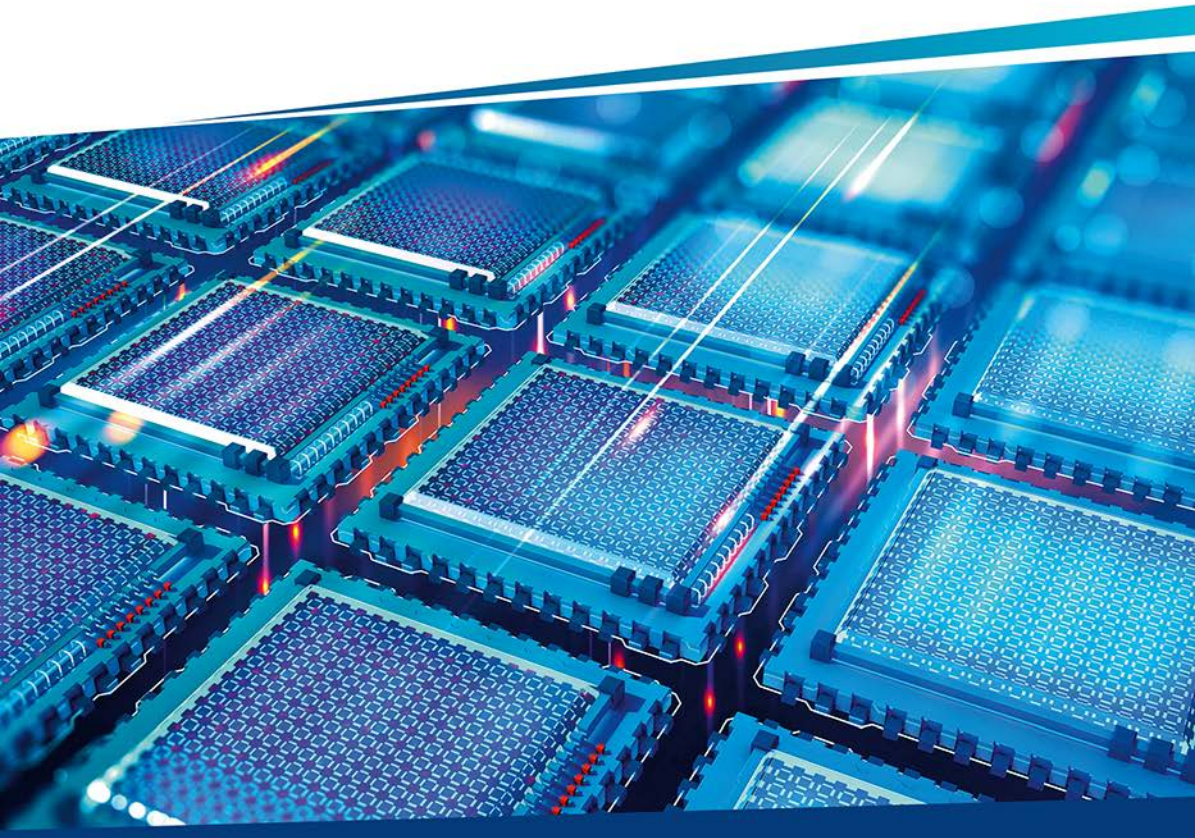
Dawid Farbaniec

Asembler

Programowanie



Helion

Redaktor prowadzący: Małgorzata Kulik

Projekt okładki: Studio Gravite / Olsztyn
Obarek, Pokoński, Pazdrijowski, Zaprucki

Grafika na okładce została wykorzystana za zgodą Shutterstock.com

Wydawnictwo HELION
ul. Kościuszki 1c, 44-100 GLIWICE
tel. 32 231 22 19, 32 230 98 63
e-mail: *helion@helion.pl*
WWW: *http://helion.pl* (księgarnia internetowa, katalog książek)

Drogi Czytelniku!
Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres
http://helion.pl/user/opinie/asempr
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

ISBN: 978-83-283-5495-1

Printed in Poland.

