Katedra Inżynierii Komputerowej
Politechnika Częstochowska

Laboratorium Programowania niskopoziomowego

Wprowadzenie do asemblera

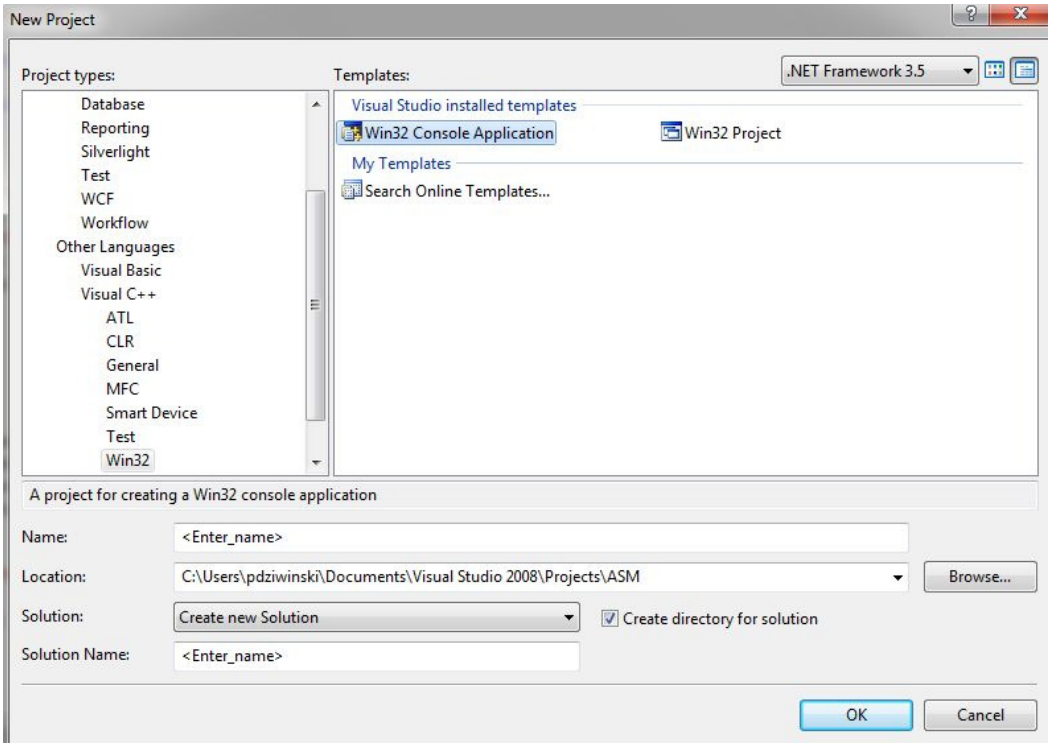
dr inż. Dziwiński Piotr

09-10-2011

1 Budowa prostego programu Win32 C++, wstawka asemblerowa.

Celem ćwiczeń laboratoryjnych z programowania niskopoziomowego jest zapoznanie studenta z podstawowymi instrukcjami asemblera, wpływem ich działania na stan poszczególnych rejestrów procesora. Zrozumienie sposobu wykonania instrukcji asemblera jest niezbędne do realizacji kolejnych ćwiczeń laboratoryjnych.

- Proszę utworzyć w Visual Studio 2010 nowy projekt Visual C++ Win32 Application,



- Należy wprowadzić fragment kodu w C++ stanowiący otoczenie dla wstawki aseblerowej

```
// ASM1.cpp : Defines the entry point for the console application.
#include "stdafx.h"
#include <iostream>

using namespace std;

int _tmain(int argc, _TCHAR* argv[])
{
        system("PAUSE");
        return 0;
}
```

- Wprowadzamy wstawkę asemblerową

```
int _tmain(int argc, _TCHAR* argv[])
{
        //Ewentualnie pobranie danych wejściowych
        //plik, konsola
        __asm {
                //wprowadzany kod asemblera
        }
        //Ewentualnie wyprowadzenie wyników
        //plik, konsola
        system("PAUSE");
        return 0;
}
```

