Projekt i implementacja jednostki zmiennoprzecinkowej w standardzie IEE-754

Projekt z organizacji i architekrury komputerów
E07-04b (środa 15:15 - 16:55 TP)

Jakub Korycki
Jan Malek

9 czerwca 2021

Spis treści

Spis tabel

Spis rysunków

1 Wstęp

Standard IEE-754 jest obecnie najbardziej popularnym sposobem komputerowej reprezentacji liczb rzeczywistch. Operacje na tym formacie są skomplikowane, dlatego istnieje potrzeba sprzętowego przyspieszania takich, gdyż implementacje programowe tych operacji mogą okazać się zbyt wolne, aby korzystanie z tak zapisanych liczb było odpowiednio szybkie. Projekt opisuje proces projektowania i implementacji w symulatorze *Logisim* operacji porównania, dodawania i odejmowania oraz mnożenia dwóch liczb w formacie IEE-754. Układ jest czysto kombinacyjny

2 Opis układu

2.1 Ogólne informacje

Opis opóźnienia i obszaru układu jest oparty na uproszczonym modelu AT z następującymi założeniami:

- Bramki logiczne zawsze są traktowane tak samo, niezależnie od liczby wejść
- Sumatory i subtraktory są traktowane jako RCA
- Wszystkie elementy, które posiadają wejścia o różnej długości słowa są traktowane jako elementy 32-bitowe (np. wszystkie sumatory dodają dwie liczby od długości 32 bitów). Dotyczy to sumatorów, subtraktorów oraz komparatorów
- Wartości obaszaru, opóźnienia oraz długość ścieżki krytycznej zostały wyznaczone, poprzez ręczną analizę struktury układów
- Parametry zostały obliczone na podstawie poniższych przybliżeń:

Element układu	Obszar (A)	Opóźnienie (T)	Opóźnienie - ścieżka krytyczna (T_{cc})
AND/OR/NOR/NAND	1	1	1
XOR/XNOR	2	2	2
Sumator/Subtraktor	224	128	64
Obliczanie uzupełnienia	224	128	64
Komparator	226	130	66
Przesuwanie w prawo/lewo	192	192	128
Wyszukiwanie ilości zer wiodących	672	384	192
Multiplekser 2 do 1	3	3	2
Układ mnożący	5728	3520	1496

Tabela 1: Wartości parametrów AT użytych podczas analizy

- Znaczenie prametrów:
  - Obczar - sumaryczna wartość obszaru wszystkich komponentów
  - Opóźnienie - sumaryczna wartość opóźnienia wszystkich komponentów
  - Opóźnienie ścieżka krytyczna - wartość opóźnienia tylko tych komponentów, które znajdują się na ścieżce krytycznej

Testy jednostkowe mają następujące założenia:

- Wartości do testów zostały wygenerowane za pomocą programu napisanego w jezyku *C++*. Kod programu jest dołączony do projektu
- Wyniki w wartościach testowych zostały zaokrąglone do najbliższej reprezentowalnej wartości. Niestety nie udało się znaleźć sposobu, żeby zmienić tryb zaokrąglania na zaokrąglanie przez obcięcie

2.2 Jednostka zmiennoprzecinkowa

2.2.1 Opis funkcjonalności układu

- Układ kombinacyjny
- Porównywanie dwóch liczb 32-bitowych w standardzie IEE-754
- Dodawanie i odejmowanie dwóch liczb 32-bitowych w standardzie IEE-754
- Mnożenie dwóch liczb 32-bitowych w standardzie IEE-754
- Zaokrąglanie przez obcięcie

