EFEKT ECHA

Paweł Zawierucha

Założenia projektu:

1. Główny program wczytujący plik wejściowy napisany w C/C++

2. DLL w C/C++ i Assemblerze które dodają efekt echa do pliku .wav

3. Program ma możliwość wyboru której DLL będziemy chcieli użyć

4. Program ma możliwość wyboru ilości wątków, domyślnie będzie ilość optymalna dla urządzenia

Wykorzystane instrukcje wektorowe:

- VMOVDQU - Vector Move Unaligned Packed Integer Values
- VPSRAW - Vector Shift Right Arithmetic Word Integers
- VPADDW - Vector Add Packed Word Integers

Implementacja

```
;initial for procedure - required
sub  rsp, 8
push rbx
push rbp

;move arguments from registers to variables
mov Source, rcx
mov dataSize, rdx
mov destination, r8
mov myBegin, r9
mov r10, myEnd
mov r12, delayStep
mov r14, delayStep
sub r14, iterationLength


mov r13,rcx ;move input addres
add r13,rdx ;add input size (to make end of input)
mov iterationLength, r13

mov r15,r8 ;bufor do zapisu do przodu
add r15,r12

add r10,rcx ; zeby myend to byl koniec dla rcx
mov myEnd, r10

add rcx,r9
add r8,r9
add r15,r9
```

```
MainLoop:
    vmovdqu ymm0,ymmword ptr [rcx] ;wczytaj dane z inputa
    vmovdqu ymm1,ymmword ptr [r8] ;wczytaj dane z outputa

    VPSRAW YMM1, YMM1, 1 ;przesuniecie bitowe arytmetyczne w prawo o 1 bit (sciszenie)

    vpaddw ymm2,ymm1,ymm0 ;ymm2 - wynik to outputu
    vmovdqu ymmword ptr[r8],ymm2 ; zapis do outputu
    vmovdqu ymmword ptr[r15],ymm2 ; zapis do outputu do przodu

    add rcx,32 ;przesuniecie o 16 wartosci (rozmiar ymm)
    add r8, 32
    add r15,32

mov Source, rcx ;debug
mov destination, r8
mov Source, rcx

    cmp rcx,r10 ; sprawdz czy input nie jest poza swoja granica
    jge addingNextDelay
    jmp MainLoop
```

Implementacja

```
addingNextDelay:
    add rcx, r14 ;przesuwa wskazniki do nastepnej iteracji o delaystep - length
    add r8, r14
    add r15, r14
    add r10,r12 ; tylko delaystep

mov Source, rcx
mov destination, r8
mov myEnd, r10

    cmp rcx,r13 ;sprawdz czy input nie jest poza tablica
    jge finishProcedure
    jmp MainLoop
```

