukcí MMX do režimu výpočtů s desetinnou čárkou. Příkazem `EMMS` ukončujte všechny části vašeho programu, které volají instrukce MMX. Jestli to neuděláte, tak:
operace s desetinnými čísly budou dávat nesprávné výsledky
někdy se při operacích s desetinnými čísly
objeví výjimky typu **Stack overflow (přetečení zásobníku)**.
Vpravo nahoře vidíte fragment programu, ve kterém hned za instrukcí `movq` následuje volání funkce používající operace s desetinnou čárkou. Ve výsledku vzniká výjimka **Stack overflow**. Dole je ukázán stejný fragment v kterém je vložena instrukce `EMMS`. Zde k přeplnění zásobníku nedojde.

Instrukce EMMS
Instrukce EMMS dosadí hodnotu 1 všem bitům stavového slova registrům pro desetinná čísla (registrům matematického koprocesoru neboli registrům FPU). Takový stav má význam ”prázdný” (empty). To zajišťuje přechod procesoru z režimu práce s technologií MMX do režimu výpočtů s desetinnými čísly. Nezapomínejte vkládat instrukci EMMS na konce procedur používajících instrukce MMX!

Instrukce movq
Instrukce `movq` kopíruje 64 bitů z jednoho MMX registru do druhého nebo z MMX registru do paměti či naopak.

Instrukce pmaddwd
Instrukce `pmaddwd` násobí jednotlivá 16-bitová slova (se znaménkem) ze vstupního a výstupního operandu, což dává čtyři 32-bitové součiny. Poté se první součin sečte s druhým a třetí se čtvrtým. Získané součty ze zapíší do 32-bitových dvojslov výstupního operandu.

Instrukce rodiny padd (cyklická aritmetika)
Instrukce `padd` sčítají vstupní a výstupní operand. Mohou to být bajty, slova nebo dvojslova. Jestli výsledek přesahuje povolený rozsah výsledku, tak v souladu s pravidly cyklické aritmetiky dochází k “protočení výsledku” jako u běžné instrukce `add` procesoru x86.
Platné instrukce: `paddb`, `paddw`, `paddd`.

Instrukce rodiny padds (aritmetika s nasycením, data se znaménkem)
Instrukce `padds` sčítají vstupní a výstupní operand. Mohou to být bajty, slova nebo dvojslova Jestli výsledek přesahuje povolený rozsah výsledku, tak se za výsledek dosadí hraniční hodnota.
Platné instrukce: `paddsb`, `paddsw`.

Instrukce rodiny paddus (aritmetika s nasycením, data bez znaménka)
Instrukce `paddus` sčítají vstupní a výstupní operand. Mohou to být bajty, slova nebo dvojslova Jestli výsledek přesahuje povolený rozsah výsledku, tak se za výsledek dosadí hraniční

hodnota.
Platné instrukce: `paddusb`, `paddusw`.

Instrukce rodiny psub (cyklická aritmetika)
Instrukce `psub` odčítají data (bajty, slova, dvojslova) vstupního operandu od výstupního operandu. (výstup = výstup - vstup) Jestli výsledek přesahuje povolený rozsah výsledku, tak v souladu s pravidly cyklické aritmetiky dochází k "protočení výsledku" jako u běžné instrukce `sub` procesoru x86.
Platné instrukce: `psubb`, `psubw`, `psubd`.

Instrukce rodiny psubs (aritmetika s nasycením, data se znaménkem)
Instrukce `psubs` odčítají data (bajty, slova, dvojslova) vstupního operandu od výstupního operandu. (výstup = výstup - vstup) Jestli výsledek přesahuje povolený rozsah výsledku, , tak se za výsledek dosadí hraniční hodnota.
Platné instrukce: `psubsb`, `psubsw`.

Instrukce rodiny psubus (aritmetika s nasycením, data bez znaménka)
Instrukce `psubus` odčítají data (bajty, slova, dvojslova) vstupního operandu od výstupního operandu. (výstup = výstup - vstup) Jestli výsledek přesahuje povolený rozsah výsledku, , tak se za výsledek dosadí hraniční hodnota.
Platné instrukce: `psubusb`, `psubusw`.

Instrukce rodiny psll (logický posun vlevo)
Instrukce `psll` provádějí posun všech podporovaných datových typů (16-, 32- nebo 64-bitového slova) ve výstupním operandu doleva o počet bitů určených ve vstupním operandu. Uvolněné bity vpravo se zaplňují nulami.
Platné instrukce: `psllw`, `pslld`, `psllq`.

