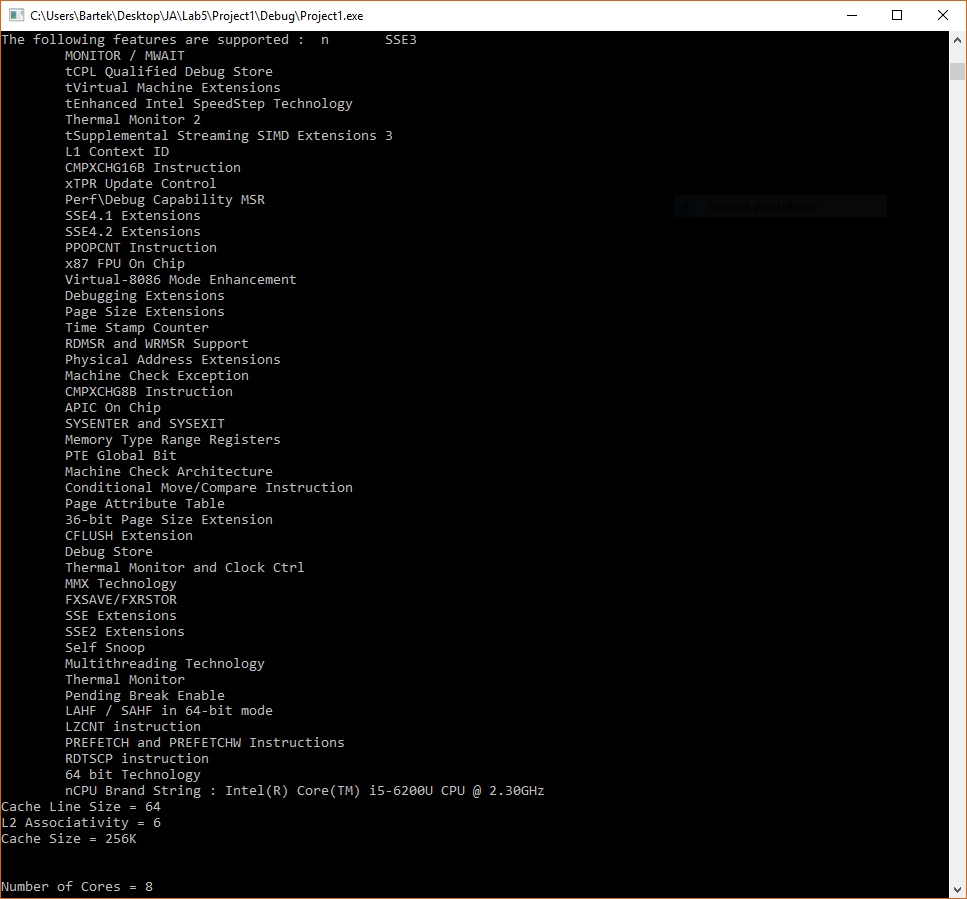
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| logowydzialu | | Instytut Informatyki Politechniki Śląskiej  Zespół Mikroinformatyki i Teorii Automatów Cyfrowych  **Laboratorium JA** | | logoii | |
| **Rok akademicki** | | **Rodzaj studiów\*: SSI/NSI/NSM** | **Numer ćwiczenia:** | **Grupa** | **Sekcja** |
| **2017/2018** | | **SSI** | **5** | **6** | **1** |
| **Data i godzina planowana ćwiczenia:**  dd/mm/rrrr - gg:mm | | **27/03/2018-11:45** | **Prowadzący**:  OA/AO | **OA** | |
| **Data i godzina wykonania ćwiczenia:**  dd/mm/rrrr - gg:mm | | **27/03/2018-11:45** |
| ***Sprawozdanie*** | | | | | |
| **Temat ćwiczenia:**  Zapoznanie się z instrukcjami MMX i SSE w j. asemblera. | | | | | |
| **Skład sekcji:** | 1.Bartłomiej Krasoń | | | | |

