# *LABORATORIUM ARCHITEKTURY KOMPUTERÓW*

## TEMAT:

Turbo Debugger – deasemblacja programów napisanych, za pomocą języka C



**ĆWICZENIA:**

Kolejność wykonywania zadań jest dowolna. Ocena uzależniona jest od ilości, jak i stopnia trudności wykonanych punktów.

**1.** Napisz w środowisku Borland C/C++3.1, prosty kod zliczający dowolną wartość w pętli. Skompiluj go w sposób umożliwiający wprowadzenie do Turbo Debuggera.

**2.** Jeżeli program uruchomił się oknem Module, to kod w C już znasz więc uruchom View|CPU. Analiza tego okna jest głównym celem laboratorium. Rozpoczynając prace w tym oknie, podświetlona jest linia _main(), poleceniem Run|Run (F9) i otwarciem Window|User screen, możesz sprawdzić działanie programu. Przeanalizuj:

a) Jak reprezentowana jest pętla języka C za pomocą składni asemblera, sprawdź jak interpretowane są inne pętle.
b) Zwróć uwagę na rozmiary otrzymywanego kodu.

**3.** Analizując programy lab2_0.exe, obsługujący przerwanie programowe za pomocą funkcji BIOS, oraz program lab3_0.exe, dotyczący przerwania sprzętowego, realizowanego bezpośrednio za pomocą portów, porównaj zachowanie flag procesora. Opisz swoje obserwacje w sprawozdaniu.

**4.** Wprowadź do Turbo Debuggera, w celu śledzenia, program lab2_0.exe.

a) Porównaj kod po deasemblacji, ze składnią języka C.
b) Sprawdź informacje zawarte w oknie File|Get info...
c) Zapoznaj się z zawartością okna View|Variables.
d) Wprowadź do okna Watches wszystkie zmienne twojego programu, poleceniem Data|Add watch...
e) Prześledź wartości tych zmiennych wykonując polecenie Run|Step Over (F8), oraz wartości wysyłane na stos (SS).
f) Prześledź działanie funkcji int86(). Zaobserwuj, kiedy funkcja przepisuje wartości unii regs do rejestrów procesora i odwrotnie.

g) Spróbuj prześledzić działanie używanych w programie funkcji programowego przerwania 10H.
h) Opisz w sprawozdaniu jak realizowane jest w asemblerze wywoływanie przerwań.
i) Wykonuj pojedynczo instrukcje asemblera, poleceniem Run|Trace into (F7). Obserwuj zmienne w oknie Watches. Prześledź dla różnych instrukcji asemblera wartość zmiennej w oknie rejestrów, przed i po uruchomieniu (F7), oraz wpływ instrukcji na flagi.
j) Sprawdź działanie polecenia Animate...

**5.** Wprowadź do Turbo Debuggera, w celu śledzenia, program lab3_0.exe (po odpowiednim skompilowaniu).
a) Wykonaj podpunkty od a) do e) jak w poprzednim ćwiczeniu.
b) Opisz w sprawozdaniu jak realizowana jest w asemblerze komunikacja z portami.

**6.** Napisz w środowisku Borland C/C++3.1, proste kody do realizacji poniższych podpunktów. Skompiluj je w sposób umożliwiający wprowadzenie do Turbo Debuggera. Porównaj, uzyskane po deasemblacji kody. W sprawozdaniu opisz różnice wygenerowanych kodów, zestawionych jak w podpunktach.
a) void fun() – void interrupt fun(),
b) int86(0x21,&regs,&regs) – intdos(&regs,&regs),
c) int86 – int86x,
d) intdos – intdosx,
e) Jakimi przerwaniami asembler zastępuje polecenia: getvect i setvect.

**7.** Zapoznaj się z innymi opcjami Turbo Debuggera, nie używanymi do tej pory.

**8.** Przygotować sprawozdanie z laboratorium zawierające:
a) temat ćwiczenia,
b) krótki opis zagadnienia,
c) kody asemblerowe programów lab2_0 i lab3_0, opatrzone szerokimi i wyczerpującymi komentarzami (kod C na ASM: BCC –mt plik.c).

***Zagadnienia do przygotowania na następne laboratoria:***
**1.** Łącze szeregowe (rodzaje transmisji). Interfejs RS-232C.
**2.** Ramka danych.
**3.** Układ UART (i jego rejestry). Programowanie łącza na poziomie portów.
**4.** Informacje dostępne o łączu szeregowym, w obszarze danych BIOS (BDA).
**5.** Przerwania BIOS i DOS obsługujące COM.