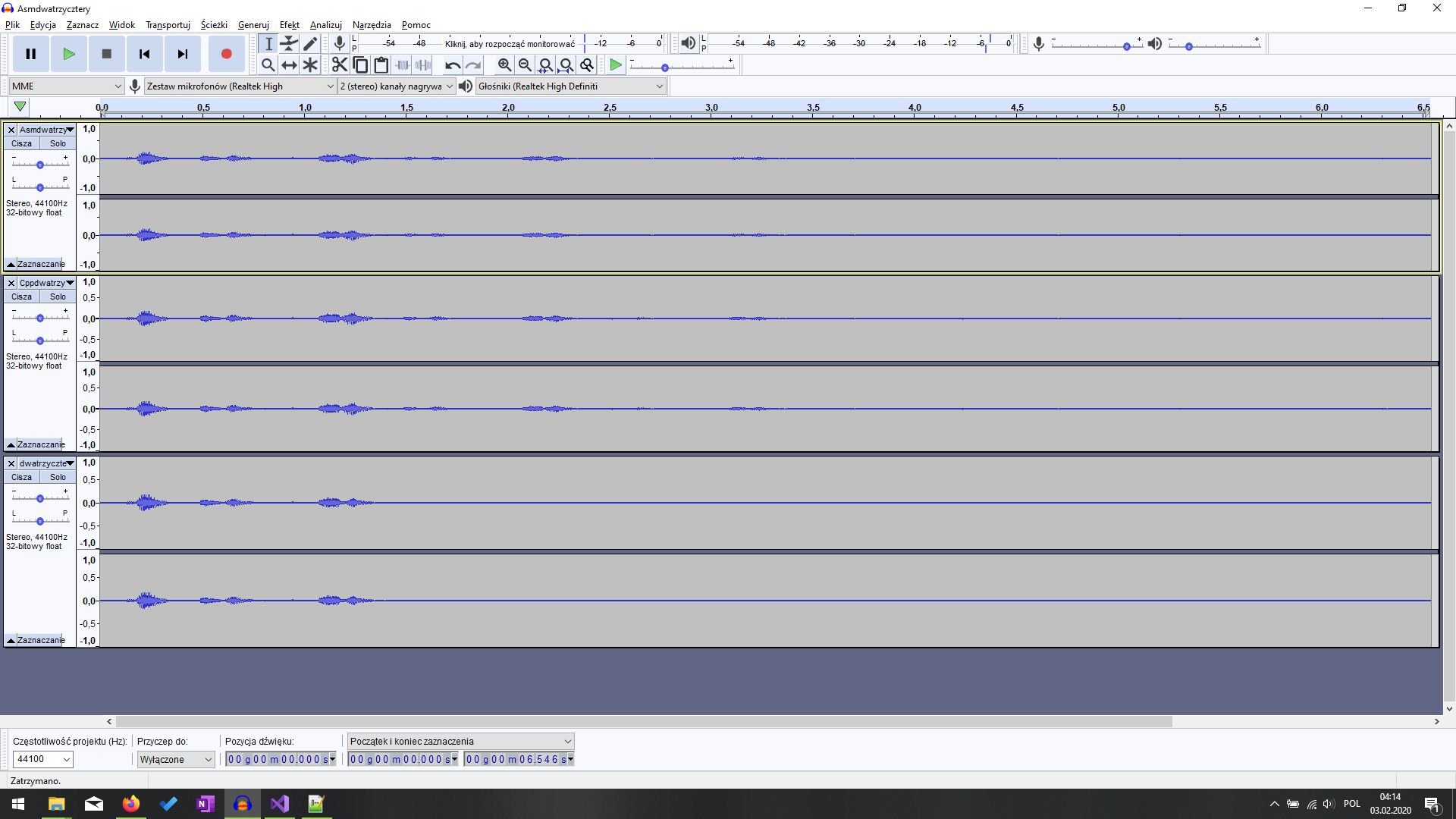
```
finishProcedure:
    pop rbp
    pop rbx

    add rsp,28
    ret
```

Wyniki:

ilość wątków	średni czas ASM	średni czas CPP
1	403,65	5381,2
2	346,11	3 324,56
3	270,60	2 528,21
4	208,81	2 102,69

Wnioski:

Dzięki zastosowaniu instrukcji wektorowych operujących na 16 zmiennych na raz byłem w stanie otrzymać ponad 10-krotny wzrost prędkości przetwarzania plików względem funkcji w c++, która operowała tylko na jednej zmiennej na raz.


Asmdwatrzycztery
Plik Edycja Zaznacz Widok Transportuj Ścieżki Generuj Efekt Analizuj Narzędzia Pomoc
Kliknij, aby rozpocząć monitorować
MME
Zestaw mikrofonów (Realtek High
2 (stereo) kanały nagrywa
Głośniki (Realtek High Definiti
Asmdwatrzy
Cisza Solo
Stereo, 44100Hz
32-bitowy float
Zaznaczanie
Cppdwatrzy
dwatrzyczte
Częstotliwość projektu (Hz):
Przyczep do:
Pozycja dźwięku:
Początek i koniec zaznaczenia
44100
Wyłączone
Zatrzymano.
POL
04:14
03.02.2020