Київський національний університет імені Т.Шевченка

Звіт

До лабораторної роботи №2

ІМІТАЦІНА МОДЕЛЬ ПРОЦЕСОРА

Кирило Байбула Аленович
Група К-21
Факультету комп'ютерних наук
та кібернетики

Київ
2021

МЕТА

Необхідно розробити програмну модель процесора та реалізувати його імітаційну (тобто комп'ютерну) модель. Мені було запропоновано індивідуальний варіант, в якому визначена конкретна:

1. Адресність процсеора - 2-адресний;
2. Бітність процесора - 14-бітний;
3. Обов'язкова для реалізації команда процесора - цілочисельне віднімання;

Також мають обов'язково бути реалізовані:

1. розміщення інтерпретуємої програми у текстовому файлі (наприклад, один рядок=одна команда);
2. мінімум 2 команди (одна з них - занесення значення у регістр, інші задаються варіантом);
3. для операндів/регістрів представлення побітно, можливо, для деяких варіантів із побайтним групуванням бітів;
4. фіксація у регістрі стану як мінімум знаку результату виконання команди;
5. потактове виконання команд (наприклад, 1-й такт – занесення команди у регістр команди, 2-й такт - виконання операції і занесення результату).

НАПИСАННЯ ЛАБОРАТОРНОЇ

Лабораторна була написана мовою прогамування **GO** у системі сімейства Unix . За основу для описання команд у текстовому файлі був взятий синтаксис асемблера **nasm**, де **mov** - оператор занесення значення у регістр та між регістрами і **sub** - оператор цілочисельного віднімання. Результат останього заноситься у перший операнд. Поточний стан процесора у програмі має виглядати як поточна команда, яку потрібно виконати(IR), список регістрів і побітовому вигляді(R1-R8), знак останьої резулбтату виконання останьої дії(PS), лічильник виконаних команд(PC), лічильник тактів(TC);

Програма має у стандартний поток виводу заносити команду яку він збирається виконувати, поточний стан процесора (його регістри, всі його флаги та лічильники) та стан після виконання команди. Наприклад, так буде виглядати обробка рядку **mov R3, -2**:

```
IR: mov R3, -2
R1: 00000000000000
R2: 00000000000000
R3: 00000000000000
R4: 00000000000000
R5: 00000000000000
R6: 00000000000000
R7: 00000000000000
R8: 00000000000000
PS: +
PC: 0
TC: 1

IR: mov R3, -2
R1: 00000000000000
R2: 00000000000000
R3: 11111111111110
R4: 00000000000000
R5: 00000000000000
R6: 00000000000000
R7: 00000000000000
R8: 00000000000000
PS: -
PC: 1
TC: 2
```

Також, як можна побачити вище, від'ємні числа зберігаться по іншому на відміну від більшості сучасних Сі-подібних мов програмування. Для лабораторної числа потрібно було побітово інвертувати і додавати 1, але, на щастя, **GO** вже так числа і хранить, тому ніяких додаткових дій не потрібно було робити.

Приведемо приклад виконання лабораторної для **sub**:

```
IR: sub R3, 3
R1: 00000000000000
R2: 00000000000000
R3: 11111111111110
R4: 00000000000000
R5: 00000000000000
R6: 00000000000000
R7: 00000000000000
R8: 00000000000000
PS: -
PC: 1
TC: 1

IR: sub R3, 3
R1: 00000000000000
R2: 00000000000000
R3: 11111111111011
R4: 00000000000000
R5: 00000000000000
R6: 00000000000000
R7: 00000000000000
R8: 00000000000000
PS: -
PC: 2
TC: 2
```

ВИСНОВОК

Для обробки команд у 2-адресному процессорі достатньо лише двох тактів. Бітність процессора не впливає на алгоритми виконання команд, а впливає лише на вмістимість числових регістрів. У регістрах зберігається вся інформація, яку обробляє процесор, вони необхідні для виконання команд, зв'язку процесора з іншими частинами обчислювальної системи, та більша частина коду програм складається саме з обробки значень регістрів. Така система роботи не є ідеальною, але через застійлі стандарти та сучасний стан індустрії такий підхід є необхідним. Також для реалізації підпрограми нашому процесору потрібний стек та команди для переходу між мітками програмами та завантаження та вигружання значень зі стеку.

Посилання

- Код лабораторної: `https://github.com/Velnbur/AsmVM.git`
- Вимоги лабораторної: `https://sites.google.com/site/byvkyiv1/arhiteom_stac/arhiteom_lab_06`