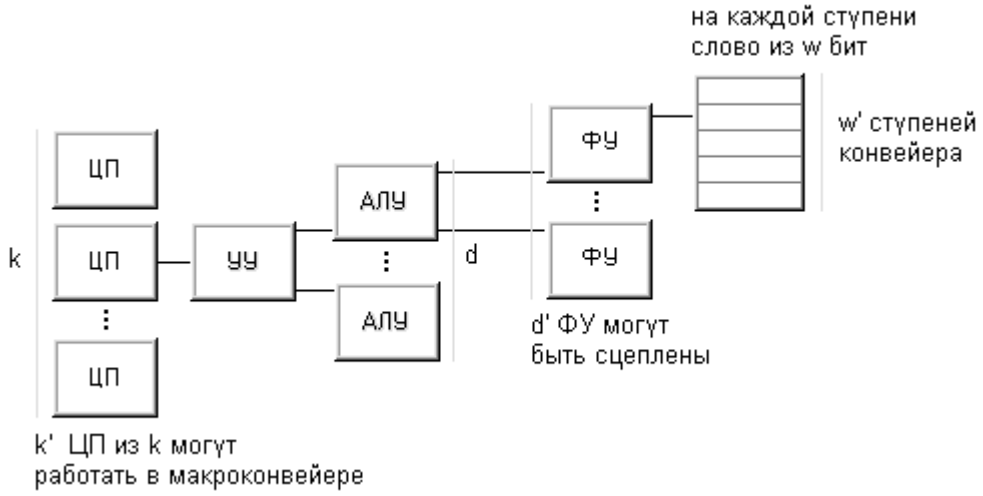
Image not found or type unknown

В основу классификации В.Хендлер закладывает явное описание возможностей параллельной и конвейерной обработки информации вычислительной системой. При этом он намеренно не рассматривает различные способы связи между процессорами и блоками памяти и считает, что коммуникационная сеть может быть нужным образом сконфигурирована и будет способна выдержать предполагаемую нагрузку.

Предложенная классификация базируется на различии между тремя уровнями обработки данных в процессе выполнения программ:

- уровень выполнения программы - опираясь на счетчик команд и некоторые другие регистры, устройство управления (УУ) производит выборку и дешифрацию команд программы;
- уровень выполнения команд - арифметико-логическое устройство компьютера (АЛУ) исполняет команду, выданную ему устройством управления;
- уровень битовой обработки - все элементарные логические схемы процессора (ЭЛС) разбиваются на группы, необходимые для выполнения операций над одним двоичным разрядом.



Таким образом, подобная схема выделения уровней предполагает, что вычислительная система включает какое-то число процессоров каждый со своим устройством управления. Каждое устройство управления связано с несколькими арифметико-логическими устройствами, исполняющими одну и ту же операцию в каждый конкретный момент времени. Наконец, каждое АЛУ объединяет несколько

элементарных логических схем, ассоциированных с обработкой одного двоичного разряда (число ЭЛС есть ничто иное, как длина машинного слова). Если на какое-то время не рассматривать возможность конвейеризации, то число устройств управления k , число арифметико-логических устройств d в каждом устройстве управления и число элементарных логических схем w в каждом АЛУ составят тройку для описания данной вычислительной системы C:

t(C) = (k, d, w)

В таких обозначениях описания некоторых хорошо известных вычислительных систем будут выглядеть следующим образом:

t(MINIMA) = (1,1,1);
t(IBM 701) = (1,1,36);
t(SOLOMON) = (1,1024,1);
t(ILLIAC IV) = (1,64,64);
t(STARAN) = (1,8192,1) - в полной конфигурации;
t(C.mmp) = (16,1,16) - основной режим работы;
t(PRIME) = (5,1,16);
t(BBN Butterfly GP1000) = (256,~1,~32).

Несмотря на то, что перечисленным системам присущ параллелизм разного рода, он без особого труда может быть отнесен к одному из трех выделенных уровней.

Теперь можно расширить возможности описания, допустив возможность конвейерной обработки на каждом из уровней. В самом деле, конвейерность на самом нижнем уровне (т.е. на уровне ЭЛС) это конвейерность функциональных устройств. Если функциональное устройство обрабатывает w-разрядные слова на каждой из w' ступеней конвейера, то для характеристики параллелизма данного уровня естественно рассмотреть произведение w×w'. Знак умножения × будем использовать на каждом уровне чтобы отделить число, представляющее степень параллелизма, от числа ступеней в конвейере. Компьютер TI ASC имеет четыре конвейерных устройства по восемь ступеней в каждом для обработки 64-х разрядных слов, следовательно, он может быть описан так:

t(TI ASC) = (1,4,64×8)

Следующий уровень конвейерной обработки - это конвейеризация на уровне команд. Предполагается, что в вычислительной системе есть несколько функциональных устройств, которые могут работать одновременно в рамках

одного потока команд (в настоящее время используется специальный термин для обозначения данной возможности - сцепление функциональных устройств). Классическим примером этому могут служить компьютеры фирмы Cray Research(link is external). А исторически первой, по всей вероятности, является машина CDC 6600, содержащая десять независимых последовательных функциональных устройств, способных подавать результат своей работы на вход другим функциональным устройствам, образуя единый поток команд:

t(CDC 6600) = (1,1×10,~64)

(описан только центральный процессор без учета управляющих и периферийных подсистем).

