Wrocław, 09.04.2018

Adam Karczewski, 235039
WT-N-7

prowadzący: Aleksandra Postawka

Laboratorium Architektury Komputerów

(2) Utrwalenie umiejętności tworzenia prostych konstrukcji programowych

2.1 Treść ćwiczenia

Zakres i program ćwiczenia:
Utrwalenie umiejętność tworzenia prostych konstrukcji programowych oraz wczytania i zapisu do pliku tekstowego.

Zrealizowanie zadania:
Wczytanie z pliku dwóch dużych liczb czwórkowych. Zapis do pamięci komputera w konwencji litle endian, dodanie ich z użyciem flagi przeniesienia i rejestrów 8 bajtowych. Wynik konwertowany jest na system szesnastkowy i zapisywany do pliku wynikowego.

2.2 Budowa kodu źródłowego

Pierwszym etapem jest otwarcie pliku wejściowego.

```
movq $SYSOPEN, %rax
movq $file_in1, %rdi
movq $FREAD, %rsi
movq $0, %rdx
syscall
movq %rax, %r8
```

W tym celu wywołujemy funkcję SYSOPEN, kolejnymi parametrami są nazwa pliku i tryb otwarcia. Funkcja zwraca uchwyt do pliku, który zapisujemy do rejestru r8. Odczyt z pliku odbywa się analogicznie jak z konsoli, jako strumień wejścia podajemy uchwyt do pliku- r8.

```
movq $SYSREAD, %rax
movq %r8, %rdi
movq $in1, %rsi
movq $1024, %rdx
syscall
```

Dane zostają zapisane w buforze in1 o wielkości 1024 bajtów. Po zakończeniu pracy z plikiem należy go zamknąć, za pomocą funkcji SYSCLOSE.

```
movq $SYSCLOSE, %rax
movq %r8, %rdi
movq $0, %rsi
movq $0, %rdx
syscall
```

Kolejnym krokiem jest odkodowanie liczby znajdującej się w buforze i zapis w konwencji litle endian. Odczyt zaczynamy od końca.

```
movb in1(, %r9,1), %al
sub $'0',%al
cmp $0, %r9
jle zapiszA
```

Wczytujemy znak do rejestru al, dekodujemy ascii, jeżeli nie doszliśmy do końca wczytujemy kolejny znak.

```
dec %r9
movb in1(,%r9,1), %bl
sub $'0', %bl
shl $2, %bl
add %bl, %al
cmp $0, %r9
jle zapiszA
```

Kolejny znak pobieramy do rejestru bl, odkodowujemy, przesuwamy w lewo o 2 i dodajemy do rejestru al. Operację powtarzamy w sumie 4 razy, gdyż na jednym bajcie mieszczą się 4 cyfry w systemie czwórkowym. Za każdym razem zwiększamy przesunięcie w lewo: 0,2,4,6 oraz sprawdzamy, czy nie doszliśmy do końca bufora. Wynik zapisujemy w buforze wartości (od początku).

```
movb %al, value1(,%r10,1)
cmp $0, %r9
jg zapis1
```

Opisane powyżej operacje powtarzamy także dla liczby w drugim pliku. Kolejną operacją jest dodanie liczb. W tym celu pobieramy liczby z bufora do rejestru 64 bitowego i dodajemy z użyciem flagi przeniesienia, w celu ochrony flagi CY przed niepożądaną modyfikacją (np. podczas operacji porównania) odkładamy ją na stos pomiędzy operacjami (przed rozpoczęciem dodawania flaga została wyzerowana).

```
clc #czyszczę flagę przeniesienia
pushfq #umieszczam rejestr z flagami na stosie
movq $0, %r8 #licznik pętli

dodaj:
movq value1(,%r8,8), %rax
movq value2(,%r8,8), %rbx
popfq
adc %rbx,%rax
jnc s_koniec
pushfq
n_koniec:
movq %rax, sum(,%r8,8)  #zapis wyniku
add $8, %r8 #skok co 8
cmp $256,%r8
jl dodaj
```

Ponieważ pobieram co 8 bajtów, licznik pętli zmienia się w każdym obrocie o 8. Pętla kończy się, jeżeli dotarliśmy do końca bufora lub flaga przeniesienia jest równa 0 (skok jnc) oraz wynik dodawania jest 0.

```
s_koniec:
pushfq
cmp $0, %rax
jne n_koniec
```

Kolejnym etapem jest zakodowanie sumy w systemie szesnastkowym. W tym celu z bufora kopiowany jest bajt, po przez operacje iloczynu logicznego i przesunięcia bitowego w prawo rozdzielamy bajt na dwie części (jeden bajt koduje dwie cyfry szesnastkowe).

```
movb sum(,%r9,1), %al
movb %al, %bl
movb %al, %cl
shr $4, %cl  #przesuwam w prawo o 4
and $0b1111, %bl   #wyłuskuje 4 niższe bity
and $0b1111, %cl   #wyłuskuje 4 wyższe bity
add $'0', %bl
add $'0', %cl
```

Po zakodowaniu wyniku w ascii sprawdzane jest, czy cyfra szesnastkowa nie jest większa od 9. Jeśli tak kodują ją przy pomocy znaku A,B..F. Wynik zapisywany jest do bufora wyjściowego. Dodany zostaje znak końca linii.

Ostatnim etapem jest zapis do pliku. Tak jak poprzedni otwieramy plik nadając mu odpowiednie uprawnienia.

```
movq $SYSOPEN, %rax
movq $file_out, %rdi
movq $FWRITE, %rsi
movq $0644, %rdx
```

Zapis do pliku jest analogiczny do drukowania tekstu na konsoli. Jako strumień zostaje podany uchwyt do pliku. Na końcu należy uruchomić funkcję SYSCLOSE.

2.2 Wnioski

W trakcie trwania laboratorium udało się wykonać polecenie. Po przez nieuwagę operacja dodawania została zrealizowana na rejestrach 8b, w domu dokonano modyfikacji dodawania na rejestrach 8B.