Spis treści

Rozdział 12.

Łączenie Asemblera z językiem C++

12.1. Funkcje Asemblera w projekcie Visual C++

Po utworzeniu projektu w środowisku Visual Studio dla języka C++ (Visual C++) mamy możliwość pisania funkcji w języku Asembler. Poszczególne kroki tworzenia takiego projektu przedstawiono w podrozdziale 3.1. „Tworzenie projektu MASM w Visual Studio".

Łączenie programu w Visual C++ z procedurami napisanymi w Asemblerze MASM polega na dodaniu do kodu w C++ zapisu:

```
extern "C" void my_assembler_procedure1();
```

Oczywiście nazwa procedury (`my_assembler_procedure1`) oraz zwracany typ (`void`) powinny być zastąpione zgodnie ze składnią tworzonej procedury.

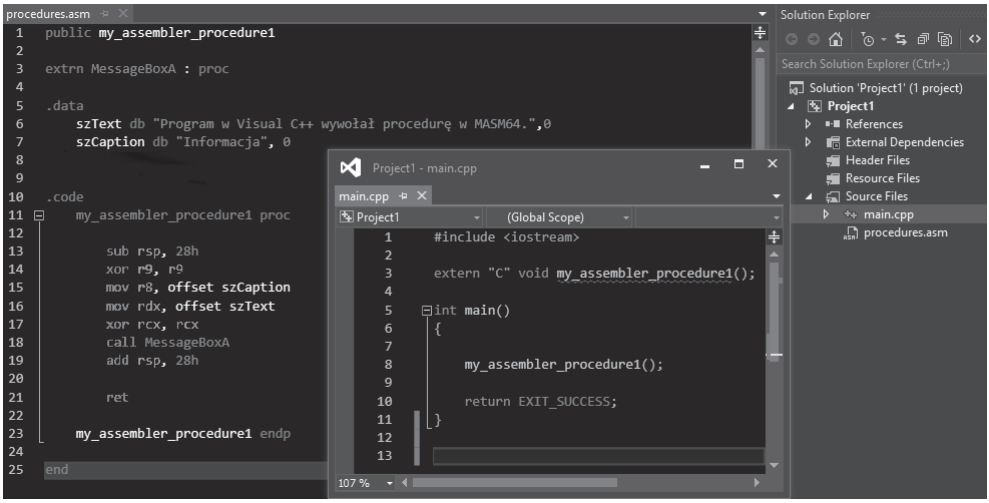
Zapis `extern "C"` powoduje tutaj odwołanie się z pliku źródłowego w C++ (np. *main.cpp*) do procedury zapisanej w języku Asembler (plik o rozszerzeniu *.asm*). Zarówno plik *.cpp*, jak i *.asm* należy standardowo dodać do plików projektu w gałęzi *Source Files* w oknie *Solution Explorer*.

W przypadku niewykrycia funkcji asemblerowej przez konsolidator (ang. *linker*) należy dodać na początku kodu źródłowego w języku Asembler (*.asm*) zapis:

```
public my_assembler_procedure1
```

Zmieni to widoczność określonej procedury lub grupy procedur.

Przykładowy projekt przedstawiono na rysunku 12.1.



RYSUNEK 12.1. *Wywołanie procedury w języku Asembler z programu w Visual C++*

12.2. Funkcje wewnętrzne (ang. intrinsics) w Visual C++

Kompilator Visual C++ ze środowiska Visual Studio nie wspiera wstawek w języku Asembler w postaci *inline assembly* dla architektury 64-bitowej (x86-64). Zamiast tego udostępnione zostały funkcje wewnętrzne kompilatora, nazywane *intrinsics*. Funkcje te są odwzorowywane na określone instrukcje języka Asembler, które nie są standardowo dostępne z poziomu języka C++.

