Annotated Buzzwords

From Fall 2005 through Fall 2016

0, 1, 2, 3 address machine
: An n address machine is an ISA where n indicates the number of operands that need to specified EXPLICITY for binary operations in the ISA. The ADD is a good example of a binary operate instruction. In a 0 address machine, none need to be specified, since the microarchitecture obtains both sources from the top of the stack, and pushes the result onto the stack. In a 3 address machine, like the LC-3b, for example, both sources and the destination ALL need to be specified explicitly (ADD R1,R2,R3). The x86 is a 2 address machine.

1's complement
: one representation of integers

2's complement
: one representation of integers

2-bit saturating counter
: advance the counter depending on what the branch does, predict according to the current high bit of the counter.

2-level adaptive branch predictor
: The branch predictor we invented in 1990, where the branch decision is based on what the processor did in previous cycles where its history was the same as the current history. We need a history to store the current history, and a table of two bit counters to store what the processor did in past when it had exactly the same history.

2-pass assembler
: A simple assembler that makes two passes at an assembly language program, first to identify all the symbols and generate the symbol table, and second, to generate the actual bit patterns of the code.

3D chips
: We used to think of chips wherein all the functionality is in the x,y plane. A 3D chip recognizes that we can get additional functionality by stacking dies in the z direction. Ergo, 3D chips. We note that connecting to a single die in the x,y plane has bandwidth limited by the number of pins one can place in the perimeter of the die (4s, if the square die has length s). Connectivity between stacked dies is limited by s^2, the area of the die. Much greater bandwidth which is one of the scarce commodities in current chips.

access control violation (ACV)
: an exception caused by one of two things. Either the address was ill-formed and does not belong to the virtual address space of that process (limit violation) or the privilege of the process does not allow the access specified (protection violation).

Access to storage devices
: The names RAM, DASD, and sequential access storage device are given to storage devices based on the storage locations that can be accessed after the previous access. With RAM, the location accessed next is independent of the previous location accessed. With sequential access storage devices, the location accessed next is the next physical location after the previous location accessed. With DASD (direct access, storage devices), access is to a larger granularity unit (usually called a page) such the start of the access is independent of the previous unit of access, but the rest of the unit is accessed from seequential physical locations.

accumulator
: The term given to the gpr when there is only one gpr.

ACMP

Assymetric Chip Multiprocessor. The name given to our multi-core chip which consists of one heavyweight processor (core) and many lightweight processors.

active window

see restricted data flow.

address generation

The phase of the instruction cycle where the address of a memory operand is computed.

address lines

ditto, except address bits.

address space

the number of distinct memory addresses. That is, the maximum number of memory locations possible such that each can be specified by a unique address. Older machines, the original x86, for example, had an address space of 2^16. We sometimes say: the x86 had 16 bits of address space. The original IBM 360, I believe, had an address space of 2^24. Most ISAs today, have an address space of 2^32, and some have an address space of 2^64.

address translation

conversion of a virtual address to the physical address where the virtual location actually exists, so that the processor can read/write values into that location.

address unknown problem (memory disambiguation problem)

A ST instruction has an addressing mode that requires the contents of a register. A younger LD instruction has a known address. Can the load take place before the store. It depends on whether the addresses they are accessing are different. What if the address associated with the store is not yet known. We have a problem. Maybe yes, maybe no. I called it the unknown address problem. Josh Fisher carried the day with his name for it: memory disambiguation. The requirement to disambiguate these two address. That is what everyone calls it today. What to do about it :

- Conservative: Wait until the address of the store is known. The plus is it is simple. The minus is it loses performance.
- Aggressive: Assume they are different. The plus is you don't wait. The minus is that if you are wrong, you need to throw away a lot of work and start over. The logic to do that is substantial.

