

Wpływa na flagi: ----C
Wymagane: ADX

Instrukcja ADCX

adcx cel, źródło

Dodaje bez znaku zawartość źródła, celu i przeniesienia, sumę umieszcza w miejscu przeznaczenia (cel – rejestr 32|64 bitowy).

cel:=cel+źródło+CF

adcx eax, zmienna
adcx edx, [ebx+esi*4]
adcx rcx, rbx

Wpływa na flagi: O----Wymagane: ADX

Instrukcja ADOX

adox cel, źródło

Dodaje bez znaku zawartość źródła, celu i flagi
przepełnienia, sumę umieszcza w miejscu przeznaczenia
(cel – rejestr 32 | 64 bitowy).

cel:=cel+źródło+OF

adox eax, zmienna
adox edx, [ebx+esi*4]
adox rcx, rbx

Instrukcja SUB

sub cel, źródło
Odejmuje zawartość źródła od celu, różnicę umieszcza w miejscu przeznaczenia (cel).
cel:=cel-źródło
sub ecx, zmienna
sub ebx, [ebx+esi*4]
sub rax, rdx

Uwaga:
Jeśli źródło jest adresowane w trybie prostym może mieć do 32 bitów.

Instrukcja SBB

sbb cel, źródło
Odejmuje zawartość źródła od celu z uwzględnieniem pożyczki, różnicę umieszcza w miejscu przeznaczenia (cel).

cel:=cel-(źródło+CF)

sbb edx, zmienna
sbb eax, [ebx+esi*4]
sbb rax, rdx

Uwaga:
Jeśli źródło jest adresowane w trybie prostym może mieć do 32 bitów.

(C) IISI d.KIK PC2 2019 Programowanie niskopoziomowe

Instrukcja MUL

mul źródło

Mnoży bez znaku zawartość akumulatora (al, ax, eax, rax) i źródła, wynik umieszcza w miejscu przeznaczenia (ax, dx:ax, edx:eax, rdx:rax). Flagi CF i OF są zerem, jeśli starsza połowa bitów wyniku jest równa zero.

wynik:=acc*źródło

mul zmienna
mul word ptr[ebx+esi*4]

Wpływa na flagi:

Wymaga BMI2

Instrukcja MULX

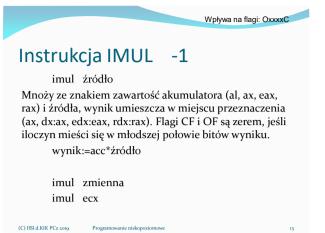
mulx celı,cel2,źródło

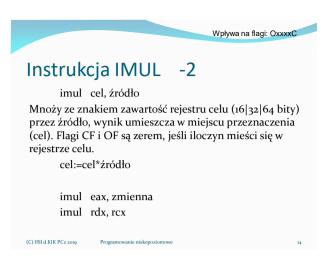
Mnoży bez znaku zawartość rejestru edx|rdx i źródła, wynik umieszcza w rejestrach celu(celı:cel2).

celı:cel2:=e(r)dx*źródło

mulx ecx, ebx, [tab+esi*4]

mulx rcx, rbx, rax

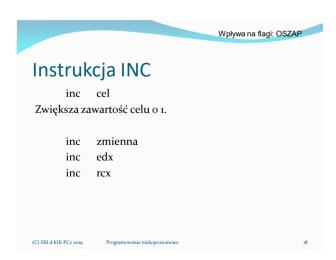


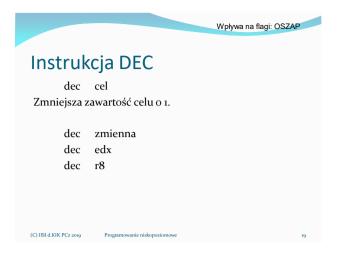


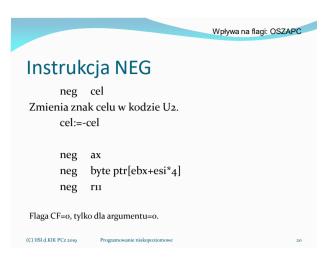
Instrukcja IMUL -3 imul cel, źródło1, źródło2 Mnoży ze znakiem zawartość źródła1 (16|32|64 bity) przez źródło2 (stała), wynik umieszcza w miejscu przeznaczenia (cel). Flagi CF i OF są zerem, jeśli iloczyn mieści się w rejestrze celu. cel:=źródło1*źródło2 imul eax, zmienna, 5 imul cx, dx, 77 Uwaga: Jeśli źródło2 jest adresowane w trybie prostym może mieć do 32 bitów. (C) IISI dXIK PCZ 2009 Programowanie niskopoziomowe







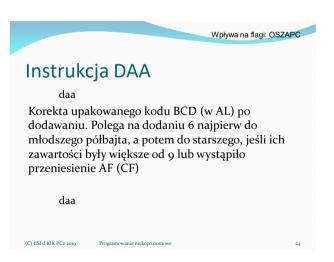


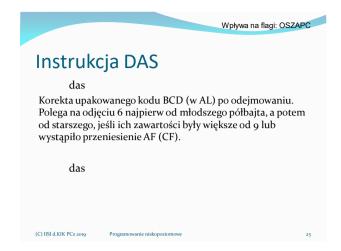


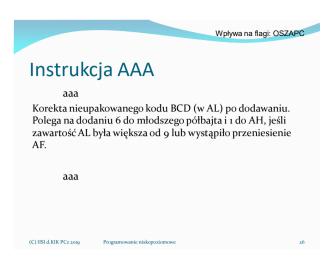
Instrukcja CMP cmp źródło1, źródło2 Porównuje zawartość źródła1 i źródła2, wynik nie jest zapamiętywany, tylko są ustawiane flagi. źródło1-źródło2 cmp ax, zmienna cmp edx, [ebx+esi*4] cmp rcx, 123 Uwaga: Jeśli źródło2 jest adresowane w trybie prostym może mieć do 32 bitów.

Przykład – oblicz sumę kwadratów liczb w tablicy ;wartość początkowa sumy mov eax, o mov esi, eax ;indeks ebx, offset wektor ;tablica liczb całkowitych mov ;licznik mov ecx, 123 edx, [ebx+esi*4] petla: mov imul edx, edx ;kwadrat liczby add eax, edx ;suma inc esi dec ecx jnz petla ;wynik w eax









Instrukcja AAS

aas

Korekta nieupakowanego kodu BCD (w AL) po odejmowaniu. Polega na odjęciu 6 od młodszego półbajta i 1 od AH, jeśli zawartość AL. była większa od 9 lub wystąpiło przeniesienie AF.

aas

Instrukcja AAM

aam

Korekta nieupakowanego kodu BCD (w AX) po mnożeniu.
Polega na jednoczesnym wykonaniu:
AH=AL div 10
AL:=AL Mod 10.

aam

Instrukcja AAD

aad
Korekta nieupakowanego kodu BCD (w AX) przed dzieleniem. Polega na jednoczesnym wykonaniu:
AL:=AH*10+AL.
AH=0

aad

Przykład – dodawanie liczb BCD al, o mov add al, al ;pobierz cyfrę źródła petla: mov al, [esi] al, ds:[edi] ;dodaj cyfrę celu z przeniesieniem adc ·korekta aaa ds:[edi], al mov ;zapamiętaj cyfrę inc esi ;następna cyfra inc edi dec ecx petla ;CF nie zmieniło się od AAA!!! inz (C) IISI d.KIK PCz 201