Наконец, нам осталось рассмотреть конвейеризацию на самом верхнем уровне, известную как макро-конвейер. Поток данных, проходя через один процессор, поступает на вход другому, возможно через некоторую буферную память. Если независимо работают n процессоров, то в идеальной ситуации при отсутствии конфликтов и полной сбалансированности получаем ускорение в n раз по сравнению с использованием только одного процессора. Так компьютер PEPE, имея фактически три независимых системы из 288-ми устройств, описывается следующим образом:

t(PEPE) = (1×3,288,32)

После расширения трехуровневой модели параллелизма средствами описания потенциальных возможностей конвейеризации каждая тройка

t(PEPE) = (k×k',d×d',w×w')

интерпретируется так:

- k - число процессоров (каждый со своим УУ), работающих параллельно
- k' - глубина макроконвейера из отдельных процессоров
- d - число АЛУ в каждом процессоре, работающих параллельно
- d' - число функциональных устройств АЛУ в цепочке
- w - число разрядов в слове, обрабатываемых в АЛУ параллельно
- w' - число ступеней в конвейере функциональных устройств АЛУ

Очевидна связь между классификацией Фенга и классификацией Хендлера: для получения максимальной степени параллелизма в терминах Фенга надо найти произведение всех шести величин в описании Хендлера. Здесь же заметим, что

заложив в основу своей схемы явное указание на присутствующий параллелизм и возможную конвейеризацию, В.Хендлер сразу снимает массу вопросов, характерных для предшествующих схем Флинна, Шора и Фенга, по крайней мере, в плане описания векторно-конвейерных машин.

В дополнение к изложенному способу описания архитектур Хендлер предлагает использовать три операции, которые будучи примененными к тройкам, позволят описать:

- сложные структуры с подсистемами ввода-вывода, хост-компьютером или какими-то другими особенностями;
- возможные режимы функционирования вычислительных систем, поддерживаемые для оптимального соответствия структуре программ.

Первая операция (×) в каком-то смысле отражает конвейерный принцип обработки и предполагает последовательное прохождение данных сначала через первый ее аргумент-подсистему, а затем через второй. Описание упомянутого выше компьютера CDC 6600 можно уточнить следующим образом:

t(CDC 6600) = (10,1,12) × (1,1×10,64),

где первый аргумент отражает существование десяти 12-ти разрядных периферийных процессоров и тот факт, что любая программа должна сначала быть обработана одним из них и лишь после этого передана центральному процессору для исполнения. Аналогично можно получить описание машины PEPE, принимая во внимание, что в качестве хост-компьютера она использует CDC 7600:

t(PEPE) = t(CDC 7600) × (1×3, 288, 32) = = (15, 1, 12) × (1, 1×9, 60) × (1×3, 288, 32)

Поток данных последовательно проходит через три подсистемы, что мы и отразили, соединив их знаком '×'.

Заметим, что все подсистемы последнего примера достаточно сложны и, вообще говоря, исходя только из данного описания могут представляться по-разному. Чтобы внести большую ясность, аналогично операции конвейерного исполнения, Хендлер вводит операцию параллельного исполнения (+), фиксирующую возможность независимого использования процессоров разными задачами:

t(n,d,w) = [{(1,d,w) + ... + (1,d,w)] {n раз}

В случае CDC 7600 уточненная запись вида:

(15, 1, 12) × (1, 1×9, 60) = [(1, 1, 12) + ... +(1, 1, 12)]} {15 раз} × (1, 1×9, 60)

говорит о том, что каждая задача может захватить свой периферийный процессор, а затем одна за одной они будут поступать в центральный процессор. И наконец третья операция - операция альтернативы (V), показывает возможные альтернативные режимы функционирования вычислительной системы. Чем больше для системы таких режимов, тем более гибкой архитектурой, по мнению Хендлера, она обладает. Например, компьютер C.mmp может быть запрограммирован для использования в трех принципиально разных режимах:

t(C.mmp) = (16, 1, 16) V (1×16, 1,1 6) V (1, 16, 16).