2.2.2 Opis wejść i wyjść

- Wejście **X** (32 bity) - pierwsza liczba w operacjach
- Wejście **Y** (32 bity) - druga liczba w operacjach
- Wejście **OPCode** (2 bity) - kod operacji jaka ma zostać wykonana
  - 00 - dodawanie $X + Y$
  - 01 - odejmowanie $X - Y$
  - 10 - Mnożenie $X \cdot Y$
  - 11 - brak
  - Porównanie wykonywane jest zawsze po ustawieniu wartości X i Y.
- Wyjście **Out** (32 bity) - wyniki wybranej operacji
- Wyjście **Status** (8 bitów) - informacje na temat wykonanej informacji. Opis bitów (Bit 0 jest na pozycji najmłodszej, a 7 na najstarszej):
  - bit 0 - Znak wyniku (wartość 0 jeśli wynik jest dodatni, 1 jeśli jest ujemny)
  - bit 1 - Normalizacja wyniku (wartość 0 jeśli wynik jest zdenormalizowany, 1 jeśli jest znormalizowany)
  - bit 2 - Flaga 0 (wartość 0 jeśli wynikiem operacji jest liczba 0, bądź -0, w przeciwnym wypadku 1)
  - bit 3 - Flaga INF (wartość 0 jeżeli wynikiem operacji nie jest nieskończoność, w przeciwnym wypadku 1)
  - bit 4 - Flaga NaN (wartość 0 jeżeli wynikiem operacji jest poprawna liczba, w przeciwnym wypadku 1)
  - bit 5 - Niepoprawna operacja (wartość 0 jeśli operacja może zostać wykonana, w przeciwnym wypadku 1)
  - bit 6 - Flaga nadmiaru (wartość 0 jeśli podczas operacji nie wystąpił nadmiar, w przeciwnym wypadku 1)
  - bit 7 - Flaga niedomiaru (wartość 0 jeśli podczas operacji nie wystąpił niedomiar, w przeciwnym wypadku 1)
- Wyjście **Compare** (4 bity) - informacje na temat porównania liczb X i Y. Opis bitów (Bit 0 jest na pozycji najmłodszej, a 4 na najstarszej):
  - bit 0 - wartość 1 jeśli liczba X jest większa od Y
  - bit 1 - wartość 1 jeśli liczba X jest równa Y
  - bit 2 - wartość 1 jeśli liczba X jest mniejsza od Y
  - bit 3 - wartość 1 jeśli wynik porównania nie jest znany (Flaga Unordered)

2.3 Układ porównujący

2.3.1 Opis funkcjonalności układu

- Układ kombinacyjny
- Poprawne porównywanie liczb znormalizowanych i zdenormalizowanych
- Poprawne wykrywanie niepoprawnego wyniku porównania

2.3.2 Zasada działania

Porównywanie liczb zmienno przecinkowych można rozwiązać na dwa sposoby. Liczby porównywane można od siebe odjąć i sprawdzić znak wyniku oraz to czy wynik jest zerem. Można też porównywać kaskadowo każdy element liczby (znak, wykładnik oraz mantysa) i na podstawie relacji między tymi elementami stworzyć ostateczny wynik. W układzie zaimplementowana została druga opcja.
Najważniejszym elementem w porównaniu jest znak (liczba dodatnia jest zawsze większa od ujemnej). W przypadku takich samych znaków, wynikiem porównania jest porównanie wykładników, a jeśli one są sobie równe, to wynikiem jest porównanie mantys. W przypadku jeśli obie liczby są ujemne, to należy odwrócić wynik prównania (Liczba ujemna o mniejszej wartości bezwzględnej jest większa).
Standard IEE-754 mówi od dwóch wyjątkowych sytuacjach które mogą się zdarzyć podczas prównywania liczb:

- W standardzie istnieje 0 oraz -0, te liczby mają być sobie równe
- Porównywanie liczby z nieliczbą, lub nieliczby z nieliczbą zawsze ma ustawiać flagę "Unordered", która oznacza że wynik porównania nie jest znany

2.3.3 Napotkane problemy

W pierwotnym projekcie układu wynik nie był wyświetlany poprawnie jeśli obie liczby były ujemne. W przypadku równości znaków układ porónywał moduły liczb, co było poprawnym wynikiem w przypadku porównywania dwóch liczb dodatnich. Jednak w przypadku porównania dwóch liczb ujemnych, liczba z mniejszym modułem jest większa. Rozwiązaniem tego problemy było odwócenie wyniku porównania, w przypadku wykrycia dwóch liczb ujemnych