Address, Data and Control Wires

A bus is made up of wires. Some are used for carrying the addressed required in a transfer, some for the data being transfered, and some carry control signals to arbitrate for the bus and coordinate the transfer. BBSY, SACK, BR, BG, MSYN, SSYN are examples of control signals.

addressability

the number of bits stored at each memory location. Most modern computers are byte addressable. That is, 8 bits are stored at each location.

addressing modes

mechanisms for obtaining operands required by an instruction. For example, the LDW instruction obtains the address of the memory location to be accessed by adding an offset to a Base Register. We say the addressing mode is Base+offset. The JSR instruction computes the starting address of the subroutine by adding an offset to the PC. We say the addressing mode is PC-relative.

advantages/disadvantages of condition codes

aerial density

the number of bits per unit length along a track.

agents of evolution

My (YNP) term, expressed in my Proceedings of IEEE, November 2001 issue on microprocessors where I attribute all developments in microarchitectures to one of three agents: a bottleneck to be removed, a performance improvement to be achieved, or simply good luck.

algorithm

a step by step procedure, wherein each step is definite (precisely defined), effectively computable (able to be carried out by a computer), and wherein the procedure is guaranteed to terminate (finite). For a problem to be solved with a computer, one must be able to construct an algorithm to solve it.

allocate on write miss

the act of setting aside space in the cache for data to be stored, if the line written to is not in the cache.

always taken/always not taken

the prediction is determined by the designers of the chip.

Amdahl's Law

observation made by Gene Amdahl that maximum speed up of a parallel processor is limited by the fraction of the problem that can not take advantage of more than one processor.

Amdahl's Law

an observation by Gene Amdahl that most problems have a parallel part and a sequential part, such that the addition of p processors reduces the execution time of the parallel part by p, but does nothing to improve the execution of the sequential part. We call the sequential part, for which the p processors can do nothing, the sequential bottleneck. If we let a equal the part of the problem that is the parallel part, then we can state Amdahl's Law in terms of speed up :

$$\textbf{Speedup}(p) = \textbf{Time}(1) \div ((a \times \textbf{Time}(1)) \div p + (1-a) \times \textbf{Time}(1))$$

anti-dependency

using a register to store more than one value. Before we write to that register, all instructions which need to read the old value have to read it. Sometimes called write after read hazard.

arbitration

the act of determining who gets the bus for the next bus cycle.

architectural/physical register

Architectural registers are what the software knows about. Physical registers are internal storage that the microarchitecture uses to maintain the correctness of the execution. The re-order buffer for example is a file of physical registers.

Architecture Bits

term used to describe the bits in an FPGA that particularize the function of its logic blocks and their connectivity.

arithmetic/logical shift

A right shift is conveniently used to perform an integer divide by 2, where the value being divided is expressed in 2's complement representation. Thus, we need to insert into the high bit position the sign bit of the value. On the other hand, bit patterns are often right-shifted for other (logical) purposes, in which case we need to insert a 0 into the the high (left-most) bit position. Given these two uses, we need to differentiate an arithmetic shift where we insert the sign bit into the left-most position from a logical shift where we insert a 0 into the left-most position. [Note that an arithmetic left-shift, corresponding to multiply by 2, requires inserting a 0 into the right-most bit position, exactly the same as what we would do for a logical left-shift. Therefore, we generally do not differentiate arithmetic left shift from logical left shift.]

array processor

a set of processors such that all execute the same program, each executing the identical instruction in lock step, each step of the way.

assembly language

a language very close in semantics to the ISA of the machine, with the advantage that bit patterns are replaced by mnemonic devices, and some additional useful features are often present. Nonetheless, the defining characteristic of an assembly language is that the level of control over changing the machine state is very much like what one would have if one programmed in 0s and 1s. Also, sometimes called a "low level" language to distinguish it from "high level" languages that are more tuned to the expressiveness of humans. High level languages (more than 1000 out there): C#, C++, Java, C, all the way back to antiquity, when Fortran and Cobol showed up. Assembly language programs are translated by an assembler which does little more than replace all the mnemonic devices with 0s and 1s. It would be a tough job to try to assemble a program written in the assembly language for ISA A into the 0s and 1s of ISA B. On the other hand, high level language programs are generally translated by a compiler. Since the instructions in the high level language are quite a bit removed from the intricasies of any ISA, it is not difficult to change the ISA that a compiler translates the high level language program into.