Instrukce rodiny psra (aritmetický posun vpravo)
Instrukce `psra` provádějí posun všech podporovaných datových typů (16-, 32- nebo 64-bitového slova) ve výstupním operandu doprava o počet bitů určených ve vstupním operandu. Pokud se posouvá kladné číslo, tak se uvolněné bity vlevo zaplňují nulami. Pokud je záporné, tak se zaplňují jedničkami.
Platné instrukce: `psraw`, `psrad`.

Instrukce rodiny psrl (logický posun vpravo)
Instrukce `psrl` provádějí posun všech podporovaných datových typů (16-, 32- nebo 64-bitového slova) ve výstupním operandu doprava o počet bitů určených ve vstupním operandu. Uvolněné bity vpravo se zaplňují nulami.
Platné instrukce: `psrlw`, `psrld`, `psrlq`.

Instrukce pcmpeqb
Instrukce `pcmpeqb` porovnává bajty vstupního operandu s odpovídajícími bajty výstupního operandu. Pokud je bajt výstupního operandu roven bajtu vstupního, zaplní se takový bajt jedničkami. Pokud není roven, zaplní se nulami.

Instruce pand
Instrukce `pand` provádí bitový logický součin (operace AND) vstupního a výstupního operandu. Při této operaci zůstane hodnota 1 pouze těm bitům výstupního operandu, na jejichž místě je u vstupního operandu také 1. Ve všech ostatních případech se daný bit vynuluje.

Instrukce pandn (logické NOT a AND)
Instrukce `pandn` napřed provede logické NOT cílového operandu(jinak řečeno změní jeho znaménko) a potom logické AND mezi cílovým operandem a zdrojovým operandem.

Instrukce por
Instrukce `por` provádí bitový logický součet (operace OR) vstupního a výstupního operandu. Při této operaci se hodnota 1 zapíše do těch bitů výstupního operandu, u kterých byla hodnota 1 předtím anebo byla hodnota 1 v odpovídajícím bitu vstupního operandu. Hodnota 0 bude výsledkem pouze, pokud je nulový jak daný bit výstupního operandu, tak i příslušný bit operandu vstupního.

Instrukce pxor (logické XOR)
Instrukce `pxor` provede logickou nonekvivalenci svých operandů.

Instrukce pmaddwd (cyklická aritmetika)
Instrukce `pmaddwd` postupně provádějí součin operandů se znaménkem po 16-bitových párech, což dává čtyři 32-bitové mezivýsledky se znaménkem. Potom se první mezivýsledek sečte s druhým a třetí se čtvrtým. Výsledky se zapíší do dvou 32-bitových slov výstupního 64-bitového operandu. Jestliže všechny 16-bitová slova na vstupu budou rovny 8000h, bude výsledkem 80000000h (to je jedinný případ, kdy násobení záporných čísel dá *záporný* výsledek).

Instrukce pmulhw (cyklická aritmetika)
Instrukce `pmulhw` postupně provádějí součin operandů se znaménkem po 16-bitových párech, což dává čtyři 32-bitové mezivýsledky se znaménkem. Vyšších 16 bitů z každého mezivýsledku se zapisuje do výstupního operandu. Nižších 16 bitů z každého mezivýsledku se ztrácí.

Instrukce pmullw (cyklická aritmetika)
Команда `pmullw` postupně provádějí součin operandů se znaménkem po 16-bitových párech, což dává čtyři 32-bitové mezivýsledky se znaménkem. Nižších 16 bitů z každého mezivýsledku se zapisuje do výstupního operandu. Vyšších 16 bitů z každého mezivýsledku se ztrácí.

Instrukce rodiny pcmpeq
Instrukce `pcmpeq` párově porovnávají výstupní a vstupní operand (bajty, 16- nebo 32-bitová slova). Pokud je daný pár (vstupní operand/úsek výstupního operandu) shodný, je daný úsek výstupního operandu vyplněn jedničkami. Pokud ne, je tento úsek vyplněn nulami.
Platné instrukce: `pcmpeqb`, `pcmpeqw`, `pcmpeqd`.

MMX instrukce komprese dat pakují dlouhé datové typy (16- nebo 32-bitová slova) do kratších. Pokud se spakovaná forma nevejde do číselného rozsahu cílového typu, proběhne "nasycení" – za výsledek se dosadí hraniční hodnota přípustného rozsahu hodnot výstupního typu.
Instrukce dekomprese po párech zpracovávají vstupní data z obou operandů a rozpakují je do delších typů výstupního operandu. Tyto instrukce lze využít pro zvýšení počtu významných bitů při výpočtech.