2.3.4 Testy jednostkowe

- Ilość testów: **841**
- Oczekiwane wyniki: **784**
- Inne wyniki: **57**

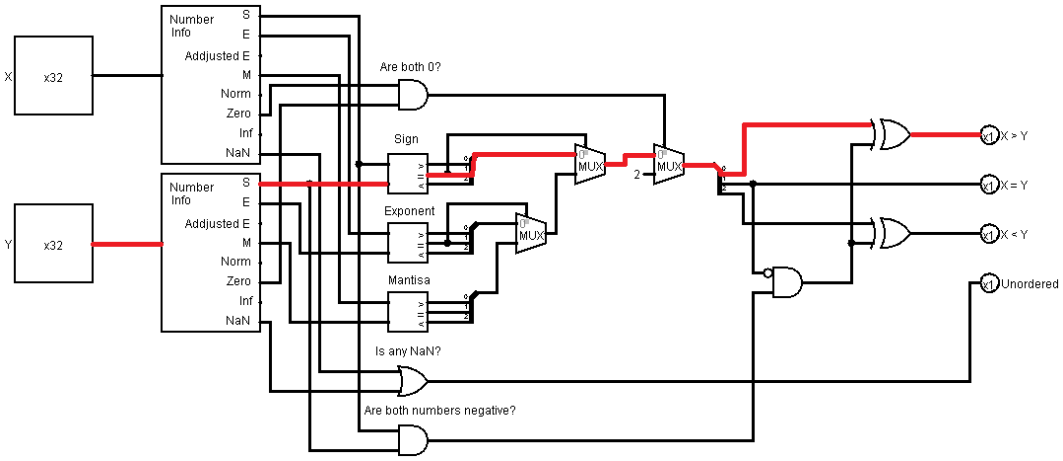
Dla podanego wektora testowego tylko testy, w których występowało porónanie z nieliczbą były błędne. Wynika to z faktu, że język *C++* nie generuje wyniku porównania z nieliczbą. Zaimplementowany układ informuje o tym błędzie za pomocą ustawienia wartości odpowiedniej flagi, ale mimo to nadal podaje wynik tego porównania tak jakby porównywane wartości były by poprawnymi liczbami. Nie jest to zachowanie niezgodne ze standardem, ale jest ono inne od konwencji przyjętej w języku *C++*

2.3.5 Oszacowanie parametrów układu

Oczacowanie zostało opracowane na podstawie danych z rozdziału 2.1

Komponent	Ilość	A	T
AND/OR/NAND/NOR	16	16	16
XOR/XNOR	2	4	4
Sumator/Subtraktor	2	448	256
Komparator	3	678	390
Multiplekser	5	15	15
Suma	28	1161	681
Ścieżka krytyczna:	72		

Tabela 2: Parametry obszaru i opóźnienia dla komparatora



Rysunek 1: Oszacowanie ścieżki krytycznej dla komparatora

Największy wpływ na czas działania układu mają komparatory. Pozostałe komponenty mają mały wpływ na opóźnienie układu

2.4 Układ dodająco/odejmujący

2.4.1 Opis funkcjonalności układu

- Układ kombinacyjny
- Poprawne wykrywanie niepoprawnego wyniku operacji
- Możliwość dodawania oraz odejmowania liczb przez jeden układ
- Poprawne operacje na liczbach znormalizowanych jaki i zdenormalizowanych
- Wykrywanie nadmiaru oraz niedomiaru podczas operacji
- Układ wykorzystuje zaokrąglanie przez obcięcie

2.4.2 Zasada działania

Podczas dodawania na początku należy znaleźć większą, co do modułu liczbę. Następnie obliczana jest różnica wykładników, o tę różnicę przesuwana w prawo (czyli dzielona przez 2 do potęgi różnicy wykładników) jest mantysa liczby mniejszej. Jest to niezbędny krok, gdyż wyrónuje on wartość wykładników obu liczb, przez co możliwe jest ich dodanie lub odjęcie. Po tym kroku następuje dodanie mantys i próba sprowadzenia liczby do postaci znormalizowanej. Do sprowadzenia liczby do postaci normalnej korzysta się z układu wyszukującego ilości zer wiodących. Od ustalonego wykładnika odejmuje się ilość zero oraz przesuwa wynik dodawania mantys w lewo o tę wartość. Istnieje możliwość, że liczby nie da się znormalizować (np. jeśli w mantysie nie ma żadnej "jedynki", lub jeśli odjęcie od wykładnika skutkowało by liczbą ujemną), w takim wypadku liczba zostaje w postaci zdenormalizowanej.
Wyjątki opisane w standardzie IEE-754 dotyczące dodawania i odejmowania:

