Organizacja i architektura komputerów

Magda Cieżka 235083

Wtorek TP 7:30

1 Cel laboratorium

Celem laboratorium było zapoznanie sie z działaniem jednostki zmiennoprzecinkowej oraz opanowanie jej podstawowych rozkazów.

2 Zadanie

Zadaniem było napisanie programu w jezyku assemblerowym, który wykonywałby wszystkie rodzaje podstawowych operacji arytmetycznych na liczbach zmiennoprzecinkowych pojedyńczej i podwójnej precyzji (dodawanie, odejmowanie, mnożenie, dzielenie), pozwalałby na manipulowanie precyzja obliczeń oraz typem zaokraglenia oraz wywoływał wszystkie dostepnie w standardzie IEEE-754 wyjatki (+0,-0,+INF,-INF,NaN).

3 Jednostka zmiennoprzecinkowa

Jednostka zmiennosprzecinkowa lub inaczej koprocesor arytmetyczny wspomaga procesor przy wykonywaniu obliczeń zmiennoprzecinkowych. Współcześnie stanowi nierozłaczna i integralna cześc układu scalonego zawierajacego CPU. W przeszłości jednak była całkowicie opcjonalna i niedostepna na wielu komputerach.

4 Przebieg pracy nad programem

4.1 Słowo sterujaca, precyzja i zaokraglenia

Słowo sterujace ustala precyzje obliczeń, tryb zaokraglenia oraz maski wyjatków. Słowo sterujace jest zawsze 16-bitowe. Ustawieniami tymi najlepiej manipulować poprzez dodawanie do słowa sterujacego odpowiednich wartości, które zmienia ustawienie bitów na odpowiednich pozycjach.
Rodzaje precyzji użyte w programie.

- precyzja pojedyńcza (ustawienie bitów w słowie sterujacym 00)
- precyzja podwójna (ustawienie bitów w słowie sterujacym 10)

Tryby zaokraglenia:

- do najbliższej liczb (ustawienie bitów w słowie sterujacym 00)
- w dół (ustawienie bitów w słowie sterujacycm 01)
- w góre (ustawienie bitów w słowie sterujacym 10)
- do zera - inaczej obciecie (ustawienie bitów w słowie sterujacym 11)

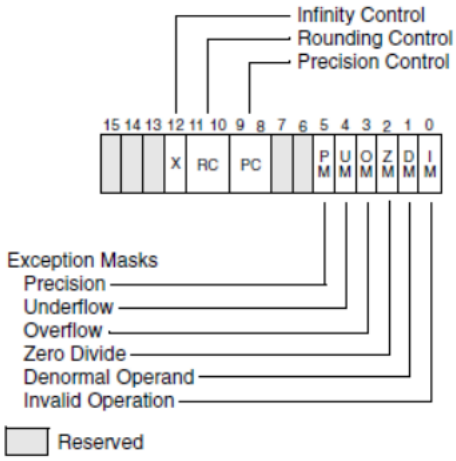

Figure 1: Budowa słowa sterujacego

Listing 1: Implementacja operacji matematycznych

```
ustawienie_slowa_sterujacego:
mov cw,%eax
mov double,%ebx
add %ebx,%eax
mov up,%ebx
add %ebx,%eax
mov %eax,cw
fldcw cw
```

4.2 Działania na liczbach

Jednostka zmiennoprzecinkowa udostepnia nam instrukcje umożliwiajace dokonanie podstawowych działań na liczbach zmiennoprzecinkowych. Sa to:

- Dodawanie- rozkaz faddp
- Odejmowanie- rozkaz fsubp
- Mnożenie - rozkaz fmulp
- Dzielenie - rozkaz fdivp

Przed

Listing 2: Implementacja operacji matematycznych

```
dodawanie_float:
flds float1
flds float2
faddp

odejmowanie_float:
flds float1
flds float2
fsubp

mnozenie_float:
```

```
flds float1
flds float2
fmulp
dzielenie_float:
flds float1
flds float2
fdivp
```

```
(gdb) run
Starting program: /home/magda/Programming/OIAK/Lab3/calculator

Breakpoint 1, exit () at calculator.s:99
99      movl $SYSEXIT,%eax
(gdb) info float
  R7: Valid   0x400199999a0000000000 +4.800000190734863281
  R6: Valid   0xc002a428f60000000000 -10.26000022888183594
  R5: Valid   0xc003a474885ce0760000 -20.55690071640015049
=>R4: Valid   0xbffdb9a020622c232800 -0.36254979322808892788
  R3: Empty   0xc000aeb8520000000000
  R2: Empty   0x00000000000000000000
  R1: Empty   0x00000000000000000000
  R0: Empty   0x00000000000000000000

Status Word:         0x2020                PE
                       TOP: 4
Control Word:        0x1a7f   IM DM ZM OM UM PM
```

Figure 2: Przykładowe wyniki obliczeń w debbugerze

4.3 Wyjatki

Wyjatkami, które należało wywołać były:

- NaN - wywołany przy pomocy operacji dzielenia zera przez 0
- +INF - wywołany przy pomocy operacji dzielenia liczby dodatniej przez zero
- -INF - wywołany przy pomocy operacji dzielenia liczby ujemnej przez zero
- +0 - wywołany przy pomocy operacji dzielenia zera przez liczbe dodatnia
- -0 - wywołany przy pomocy operacji dzielenia zera przez liczbe ujemna

Listing 3: Implementacja wyjatków

```
wyjatki:
#NaN:
fldz
fdiv zero

#+inf:
flds float1
fdiv zero

#-inf
```

```
flds  float2
fdiv  zero

#+0
fldz
fdiv  float1

#-0
fldz
fdiv  float2
```

```
(gdb) run
Starting program: /home/magda/Programming/OIAK/Lab3/calculator

Breakpoint 1, wyjatki () at calculator.s:99
warning: Source file is more recent than executable.
99      movl $EXIT_SUCCESS,%ebx
(gdb) info float
  R7: Special 0xffffc000000000000000 Real Indefinite (QNaN)
  R6: Special 0x7fff8000000000000000 +Inf
  R5: Special 0xffff8000000000000000 -Inf
  R4: Zero    0x00000000000000000000 +0
=>R3: Zero    0x80000000000000000000 -0
  R2: Empty   0x00000000000000000000
  R1: Empty   0x00000000000000000000
  R0: Empty   0x00000000000000000000

Status Word:         0x1805   IE    ZE
                       TOP: 3
Control Word:        0x146d   IM    ZM OM    PM
```

Figure 3: Widok wyjatków w debugerze

Wszystkie wyjatki zostały wywołane poprawnie.

5 Uruchomienie programu

Aby uruchomić prgram nazleży wykorzystać z narzedzia make i pliku makefile, wktórym zapisane sa komendy do kompilacji i linkowania wzbogacone o odpowiednie flagi. Wartości wyników działań oraz wywoływanych wyjatków można sprawdzić przy pomocy debbugera.

6 Podsumowanie

Po skompilowaniu, linkowaniu i uruchomieniu w debugerze program zadziałał z godnie z przewidywaniami. Podczas jego programu najwieksza trudność sprawiły takie zagadnienia jak słowo sterujace, ustawienie precyzji oraz typu zaokraglenia. W zrozumieniu i implementacji owego zagadnienia pomocna okazała sie zamieszczona w bibliografii pozycja.

7 Bibliografia

http://jedrzej.ulasiewicz.staff.iiar.pwr.wroc.pl/Architektura-Komputerow/lab/Architektura-79.pdf