ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

В теоретической части данных методических указаний особое внимание уделяется вопросам организации цикла исполнения инструкции ядром центрального процессора. Краткий обзор теоретического материала, необходимого в ходе выполнения лабораторной работы, приводится ниже.

Процедура выполнения команд

Процессор работает под управлением программы, состоящей на самом низком уровне из последовательности двоичных кодов машинных инструкций/команд. Чем сложнее машинная инструкция, тем сложнее организована структура процессорного ядра.

В модели CISC-архитектуры (к которой относиться и Intel) процессор исполняет поток сложных многотактных CISC-команд, которые очень неудобны и сложны для организации управляющих блоков ядра ЦП. Однако такая модель удобна для программирования, т.к. позволяет закодировать сложную операцию с данными из ОП. Если использовать парадигму RISC–архитектуры, то простота реализации аппаратной части ЦП и высокая скорость их работы объясняется несложным управлением исполнением потока простых однотактных RISC-команд с ограниченным набором операндов. Для объединения преимуществ обоих архитектур процессоры ВС используют программную модель CISC-архитектуры, затем трансляцию удобных, но сложных CISC-команд в простые RISC-микрооперации (мопы[1]) и параллельное их исполнение множеством исполнительных блоков [1].

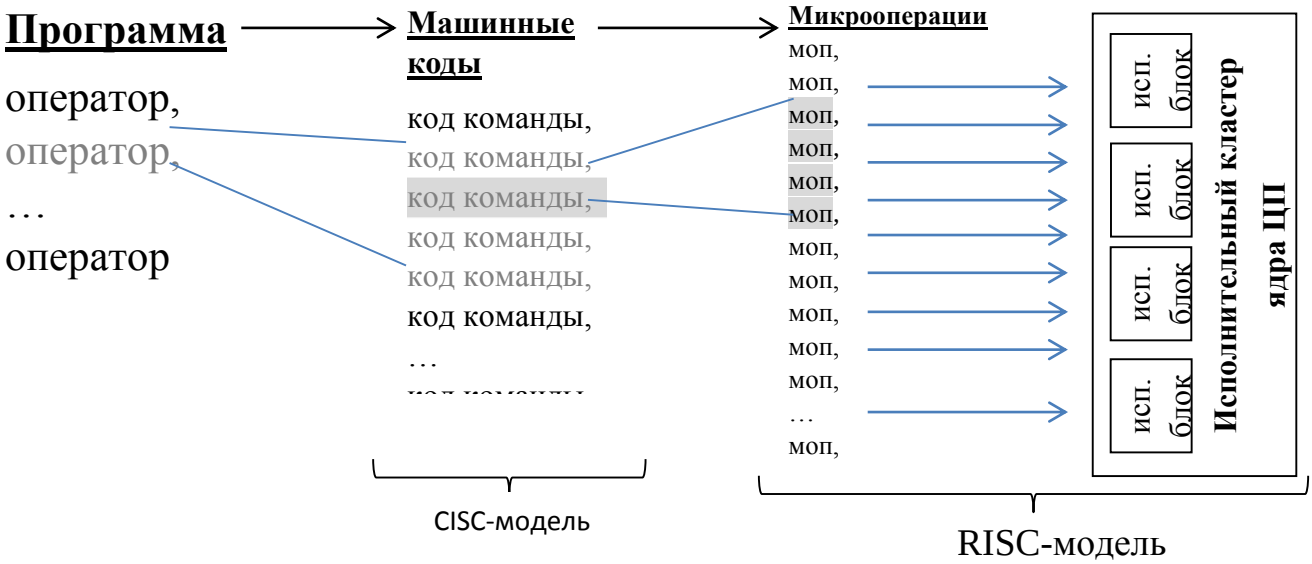

Рис.1. Порядок преобразования кода программ в исполняемые ядром ЦП микрооперации

При выполнении каждой команды процессор выполняет некоторую последовательность действий, называемую циклом выполнения. Каждый цикл

---

[1] – *микрооперация (μ-операция = μоп = моп)*

состоит из нескольких фаз, включающих от одной до нескольких микроопераций – элементарных действий, выполняемых одним из узлов операционного устройства в течение одного такта.