associative access

access is based in part at least on the contents of the storage location. That is, part of the address is actually stored in the storage location. Two common examples that you have already seen of this are (1) the tag store of the cache, where the entry contains the tag bits, which are part of the address, and (2) the TLB where each entry is a page number and its corresponding PTE. The page number is the associative part of the access.

asynchronous bus

no clock, bus cycles take as long as they need to take. Since no clock, handshaking is required.

atomic unit

a unit that can not be further partitioned without losing its intrinsic identity. It takes its meaning from chemistry where an atom consists of protons, neutrons, and electrons, but once partitioned into these particles, ceases to have the identifying characteristics of the atom. Oxygen is no longer oxygen when the 6 electrons and 6 protons (if I remember my high school chemistry correctly) are lopped off. We say an instruction is the atomic unit of processing since we execute code at that lowest granularity. That is, we never execute a piece of an instruction. Either the whole instruction or none of it. That is, we do not change the machine state corresponding to a piece of an instruction and leave the machine state corresponding to another piece of the same instruction unchanged.

Back-to-Back

Two instructions (usually operates) located sequentially in the I stream, wherein the result of the first is a source of the second.

Backup registers

Cray program controlled cache, sort of. Eight S registers are backed up by 64 T registers. Instead of going to memory again and again, one can do a vector load to the T registers.

balance set

the set of processes that have their working sets in memory.

balanced design

the whole point of a pipelined implementation is to allow the various parts of the microarchitecture to do their respective things concurrently. It makes no sense to provide capability in one stage of the pipeline that goes unused in the next stage. Better to use the extra transistors required to provide that capability for something more useful. Example: a fetch engine that can fetch and decode 6 instructions each cycle, but only has a single ALU for executing these instructions is not a balanced design. There is no point wasting resources to be able to fetch and decode 6 instructions concurrently if the instructions just sit in a queue downstream waiting for the single ALU.

bandwidth problem

There are a relatively small, finite number of pins on a chip. In order for a core to process instructions, it must obtain instructions (at least the first time they are executed) from off the chip, and obtain data, often from off the chip. We define the bandwidth as the rate at which the bits describing those instructions and data are able to get onto the chip. The more cores on the chip, the more these cores compete for the available bandwidth. This we call the bandwidth problem.

barrel shifter/rotator (shift matrix rotator)

a large combinational network that allows an n-bit value to be rotated (or simply shifted) an arbitrary number of bit positions. Example, if n is 32, the 32-bit value can be shifted 0, 1, 2, ... 31 bits. The combinational structure could be implemented with 32 32-to-1 muxes, but is more efficiently implemented in multiple layers as a shift tree, as described in class. Inputs to a barrel shifter is the n-bit source and a log n bit encoding of the shift amount. Output is the rotated result.

barrier synch

a node having k inputs and k outputs, such that when all inputs have tokens, the node fires, producing on each output line the token that was on its corresponding input line.

baseline

in any measurement where we are trying to evaluate the benefit of a feature, the baseline is the system that does not include the feature. Note: Some less than honest purveyers will quote benefits due to a feature by selecting a very wimpy baseline. In the case above which discusses Speedup with p processors, a poor baseline would be to use an algorithm for one processor that is a particularly bad algorithm if we were restricted to one processor. Thus we would have a large Time(1), which would make Time(p) look better than it really is.

Basic block
A sequence of sequential code terminating in a branch.

binade
the interval from 2^n to $2^{(n+1)}$.