Instrukce rodiny packss
Instrukce `packss` pakuje dlouhé datové typy (16 nebo 32-bitové typy se znaménkem) na kratší. (bajty nebo 16-bitová slova se znaménkem) Pokud byl vstupní typ mimo přípostný rozsah hodnot výstupního typu, pak se za výsledek komprese dosadí bližší hraniční hodnota přípustného rozsahu.
Platné instrukce: `packsswb`, `packssdw`.

instrukce packuswb
Instrukce `packuswb` pakuje 16-bitová slova se znaménkem z obou operandů na bajty bez znaménka a zapisuje je do výstupního operandu.
Pokud je vstupní slovo větší než FFh, dosadí se za výsledek FFh. Pokud je vstupní slovo se znaménkem záporné, dosadí se za výsledek 00h.

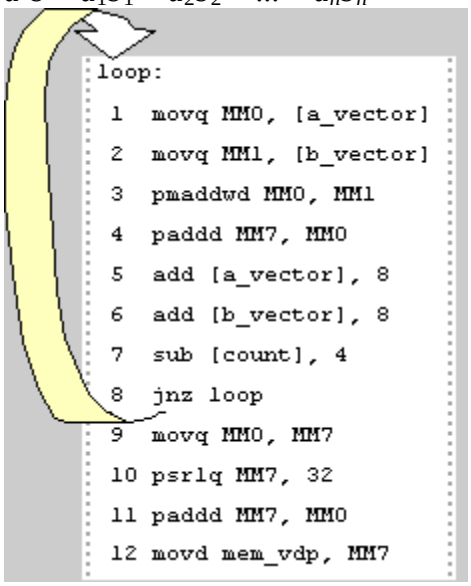
Instrukce rodiny punpckh
Instrukce `punpckh` po párech spojují vstupní data (bajty, 16 nebo 32-bitová slova), která se nacházejí ve vyšších 32 bitech obou operandů. Výsledkem jsou delší datové typy, které se zapisují do výstupního operandu. Spodních 32 bitů ve vstupních operandech nemá na výsledek vliv.
Platné instrukce: `punpckhbw`, `punpckhwd`, `punpckhdq`.

Instrukce rodiny punpckl
Instrukce `punpckl` po párech spojují vstupní data (bajty, 16 nebo 32-bitová slova), která se nacházejí v nižších 32 bitech obou operandů. Výsledkem jsou delší datové typy, které se zapisují do výstupního operandu. Spodních 32 bitů ve vstupních operandech nemá na výsledek vliv.
Platné instrukce: `punpcklbw`, `punpcklwd`, `punpckldq`.

Instrukce movd
Instrukce `movd` kopíruje 32 bitů:
z nižší poloviny registru MMX do nižší poloviny jiného MMX registru. POZOR – vyšších 32 bitů výstupního MMX registru se přitom zaplní nulami.
z paměti nebo z obecného registru do nižší poloviny MMX registru. POZOR – vyšších 32 bitů výstupního MMX registru se přitom zaplní nulami.
z nižší poloviny MMX registru do paměti nebo do obecného registru.
Instrukce movq
Instrukce `movq` kopíruje 64 bitů:
z jednoho registru MMX do jiného MMX registru

z paměti do MMX registru
z registru MMX do paměti

Příklady:
Skalární součin vektorů
Tato operace se často využívá v úlohách lineární algebry a digitálního zpracování signálů. Skalární součin $a \cdot b$ vektorů a_k a b_k ($k = 1,..., n$) se vypočítá:
$a \cdot b = a_1 b_1 + a_2 b_2 + ... + a_n b_n$



Absolutní hodnoty rozdílů čísel se využívají jako vzdálenost na číselné ose v kompresních a algoritmech a v analýzách dat. MMX instrukce vám pomůžou tyto veličiny rychle vypočítat. Myšlenka programu je jednoduchá:

1. Zkopírovat elementy vstupních vektorů z paměti do MMX registrů pomocí movq.
2. Odečíst elementy druhého vektoru z elementů prvního pomocí aritmetiky s nasycením, přičemž na ně budem pohlížet jako na čísla bez znaménka. Pokud je rozdíl záporný, dostane příslušný element výstupního operandu hodnotu nula. Pokud je rozdíl kladný, dostane se nezměněm do výstupního operandu.
3. Stejným způsobem odečíst elementy prvního vektoru od elementů druhého. Tam, kde se v předchozím případě objevily nuly, teď budou nezáporné hodnoty.
4. Provést bitové logické OR výsledků kroků 2 a 3. Tímto způsobem se vybere nenulový (tedy kladný) výsledek z dvojice. Pouze, pokud jsou jsou obě varianty dvojice nulové, bude výsledkem nula.