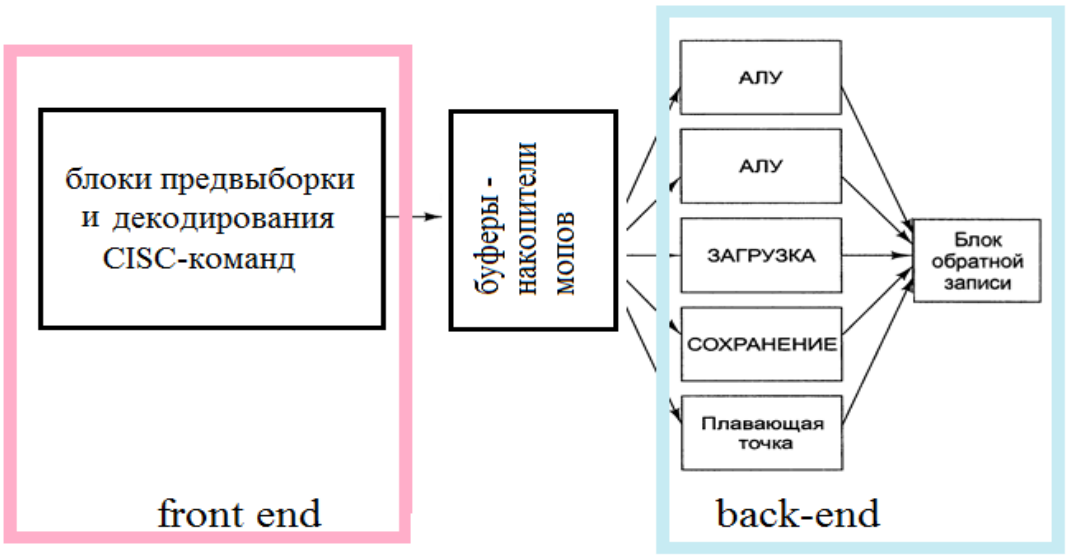

Рис.2. Две части конвейера (front-end и back-end) ядра ЦП (трансляция CISC-to-RISC).

Стандартные фазы цикла работы ЦП включают в себя [2]:

- Ф1.Выборку команды.
- Ф2.Дешифрацию команды.

} Относятся к front-end *(in order cluster)*

- Ф3.Выборку операнда/ов.
- Ф4.Исполнение операция.
- Ф5.Запись результатов.

} Относятся к back-end *(OOO – out of order cluster)*

Первые две фазы цикла выполняются начальными ступенями конвейера ядра ЦП, называемыми front-end (предвыборка), а последние три – исполнительным кластером ЦП, называемым back-end (исполнение-запись) – см. рис.2. По сути это два независимых конвейера с накопительным буфером микроопераций между ними[2]. Если буфер мопов переполнен, то работа front-end приостанавливается [4].

Блоки предвыборки конвейера (front-end)

Для ускорения работы процессора команды считываются из памяти программ (КЭШа инструкций LI1) «пачками» определённого размера (по несколько штук) в буфер предварительной выборки (см. рис.3).