Binary Coded Decimal (BCD)
an encoding of integers, such that each decimal digit is encoded in 4 bits. Number of digits is usually arbitrary, which means one needs two pieces of information to specify a BCD number: starting address and number of digits.

Bit Line
In memory, a line that ties together the same bit of all words. For example, a bit line ties together A[28], B[28], C[28] of memory words A,B,C.

block move
A data movement instruction that requires multiple values to be copied to other locations.

Booth's algorithm
an improvement over the standard shift and add algorithm for multiplication by enabling each iteration to consume two bits of multiplier at each step, rather than one.

branch misprediction penalty
the number of cycles wasted due to branch misprediction where everything speculatively fetched needs to be thrown away before we can start along the correct instruction path.

branch prediction
rather than wait for the condition code to be resolved upon which a branch instruction is based (taken or not not taken) and creating a hole in the pipeline, the hardware guesses and fetches based on that guess. The guess is called a branch prediction.

branch target buffer (branch history table)
A mechanism that is helpful in predicting branches. it generally contains the target address of a taken branch plus a bit to tell the instruction fetch mechanism whether to predict taken or the fall through path. The prediction can be used to select the source of a mux that will get loaded into the PC for the next cycle. If the prediction is taken, we load the address from the BTB. If we predict not taken, we load the existing PC plus the instruction size in bytes.

bread and butter design
neither the supply of transistors on the chip nor the design time of the designer are infinite. With finite resources, bread and butter design suggests applying both to the chip's bread and butter, that is, those opcodes, addressing modes, and data types that are important to optimize for the expected usage of the chip.

BTFN
guess taken if the branch is to an earlier location in the code, not taken if the branch is to a location later in the program. This is consistent with the behavior of for and while constructs in high level languages. Best example of a compile time predictor.

burst error
.
A burst error is a sequence of bits that have been transmitted incorrectly, as opposed to a single bit error, wherein all bits except one are transmitted correctly.

burst mode
successive data transfers in the same bus cycle.

bus busy
A control signal indicating that a bus cycle is in progress.

bus cycle
one unit of bus transaction time.

bus grant
signal from the PAU granting the bus

bus master
device controller in charge of the current bus cycle.

bus request
signal from a controller requesting a bus cycle

bus
: an interconnection network where every element is connected to the same wire. Only one transaction can occur on the bus at any point in time. This is the structure of worst case contention. See my handout on interconnection networks.

Buzbee's observation
: Bill Buzbee observed that the time required to execute a task with p processors was not simply aT1/p + (1-a)T1, as specified by Amdahl's Law. He introduced an additional term sigma(p), such that Tp = aT1/p + (1-a)T1 + sigma(p). The extra term reflected the extra overhead required for coordinating the activities of the p processors.

byte rotator
: a piece of combinational logic that rotates the bytes of a bit pattern. Example, the 32 bits 0xFFFCC771, if byte rotated 3 bytes left would become 0x71FFFCC7.

byte write
: Although memory is usually byte addressable, the memory bus is usually multiple bytes wide. A byte write is a memory access which stores one byte of information into a designated byte-addressable location while ignoring the rest of the bytes on the memory bus. Implementation requires write enable signals of byte granularity.

byte-addressable
: see addressability.

cache coherence problem (cache consistency)
: in a shared memory multiprocessor, it is most common for each processor to have its own private cache. If more than one processor have in their respective caches the contents of the same memory location, it is important that those respective contents be identical. We say that caches are coherent when this is the case. Caches are not coherent, for example if both P1 and P2 cache location A, but P1 has one value for A in its cache, and P2 has a different value for A in its cache.

cache consistency/coherency
: in a shared memory multiprocessor system, where each processor has its own cache, it is important that the contents of a single location that is stored in multiple caches has the same information in all caches that store that location. If yes, we say the cache is coherent with respect to that location.

cache hit/miss
: on a load or a store, the cache is queried to determine if the location's information is contained in the cache. If yes, we say it is a cache hit. If no, we say it is a cache miss.

cache line (block)
: the granularity of storage, see cache memory. This size is generally greater than the size of a single access, due to our faith that spatial locality exists.

cache memory
: a small, fast-access memory structure that stores a subset of the physical memory, thereby reducing memory access time if the desired location is present in the smaller structure. Since the cache only stores a subset of the physical memory, it must store both the addresses of the locations it contains as well as the data at those locations. The granularity of a cache entry is a called a cache line (or synonymously, a cache block).

