Kamil Skarżyński

Programowanie niskopoziomowe
Asemblacja i konsolidacja
Laboratorium 01

1. Wprowadzenie

Podczas laboratoriów zapoznamy się z:
1. Podstawową strukturą programu,
2. Na jakich typach danych operuje procesor,
3. Czym różni się stała od zmiennej,
4. Czym różni się dyrektywa od instrukcji,
5. W jaki sposób asembler przekształca kod programu w kod maszynowy (proces asemblacji),
6. W jaki sposób program wykorzystuje funkcje WinApi32,
7. Na czym polega proces konsolidacji

2. Pytania

1. Po co wskazywany jest zestaw instrukcji w kodzie programu? (.386)
2. Po co specyfikowany jest model adresowania?
3. Dlaczego specyfikuje się konwencje wywołan?
4. Co to jest zmienna?
5. Jakimi rodzajami zmiennych dysponujemy podczas programowania w środowisku masm32?
6. Czym różni się stała od zmiennej?
7. Dlaczego importujemy biblioteki w kodzie programu?
8. Czym jest segment danych?
9. Czym jest segment kodu?
10. Co to jest dyrektywa i czym różni się od instrukcji?

3. Przydatne linki

1. Hex-Editor - plugin do np++ w celu przeglądania plików w postaci hex
2. Paramtry dla asemblera
3. Kody instrukcji
4. Konwencje wywołań

4. Informacje uzupełniające

4.1. Zestaw instrukcji

Zestaw instukcji jest niezbędny dla kompilatora. W celu przetłumaczenia instrukcji np: Add na kod maszynowy odpowiadający konkretnemu modelowi procesora. Dla przykładu instrukcja:

add eax, 5

Zostanie przetłumaczona na kod maszynowy w formie szesnastkowej:

83 C0 05

Z czego 83 C0 to kod instrukcji Add w ktorej pierwszym parametrem jest rejestr EAX, a 05 to wartość dodawana do tego rejestru. Więcej przykładowych kodów instrukcji możemy znaleźć w sekcji Przydatne linki punkt 3.

4.2. Model adresowania

Aktualnei wykorzystywany model adresowania to FLAT, pozwala to na traktowanie całej pamięci jako ciągła przestrzeń bez uwzględniania segmentacji. Pozwala to na adresacje przestrzeni za pomocą 32 bitowego rejestru czyli 2^{32} bajtów == 4GB.

4.3. Konwencja wywołań

Konwencja wywołań pozwala na specyfikację w jaki sposób program który piszemy będzie osbługiwał przekazywanie parametrów do procedur.

4.4. Zmienne

Przykładowa deklaracja zmiennej w środowisku masm32:
Nazwa typ danych wartość.

zmienna1 **db** 10h

Deklaracja zmiennych jest niezbędna w celu alokacji pamięci. Kompilator musi zanotować niezbędną przestrzeń, by system operacyjny wiedział czy jest w stanie wygospodarować odpowiednią ilość poamięciu by uruchomić program.

4.5. Rodzaje zmiennych

Podstawowe rozmiary zmiennych to:
DB == Byte == 8 bitów
DW == Word == 16 bitów
DD == DWord == 32 bity

Przykłady deklaracji:

zmiennaBajtowa **db** 10h
zmiennaDwuBajtowa **dw** 10h
zmiennaCzteroBajtowa **dd** 10h

4.6. Stałe i zmienne

Stałe różnią się od zmiennych tym, że nie jest dla nich alokowana dodatkowa pamięć podczas uruchomienia programu ponieważ są wstawiane jako parametry instrukcji podczas kompilacji programu.

4.7. Import bibliotek

Bilioteki możemy zarówno importować w kodzie, jak i jako parametr podczas konsolidacji. Są one niezbędne w celu pobrania odpowiednich adresów

4.8. Segment danych

Segment danych to przestrzeń w której kompilator oczekuje deklaracji wszystkich zmiennych.

4.9. Segment kodu

Segment kodu to przestrzeń w której kompilator oczekuje instrukcji.

4.10. Dyrektywa a instrukcja

Instrukcja to polecenie, które wykonuje procesor. Dyrektywa to informacja dla kompilatora. Przykładem dyrektywy jest np deklaracja stałej.