- Dodawanie i odejmowanie liczby od nieskończoności ma zawsze generować nieskończoność (odpowiednio dodatnią, bądź ujemną)
- Operacje na dwóch nieskończonościach o tym samym znaku mają generować nieskończoność o znaku nieskończoności wejściowych
- Operacje na dwóch nieskończonościach o różnych znakach mają generować nieliczbę i informować o niepoprawnym wyniku

2.4.3 Napotkane problemy

W pierwotnej wersji układu instniał problem z odejmowanie tych samych liczb. Problem ten został rozwiązany za pomocą modułu porównującego liczby. Jeśli wykyto, że moduły liczb są takie same oraz ich znaki były różne, to wtedy obie liczby były zastępowane liczbą 0. Dodawnie do siebie dwóch 0 zawsze skutkuje wynikiem równym 0, czyli jest to również oczekiwana wartość odjęcia od siebie dwóch tych samych liczb.
Innym napotkanym problemem było odejmowanie od liczby znormalizowanej zdenormalizowanej w przypadku kiedy wynik też musiał być liczbą zdenormalizowaną. Problem tkwił w wykrywaniu niedomiaru, który brał tylko pod uwagę, czy podczas odejmowania od obliczonego wykładnika została zasygnalizowana pożyczka. Problem został rozwiązany za pomocą sygnalizowania niedomiaru w wypadku porzyczki oraz w przypadku kiedy wynikiem odejmowania jest 0.

2.4.4 Testy jednostkowe

- Ilość testów: **841**
- Oczekiwane wyniki: **755**
- Inne wyniki: **86**

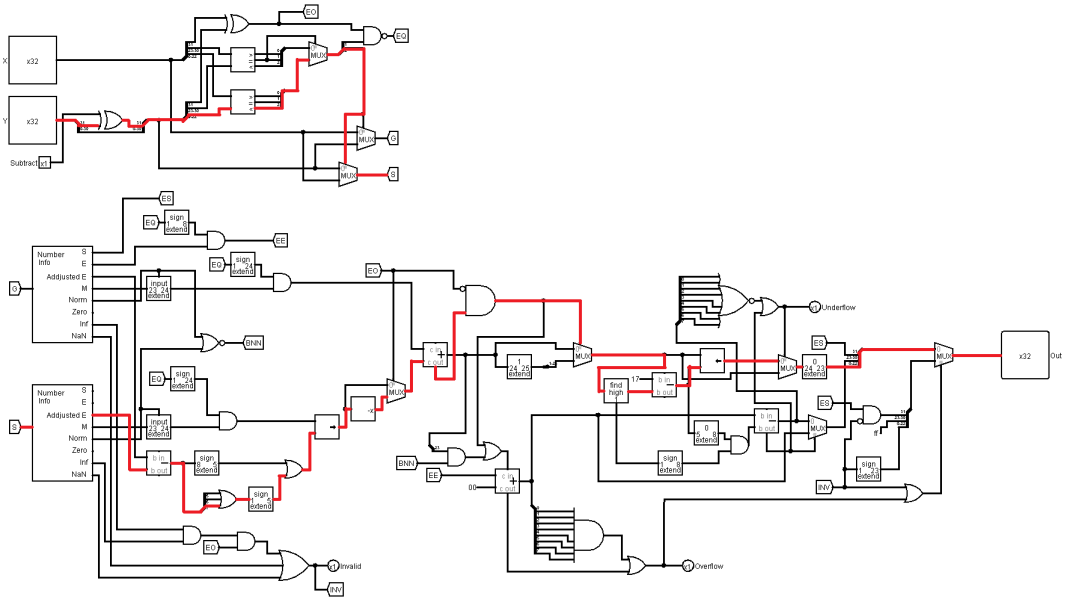
Większość przeprowadzonych testów dała oczekiwane wyniki. Niektóre testy dały wyniki inne, niż te wygenerowane przez język *C++*. Wynika to z faktu, że zaimplementowany układ zaokrągla liczby przez obcięcie, a wartości w wektorze testowym są obliczane z zaokrąglaniem do najbliższej reprezentowalnej wartości. Język *C++* nie rozróżnia też 0 i -0, więc w miejscach w któych układ obliczył -0, program testowy wygenerował wartość 0.