Cache set
: A set consists of all cache lines sharing the same index bits in their address. Usually thought of as a "row" in the Tag Store.

Cache warm up
: when a processor starts processing an executable image, the cache contains no code or data associated with that image. Warm up is that initial transient period when almost every access to the cache results in a cache miss. ...that is, until the cache becomes effective with respect to supplying instructions and/or data.

CAM
: Content addressable memory. Memory that is accessed via associative access.

capability based machines
: processors that had the feature of only allowing an instruction to be executed if the process that is running had the necessary privilege (or capability) to execute. People thought this was a good idea at the time (1970s) as a security protection. The problem was that as a result of all the time it took to check these

privileges, the programs executed much too slowly. Data General built the Fountain Head, but never shipped it. IBM produced System 38, which was not a success. Intel produced the i432 chip set which was not successful.

carrier sense multiple access/collision detect (CSMA/CD)

The way the Ethernet works. The ethernet is an asynchronous bus. Device controllers are attached to the bus. Any controller can access any bus at any time. How do we keep from transmitting garbage. Answer: CSMA/CD. Each controller senses the carrier to see if anyone else is accessing the bus. If yes, the controller does not access the bus. All (multiple) the controllers are sensing the carrier all the time. If no one else is using the bus, the controller is free to put signals on the bus. However, it may be, due to the nature of the asynchronous bus and propagation time of those signals that controller A may think there is no one else using the bus because the signals being propagated by controller B may not have propagated all the way to A yet. Thus, controller A must continue to monitor the bus to be sure it is always the case that the signals controller A is putting on the bus are the same as the signals controller A is monitoring. If they differ, controller A has detectd a collision (CD). In this case, controller A stops transmitting, waits a random amount of time, and then tries again. Meanwhile, controller B will have detected the collision due to A, so it stops transmitting, waits a random amount of time, and then tries again. The hope is that randomA is not equal to randomB. If it does, we repeat.

central arbitration

arbitration is done by one central arbiter. (e.g., PAU)

channel

Modern microprocessors have multiple ports to the memory system. We call each port a channel. Bits of the address designate which channel to use in accessing the DRAMs. For example, a chip with four channels would have two address bits for designating the channel.

check sum

a mechanism for detecting burst errors in a large stream of bits. The bits to be transmitted are run through a linear feedback shift register bit serially. Some of the bits are fed back as shown in the handout on error detection and correction. Say the shift register has k stages. At the end of n cycles, the bit stream has been input to the shift register. At that point, there are k bits in the shift register. Those k bits are then shifted out and added to the transmitted stream. We call those k bits a check sum. At the other end, we repeat the operation and compare check sums. If they are not identical, an error was introduced during transmission.

checksum

the bits used to detect whether a burst error has occurred within a large transfer of information, where the errors are not statistically independent. Suppose I wish to transfer N bits of information from X to Y. A common implemenation of checksum error detection is to input the N bits sequentially into an n-bit linear feedback shift register at X concurrently with transferring them to Y. After the transfer is complete, the n bits remaining in the shift register are then sequentially transferred to Y as well. At the destination Y, the process is repeated and the contents of the n-bit shift register after N bits have been input is compared with the last n bits received at Y. If they match, the transfer is assumed to have gone error free. Typical values for N is in the thousands of bits, typical values for n is 16 or 32.