Na przykład: nie możemy użyć rozkazu asemblerowego `BSF` (ang. *bit scan forward*) w kodzie Visual C++, ale możliwe jest wykonanie tego typu operacji za pomocą funkcji wewnętrznej o nazwie `_BitScanForward()`.

Przykład użycia funkcji wewnętrznej (ang. intrinsic)

W poniższym przykładzie wykorzystana została funkcja wewnętrzna `_mul128`, która wykonuje mnożenie dwóch liczb typu `long long` (64 bity) i zwraca 128-bitowy wynik w dwóch częściach po 64 bity (dwa razy `long long`).

Składnia tej funkcji prezentuje się następująco:

```
long long _mul128(
   long long Multiplier,
   long long Multiplicand,
   long long *HighProduct
);
```

Zatem funkcja ta przyjmuje jako argumenty: *mnożnik*, *mnożną* oraz *wskaźnik na starszą część iloczynu*, a zwraca *młodszą część iloczynu*. Przykładowe użycie tej funkcji przedstawiono na listingu 12.1.

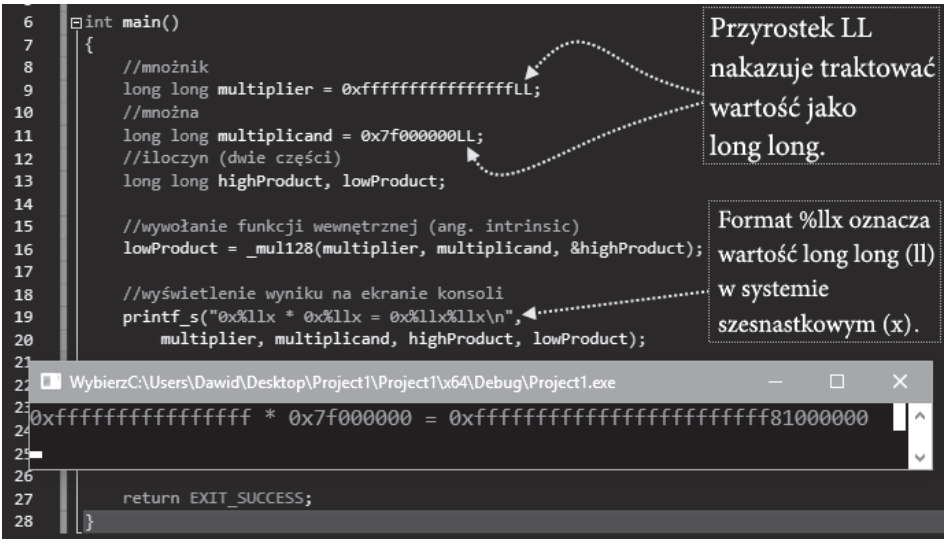
LISTING 12.1. *Mnożenie dwóch liczb typu long long za pomocą funkcji wewnętrznej _mul128 (Visual C++)*

```
#include <iostream>
#include <intrin.h>

#pragma intrinsic(_mul128)

int main()
{
    //mnożnik
    long long multiplier = 0xffffffffffffffffLL;
    //mnożna
    long long multiplicand = 0x7f000000LL;
    //iloczyn (dwie części)
    long long highProduct, lowProduct;

    //wywołanie funkcji wewnętrznej (ang. intrinsic)
    lowProduct = _mul128(multiplier, multiplicand, &highProduct);

    //wyświetlenie wyniku na ekranie konsoli
    printf_s("0x%llx * 0x%llx = 0x%llx%llx\n",
        multiplier, multiplicand, highProduct, lowProduct);

    //zatrzymanie konsoli, ale tylko w trybie DEBUG
#if _DEBUG
    std::cin.get();
#endif

    return EXIT_SUCCESS;
}
```

Działanie programu z listingu 12.1 przedstawiono na rysunku 12.2.