# Cel

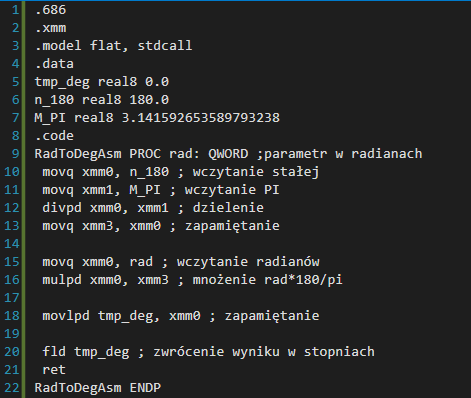
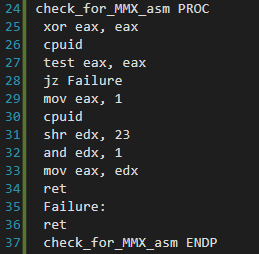
Celem ćwiczenia jest poznanie działania rozkazów MMX i SEE wykorzystywanych do wykonywania precyzyjnych obliczeń zmiennoprzecinkowych procesorów x86 firmy Intel dla środowiska Windows. *~Źródło – Instrukcja ćwiczenia LAB5*

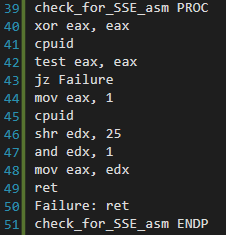
# Rozwiązanie

Korzystając z kodu zamieszczonego w instrukcji sprawdziłem czy mój procesor obsługuje instrukcje MMX, SSE i SSE2:

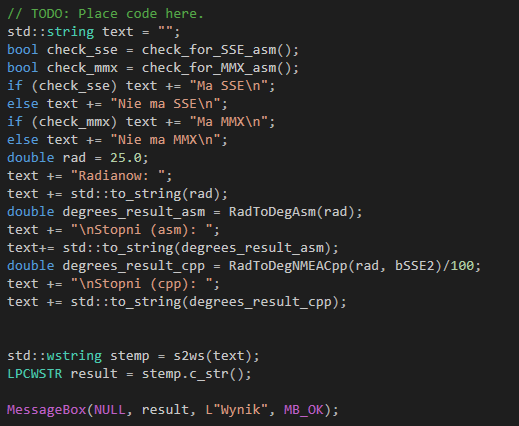


Następnie w asemblerze zaimplementowałem następujące procedury:

1. RedToDegAsm – która konwertuje radiany na stopnie
2. check\_for\_mmx\_asm – która sprawdza czy procesor obsługuje rozszerzenie MMX
3. check\_for\_sse\_asm – która sprawdza czy procesor obsługuje rozszerzenie SSE

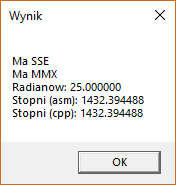


# Następnie przetestowałem działanie powyższych procedur w projekcie w języku C++ tak jak to robiliśmy na wcześniejszych zajęciach:



W celu porównania wyniku funkcji obliczającej stopnie w asemblerze, skorzystałem z funkcji RadToDegNMEACpp zamieszczonej w instrukcji , która oblicza stopnie na podstawie radianów jednakże w formacie NMEA, czyli aby uzyskać odpowiedni wynik, należy podzielić go przez 100.

Wynik pracy programu:



# Wnioski

Wykonując to ćwiczenie poznałem nowe zestawy instrukcji dla procesora, takie jak MMX, SSE, SSE2. Znajomość takich zestawów okazuje się bardzo przydatna, gdyż dzięki nim, pisząc nasze programy możemy zwiększyć dokładność wykonywanych obliczeń, co wpływa pozytywnie na poprawność i funkcjonalność naszych aplikacji. Jednakże trzeba też uważać na kwestię tego, że nie wszystkie procesory mogą obsługiwać te właśnie zestawy instrukcji. Dlatego dowiedziałem się również, jak z poziomu programu asemblerowego czy to napisanego w języku C++ można sprawdzić, jakimi zestawami instrukcji włada nasz procesor.