Chinese remainder theorem

That if I have a set of k moduli, $p_1, p_2, \ldots p_k$ that are pairwise relatively prime, and if I represent each value as the set of k residues, modulo their corresponding moduli, then the residues of the values 1 to the product of the k moduli are all distinct.

chip enable

output pins of a chip float and have no effect if the chip enable bit is not asserted.

chopping

one of the four rounding modes. Rounding is accomplished by removing (chopping) the least significant bits. Both positive and negative values are rounded toward zero.

CISC/RISC

In 1980, the term RISC was coined to characterize simple ISAs. The term was an acronym for Reduced Instruction Set Computer, but was ill-advised on a lot of levels, which we may go into late in the semester if there is time. Suffice it to say right now, that the intent was to have each instruction do a single operation in a very streamlined way. SPARC, MIPS, HPPA, Alpha, Motorola 88000 Power-PC are all examples of ISAs that attempted more or less to follow that mantra. x86, Motorola 68000, VAX IBM Zseries are all

examples of ISAs that did not. The acronym czars referred to the "other" ISAs as Complicated Instruction Set Computers. For example, the x86 which can load from memory, add to a register and store back to memory in a single x86 instruction -- three distinct operations. ...which led their advocates to shoot back: CISC stood for Comprehensive Instruction Set Computer.

clock skew

Clock skew recognizes that the clock signal does not change everywhere on the chip at exactly the same time. The effect of this is to render part of the clock cycle useless since we can not begin processing the current clock cycle until we know everything on the chip has seen the change in clock signal.

coarse-grain dataflow

a dataflow graph wherein each node performs a larger granularity function, like perhaps an entire high level assignment statement, or an iteration of a loop body.

code bloat

The opposite of dense encoding of the instruction stream. That is, extra bytes of instruction bandwidth to do the same job because the work to be performed is not encoded densely. For example, for the LC-3b to carry out A = B + R2, we would need three instructions, one to load B into R1 (for example), one to ADD R3,R1,R2, and one to store R3 into location A. These three instructions at 2 bytes each, require a total of 6 bytes. With x86, only one instruction would be required, and it could be done with three bytes.

cold start effect

in general, the transient effect that occurs in most things before a system is in service long enough to be operating in a stable mode. In the case of a cache, this refers to the case where the cache is completely empty and the first several accesses are compulsory misses.

column address strobe (CAS)

see page mode.

column-major order

an array stored in memory in the order: first the components of the first column, then the second column, etc.

common data bus (CDB)

in the data path of the IBM 360/91, the bus used to distribute values and corresponding tags to registers in the RAT and in the reservation stations that are waiting for the value uniquely assigned to that tag.

compatibility

a term used to describe the ability of new implementations of an ISA to execute code written for an older implementation. Intel has been a master at making sure that every new x86 implementation is able to execute correctly all code written for all older x86 implementations. As time passes, it becomes clear that some old opcodes, addressing modes, data types are no longer useful, and some new opcodes, addressing modes, and data types are necessary. In fact, the addition of the SSE1 and SSE2 instruction set to the x86 ISA is a reflection of the new uses of that ISA. However, Intel's insistence on compatibility means that in addition to the new SSE extensions, new implementations also have to include in their microarchitecture the capability to execute the old opcodes, addressing modes, data types, etc. since some customers may want to buy a new laptop but still use their old software.

compilation process

the act of translating a high level program into a machine code object module.

compile-time (static)

that which is done before the program is actually run. For example, a compile-time branch predictor makes its prediction based on information that is available before the program starts execution.

compile-time predictor

the guess is determined by the compiler, usually as result of profiling.

compiler

The software that translates a high level language source program (in C, Java, C++, Fortran, Cobol, APL, PL-1 and thousands of others) into the 0s and 1s of a specific ISA. Compilers are differentiated from Interpreters in the sense that interpreters work one high level language statement at a time (i.e., translate 1st, translate 2nd, translate 3rd) while compilers examine the entire high level language program before generating any 0s and 1s in the ISA.