---

[2] *В некоторых архитектурах процессоров (Intel NetBurst, лежавшая в основе микропроцессоров Pentium 4, Pentium D, Celeron и Xeon [3]) этот буфер был организован в виде КЭШа последовательностей микроопераций и заменил КЭШ L1I инструкций. Эти процессоры могли поддерживать более высокие тактовые частоты, но имели плохие показатели по рассеиваемой мощности и дальнейшее развитие пошло по пути совершенствования архитектуры семейства P6 за счёт более эффективной конвейеризации, внеочередного исполнения и многоядерности.*

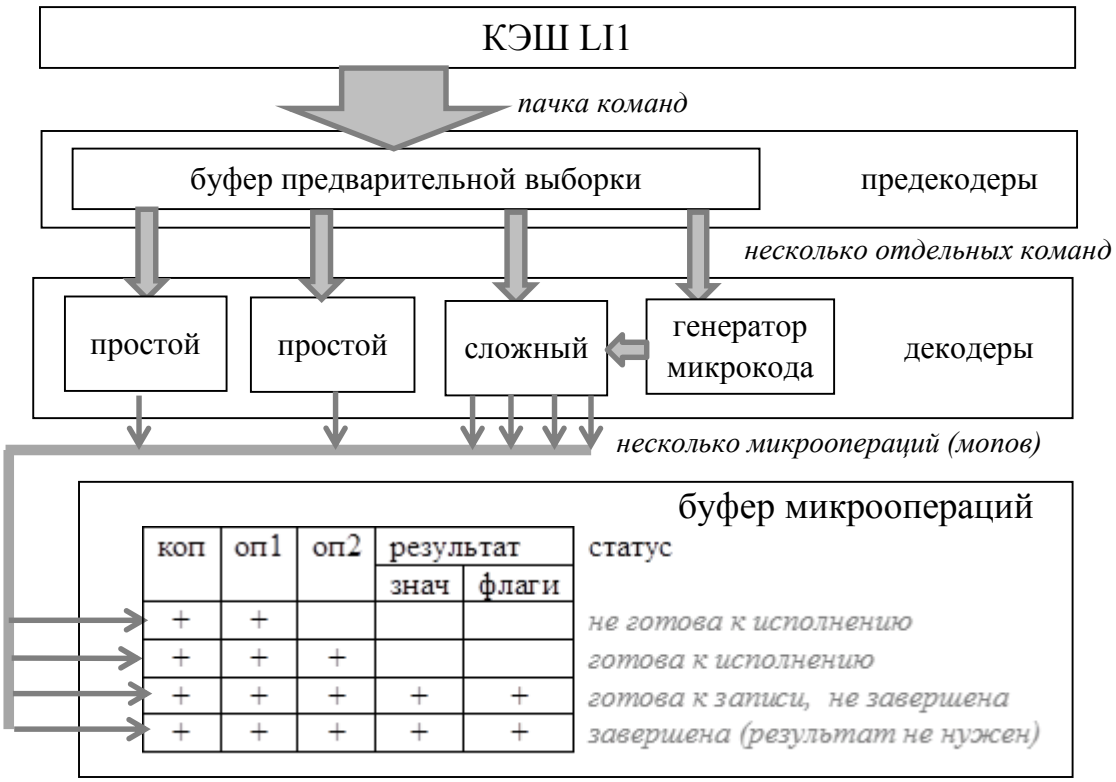

Рис.3. Предварительная выборка и декодирование команд в мопы с последующим сохранением в буфере микроопераций

Далее происходит их последовательная дешифрация несколькими рядами дешифраторов (декодеров): разбиение пачки команд на отдельные команды блоком определения длины команд, определение сложности команд (предекодер), трансляция в последовательность микроопераций (мопов) несколькими дешифраторами (simple decoder / complex decoder / microinstruction sequencer).

Вся очередь микроопераций хранится в специальном буфере микроопераций. Структура буфера микроопераций подразумевает хранение кода микрооперации (коп), кодов операндов, кодов результата и его признаков, статуса микрооперации: не готова к исполнению, готова к исполнению, исполнена (результат нужен следующим командам/мопам), завершена (результат не нужен) [4].

Команды и микрооперации

Каждую команду можно условно отнести к классам: передачи данных (mov), обработки данных (proc), передачи управления (jcc) и дополнительные команды/мопы (управления режимом работы ЦП, КЭШ, отладкой…).