Na zajęciach laboratoryjnych należy wprowadzić podstawowe instrukcje asemblera poznane na wykładach, następnie podczas procesu debugowania, należy przyjrzeć się wszystkim zmianom, jakie następują w poszczególnych rejestrach procesora. Przykładowy zestaw instrukcji do wprowadzenia:

```
int _tmain(int argc, _TCHAR* argv[])
{
        int a=10;
        int b=20;
        __asm {
                mov eax,0xaaff;
                mov ebx,eax;
                inc ea; //??
                inc eax;
                dec ecx;
                dec eax;
                add a,eax; //??
                add a,b;  //??

                mov eax,a;
                mov ebx,b;
                add eax,b;
                cmp eax, ebx;
                sub eax,b;
                mul ecx;


        }
        system("PAUSE");
        return 0;
}
```

Uwaga: niektóre instrukcje są wprowadzone celowo z błędem, należy je wtedy ustawić jako komentarz lub spróbować poprawić. W przypadku błędnych instrukcji proszę zwrócić uwagę na komunikaty o błędach.

2 Debugowanie programu w Visual Studio 2008/2010

W celu debugowania programu w Visual Studio 2008 należy zaznaczyć punkt przerwania pracy programu Rys. 1 (należy kliknąć na szarym marginesie okna edycyjnego kodu programu), następnie kompilujemy program (F6), uruchamiamy program (F5).

Pozostałe kona pomocne podczas procesu debugowania:

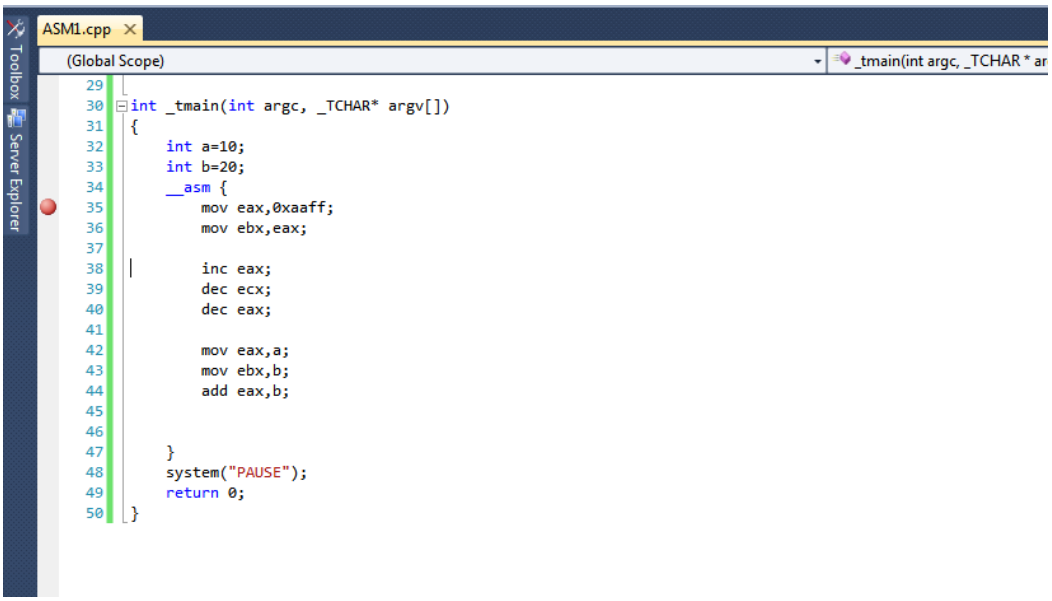
- okno podglądu zawartości rejestrów procesora - menu **Debug -> Windows -> Registers (Ctrl + D, R)**,
- inne rejestry procesora - menu kontekstowe dla okna rejestrów procesora -> właściwy zestaw rejestrów (zależnie od fizycznego procesora będą dostępne różne zestawy rejestrów),
- okno podglądu skompilowanego kodu asemblera - menu **Debug -> Windows -> Disassembly (Ctrl + Alt, D)**,