Dokumentacja opisująca funkcje wewnętrzne kompilatora Visual C++ dostępna jest pod adresem:

https://docs.microsoft.com/pl-pl/cpp/intrinsics/compiler-intrinsics.




```
int main()
{
    //mnożnik
    long long multiplier = 0xffffffffffffffffLL;
    //mnożna
    long long multiplicand = 0x7f000000LL;
    //iloczyn (dwie części)
    long long highProduct, lowProduct;

    //wywołanie funkcji wewnętrznej (ang. intrinsic)
    lowProduct = _mul128(multiplier, multiplicand, &highProduct);

    //wyświetlenie wyniku na ekranie konsoli
    printf_s("0x%llx * 0x%llx = 0x%llx%llx\n",
        multiplier, multiplicand, highProduct, lowProduct);

    return EXIT_SUCCESS;
}
```

RYSUNEK 12.2. *Działanie przykładowego programu dla funkcji wewnętrznej _mul128 (Visual C++)*

Skorowidz

J

K

L

M

O

P

R

S

T

V

W

Z


PROGRAM
PARTNERSKI
GRUPY HELION
1. ZAREJESTRUJ SIĘ
2. PREZENTUJ KSIĄŻKI
3. ZBIERAJ PROWIZJĘ
Zmień swoją stronę WWW w działający bankomat!
Dowiedz się więcej i dołącz już dzisiaj!
http://program-partnerski.helion.pl
GRUPA Helion

Asembler naprawdę jest dla każdego!

- **Masz zamiar nauczyć się programować niskopoziomowo?**
- **Chcesz poznać język Asembler dla architektury x86-64?**
- **Pragniesz wykorzystać przydatne narzędzia programistyczne?**

Jeśli chociaż na jedno z powyższych pytań odpowiedziałeś twierdząco, jesteś na dobrej drodze! Książka *Asembler. Programowanie* bezboleśnie i szybko wprowadzi Cię w tajniki programowania niskopoziomowego, nauczy instrukcji języka Asembler, przedstawi podstawowe informacje o sposobach zapisu liczb i kodowania znaków, zaprezentuje architekturę x86-64 i zasadę działania kompilatorów, a także pokaże narzędzia programistyczne, które z pewnością wzbogacą Twój warsztat.

Dzięki lekturze dowiesz się, jak za pomocą Asemblera tworzyć aplikacje działające w systemach Windows, jak korzystać w nich z danych tekstowych, plików i usług sieciowych, a także jak skutecznie łączyć je z programami napisanymi w języku C++. Poznasz też podstawy Asemblera MSIL/CIL i WebAssembly, jak również sposoby wykorzystania tego języka w inżynierii odwrotnej kodu oraz narzędzia, które mogą się do tego przydać. Całość uzupełnia wyczerpujący opis instrukcji procesorów x86(-64) wraz z praktycznymi przykładami ich zastosowania.

- **Systemy liczbowe i kodowanie znaków**
- **Podstawy działania kompilatorów**
- **Architektura procesorów x86(-64)**
- **Przegląd narzędzi programistycznych**
- **Podstawy Asemblera x86 i x86-64**
- **Zmienne, stałe, typy i struktury**
- **Instrukcje sterujące i wywoływanie funkcji**
- **Korzystanie z konsoli i okien dialogowych**
- **Zastosowanie danych tekstowych**
- **Użycie plików i funkcji sieciowych**
- **Łączenie Asemblera z kodem C++**
- **Asembler i inżynieria odwrotna kodu**
- **Podstawy Asemblera platformy .NET**
- **Podstawy WebAssembly**
- **Opis instrukcji procesorów x86(-64)**

Rozwiń skrzydła! Naucz się Asemblera!

Helion

helion.pl

HELION SA
ul. Kościuszki 1c
44-100 Gliwice
tel.: 32 230 98 63
helion@helion.pl

ISBN 978-83-283-5495-1

Cena: 44,90 zł