2.4.5 Oszacowanie parametrów układu

Oczacowanie zostało opracowane na podstawie danych z rozdziału 2.1

Komponent	Ilość	A	T
AND/OR/NAND/NOR	31	31	31
XOR	2	4	4
Sumator/Subtraktor	7	1568	896
Obliczanie uzupełnienia	1	224	128
Komparator	2	452	260
Przesuwanie w prawo/lewo	2	384	384
Wyszukiwanie ilości zer wiodących	1	672	384
Multiplekser	10	30	30
Suma:	56	3365	2117
Ścieżka krytyczna:		787	

Tabela 3: Parametry obszaru i opóźnienia dla sumatora/subtraktora



Rysunek 2: Oszacowanie ścieżki krytycznej dla sumatora/subtraktora

Na ścieżce krytycznej znajduje się dużo komponentów o wysokich opóźnienieniach (sumatory, subtraktory, przesuwanie bitów, wyszukiwanie zer wiodących). Pozostałe komponenty mają stosunkowo mały wpływ na długość ścieżki krytycznej

2.5 Układ mnożący

2.5.1 Opis funkcjonalności układu

- Układ kombinacyjny
- Poprawne wykrywanie niepoprawnego wyniku operacji
- Możliwość mnożenia dwóch liczb
- Poprawne operacje na liczbach znormalizowanych i zdenormalizowanych
- Wykrywanie nadmiaru oraz niedomiaru
- Układ wykorzystuje zaokrąglane przez obcięcie

2.5.2 Zasada działania

Podczas mnożenia należy najpierw odjąć obciążenie od wykładników mnożonych liczb. Następnie należy dodać do sumy 127 lub 128 w zależności od tego czy ostatni bit mantysy jest zerem lub jedynką. Na wyjście przekazywany jest bit znaku który uzyskujemy operacją XOR dla znaków wejściowych, wspomniany wykładnik, i przesunięta mantysa w zależności od tego czy czy najstarszy bit wyniki operacji mnożenia obu mantys jest jedynką bądź zerem. Do obsłużenia liczb zdenormalizowanych, wykorzystano liczbę zer wiodących, o którą przesuwamy mantysę by uzyskać znormalizowany wynik. Konieczne było też rozważenie przypadku w którym mnożenie dwóch liczb skutkowało wynikiem zdenormalizowanym. Zostało to rozwiązane przez sprawdzenie czy dodanie do siebie dwóch wykładników wyniesie wartość wykraczającą po za zakres liczb znormalizowanych zgodnie ze standardem IEE-754. Jeśli tak, wyjściowy wykładnik był ustawiany na same zera, a mantysa przesuwana o określoną liczbę miejsc wygenerowaną przez początkowe zera by uzyskać poprawny wynik.

2.5.3 Napotkane problemy

Problematyczne i czasochłonne okazało się obsługiwanie wartości specjalnych. By uwzględnić wszystkie przypadki w implementacji, dane wejściowe porównywane są z wartościamy specjalnymi a następnie ustawiane są odpowiednie flagi w zależności od kombinacji operacji. Przez taką implementację układ jest bardziej złożony

