Architektura komputerów

Laboratorium 3 w terminie 26 kwietnia 2021

Jakub Superczyński 241381

9 maja 2021

1 Cel laboratorium

Celem laboratorium było napisanie w języku assemblera w architekturze 32-bitowej na platformę Linux kalkulatora liczb zmiennoprzecinkowych z interfejsem użytkownika. Program ma wczytywać z klawiatury (z wykorzystaniem biblioteki *libc*) dwie liczby typu *double*, następnie wczytać pożądaną operację, a na końcu oczekiwany sposób zaokrąglania. Po wczytaniu wszystkich potrzebnych informacji, program ma wykonać odpowiednie działanie wykorzystując koprocesor FPU i wyświetlić wynik na ekranie. Dodatkowo program informuje jeśli pojawił się jakiś wyjątek.

2 Opis implementacji

2.1 Wypisywanie na ekran i wczytywanie danych

Do komunikacji z użytkownikiem wykorzystuję bibliotekę *libc*, a dokładnie funkcje *printf* do wypisywania oraz *scanf* do wczytywania danych. Dzięki temu ćwiczenie to pomogło nauczyć standardowego spsosobu wywoływania funkcji z C w języku assemblera. Najpierw musimy zaimportować odpowiednie symbole poleceniem *extern*. Następnie, gdy chcemy daną funkcję wywołać, musimy na stos położyć wszystkie potrzebne argumenty w kolejności odwrotnej niż w definicji funkcji. Dla wypisywania tekstu na ekran jest to adres łańcucha znaków który chcemy wypisać oraz adres łańcucha formatującego. Dla wczytywania danych jest to adres w pamięci do którego chcemy zapisać dane oraz po raz kolejny adres łańcucha formatującego (w tym stringu podajemy typ zmiennej jakiej oczekujemy - czyli double). Następnie, gdy już wrzuciliśmy wszystkie parametry, wywołujemy funkcję instrukcją *call*.

2.2 Obsługa zmiennego przecinka

Po prawidłowym wczytaniu danych same operacje matematyczne są stosunkowo proste do wykonania. Ponieważ wczytałem je jako typ double, są przechowywane w pamięci w postaci ciągów 64-bitowych zgodnych ze standardem IEEE, czyli w

postaci na której koprocesor może wykonywać obliczenia. Najpierw musimy odpowiednio skonfigurować jednostkę zmiennoprzecinkową. Konfiguracja dokonuje się poprzez zamianę odpowiednich bitów słowa kontrolnego. Większość ustawień pozostawiam domyślną, interesuje mnie jedynie część odpowiedzialna za ustawienia zaokrąglania. W zależności od wyboru użytkownika ładuję słowo kontrolne za pomocą instrukcji *fldcw* z odpowiednio ustawionymi bitami zaokrąglania. Następnie należy wykonać odpowiednie działanie. W tym celu ładuję obydwie liczby na stos (co ważne, jest to odzielny stos koprocesora, dlatego korzystam z instrukcji *fld*) oraz wywołuję odpowiedni rozkaz arytmetyczny w zależności od wyboru użytkownika. Po wykonaniu rozkazu wynik operacji znajduje się na szczycie stosu, dlatego pozostaje wypisać go na ekran.
Po wypisaniu wyniku pozostaje jeszcze sprawdzenie ewentualnych wyjątków. Również jest to stosunkowo proste. Przygotowane mam maski bitowe sprawdzające bity odpowiadające poszczególnym wyjątkowm. Sprawdzam więc każdy możliwy wyjątek po kolei, jeśli któryś z nich został zgłoszony, wypisuję na ekran odpowiedni komunikat.

3 Podsumowanie

Program udało się ukończyć w stopniu zadowalającym. Wydaje się że podawane wyniki są poprawne, program również sygnalizuje wyjątki. Ćwiczenie nauczyło nie tylko jednostki FPU, ale również wywoływania zewnętrznych funkcji. Początkowo problem stanowiło zrozumienie w jaki sposób program będzie zapamiętywał wczytane dane z klawiatury - nie byłem pewny czy potrzebna będzie konwersja. Szybko udało się to jednak rozwiązać przez czytanie dokumentacji oraz, przede wszystkim, przez napisanie części programu i uruchomienie go w debuggerze. Zrozumienie wywoływania funkcji oraz obsługi koprocesora wymagało nieco czasu, jednak po przeczytaniu dokumentacji nie stanowiła większego problemu.