Praca krokowa w Visual Studio

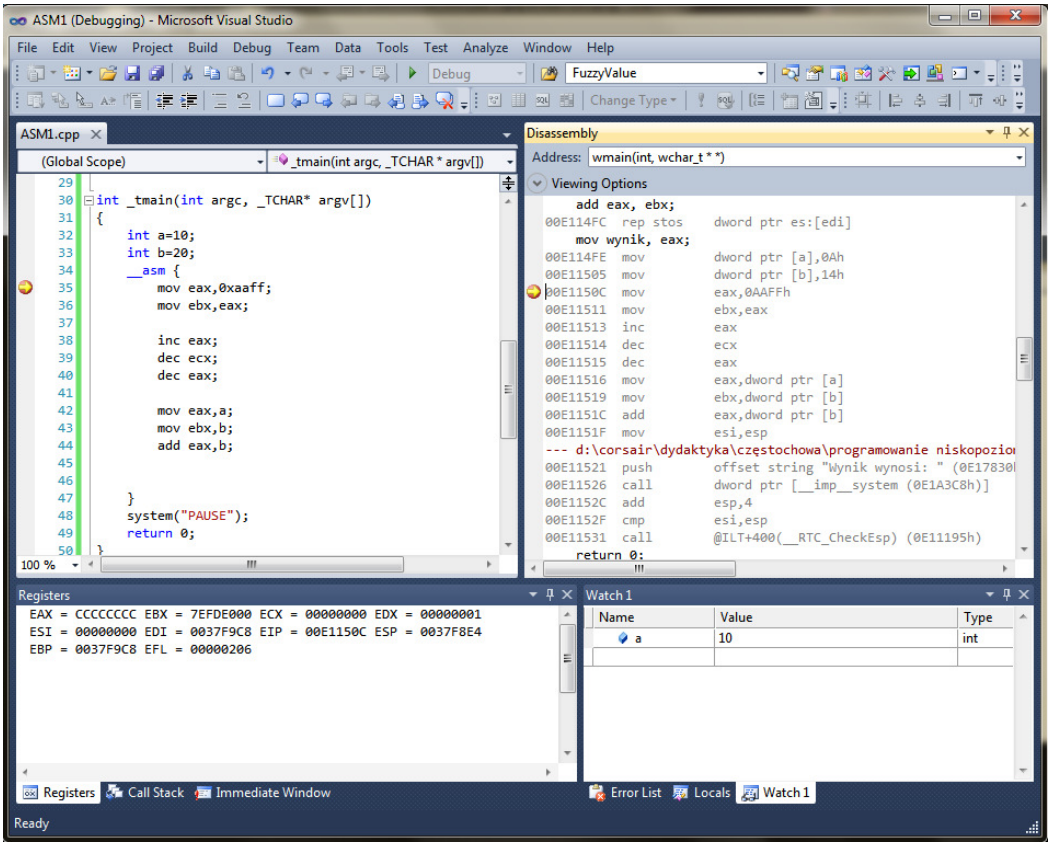
- Następna instrukcja - F10
- Następna instrukcja (jeżeli funkcja to wejdź do środka) - F11
- Następna instrukcja (jeżeli wewnątrz funkcji to wyjdź na zewnątrz) - Shift F11
- Kontynuuj - F5

Okna ustawiamy w odpowiednich miejscach tak, aby maksymalnie ułatwić proces analizy wykonania programu Rys. 2.

Proszę się przyjrzeć kodzie programu w asemblerze. Kompilator dokonuje czasami pewnych zmian w zaimplementowanym kodzie asemblera. Analizując ten kod, można często uprościć bądź przyspieszyć program. W przypadku wystąpienia błędów działania programu, można określić jego przyczynę.



Rysunek 1: Włączanie punktu przerwania pracy programu



Rysunek 2: Debugowanie programu