2.5.4 Testy jednostkowe

- Ilość testów: **841**
- Oczekiwane wyniki: **534**
- Inne wyniki: **307**

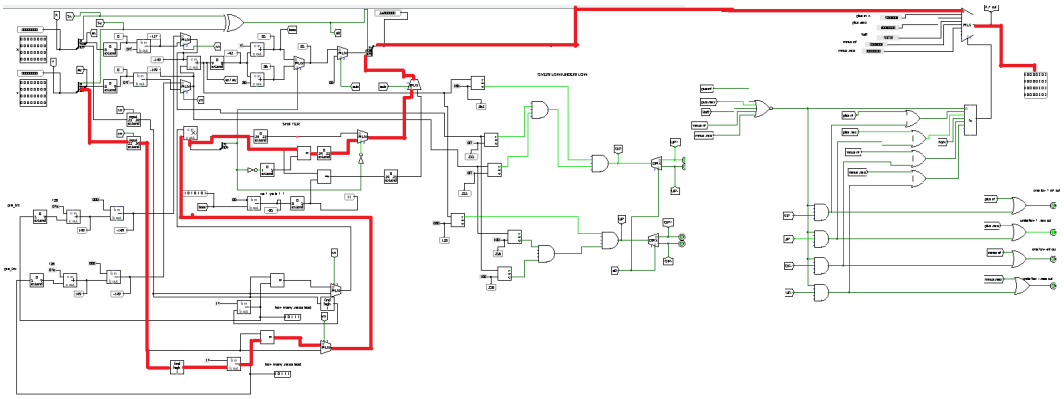
Większość przeprowadzonych testów dała oczekiwane wyniki. Niektóre testy dały wyniki inne, niż te wygenerowane przez język *C++*. Wynika to z faktu, że zaimplementowany układ zaokrągla liczby przez obcięcie, a wartości w wektorze testowym są obliczane z zaokrąglaniem do najbliższej reprezentowalnej wartości. Język *C++* nie rozróżnia też 0 i -0, więc w miejscach w któych układ obliczył -0, program testowy wygenerował wartość 0. Liczba błędów jest wieksza, niż w przypadku dodawanie, gdyż dane testowe częściej były zaokrąglane podczas mnożenia niż dodawania

2.5.5 Oszacowanie parametrów układu

Oczacowanie zostało opracowane na podstawie danych z rozdziału 2.1

Komponent	Ilość	A	T
AND/OR/NANA/NOR	67	67	67
XOR	1	2	2
Sumator/Subtraktor	12	2688	1536
Komparator	22	4972	2860
Przesuwanie w prawo/lewo	4	768	768
Wyszukiwanie ilości zer wiodących	3	2016	1152
Multiplekser	11	33	33
Układ mnożący	1	5728	3520
Suma:	121	16274	9938
Ścieżka krytyczna:		2016	

Tabela 4: Parametry obszaru i opóźnienia dla układu mnożącego



Rysunek 3: Oszacowanie ścieżki krytycznej dla układu mnożącego

Największy wpływ na długość ścieżki krytycznej ma układ mnożący. Jego opóźnienie stanowi ok. 75% długości ścieżki krytycznej.

3 Wnioski

Wartość opóźnienia na oszacowanej ścieżce krytycznej jest bardzo duża. Wynika to z bardzo złożonych operacji jakie trzeba wykonać, aby uzyskać poprawny wynik (W układzie znajduje się dużo sumatorów, subtraktorów, komparatorów oraz innych złożonych układów). Z uwagi na to duże opóźnienie, nie będzie można podłączy zaprojektowanej jednostki zmiennoprzecinkowej do układów ze zbyt wysokim taktowaniem zegara, gdyż wyjścia mogą nie zdążyć się odpowienio ustawić i wynik będzie niepoprawny. Problem ten można by rozwiązać, bez zmian w projekcie, poprzez odczekanie większej ilości cykli zegara. Jednak najlepszym rozwiązaniem było przeprojektowanie jednostki, tak żeby stała się układem sekwencyjnym. W ten sposób można by rozdzielić skomplikowaną operację na parę mniejszych segmentów. Dzięki temu ścieżka krytyczna każdego segmantu była by krótsza, co umożliwiło by podłączenie jednostki do układu z szybszym zegarem. Przy odpowiednim projekcie możliwe było by nawet potokowanie operacji, co również zwiększyłoby wydajność jednostki.
Można by uzyskać wyniki bardziej zbliżone do rzeczywistych wartości, poprzez zastosowanie innego sposobu zaokrąglania niż przez obcięcie, Zwiększyło by to precyzję wyników, kosztem większej złożoności obszarowej i czasowej układu.

4 Źródła

1. Vinayak Patil, Aneesh Raveendran, David Selvakumar - *Out of order floating point coprocessor for RISC V ISA*
   https://www.researchgate.net/publication/281405300_Out_of_order_floating_point_coprocessor_for_RISC_V_ISA (Dostęp 06.06.2021)

2. IEEE Standard for Floating-Point Arithmetic - https://ieeexplore.ieee.org/document/8766229 (Dostęp 06.06.2021)