compulsory miss, conflict miss, capacity miss

The three common causes of a cache miss. A compulsory miss occurs the first time the line is accessed, unless the line has been prefetched into the cache. No other way the line could have gotten into the cache. A capacity miss occurs if a line that used to be in the cache had been kicked out simply because there is not enough room in the cache. A conflict miss occurs if a line was kicked out, not because of the size of the cache, but rather because of the size of the set. That is, the cache was not full, but the set for which the line competes was full. Fully associative caches do not have conflict misses.

condition code 'duals'

Two condition code tests that disagree in each position. For example, p=n=1 and z=0 on the one hand, and p=n=0 and z=1. The first yields TRUE if the two source operands are unequal; the second yields TRUE if the two source operands are equal.

condition codes

single-bit registers (often referred to as "flags") which provide additional information about the result of the last instruction. The LC-3b has three condition codes (N,Z,P). Upon completion of an operate instruction, the condition codes specify whether the result of that operate was negative (N=1,Z=0,P=0), zero (N=0,Z=1,P=0), or positive (N=0,Z=0,P=1). Similarly, upon completion of a load, the condition codes specify whether the value loaded into the destination register was negative, zero, or positive. Most real ISAs also have C and V condition codes. C designates the carry out of an add instruction. V designates whether the previous integer arithmetic instruction resulted in an overflow condition. One example is the case where the source operands of an ADD are both positive 2's complement numbers, but the result produced by the ALU is the representation of a negative number. [Say, I have 4-bit integers. +6 = 0110, +5 = 0101. If I tell the ALU to add them, the output would be 1011, which is the representation for -5. The ALU is saying +6 plus +5 = -5. Ergo, set V.]

conditional nodes

a node having two inputs and two outputs, the outputs being labeled true and false. This node takes two tokens (a value *x* and a boolean value) and passes the value *x* along the output path corresponding to whether the boolean token is true or false.

Consistent state of the machine

term used to describe the state of a machine during the sequential in-order processing of instructions after one instruction has completed execution and before the next sequential instruction has started execution. This represents a point in execution where interrupts and exceptions can be taken.

consistent state

the state of the machine between the execution of instructions. In the middle of execution of an instruction, the internal state of the processor can be whatever the microarchitecture wishes. Even general purpose registers can be clobbered if the microarchitecture wishes. **But**, at the end of each instruction, the contract with the software requires that the state of the machine be recoverable to the state produced by the program up to that point. That point is the consistent state.

constants ROM

A ROM containing a set of constants needed by the microprocessor for processing instructions. Two simple examples are (a) the value x0001 used for incrementing an internal register, and (b) x003F used to extract the low six bits of an instruction by ANDing it with the contents of IR.

constructive microcode

Alternative terms for vertical and horizontal microcode. Vertical microinstructions have fewer bits, perform a single microoperation, and all the bits contribute to that single microoperation. Horizontal microinstructions have many more bits (largest one I know was the Nanodata QM-1 with 360 bits) where the bits are partitioned into fields, each field can designate a microoperation. When microprogramming a horizontal machine, in some sense one is constructing the microinstruction by specifying each of its fields. When one is microprogramming a vertical machine, one is selecting microinstructions from the collection of microinstructions that each do a single microoperation.

consumer

an instruction that requires the result as a source.

contention

one of the three common metrics for goodness of an interconnection network. Contention is a measure of how many independent transfers can be made on the network simultaneously.

context switch

the act of stopping the execution of instructions from one process and starting the execution of instructions from another process. Before the processor can do that, it needs to save the registers associated with the process being stopped, and load the registers associated with the process being started. This process of saving and loading the context is referred to as a context switch.

control dependency

dependency caused by control instructions, such as branches. Which instruction is executed next depends on which way a branch goes is an example of a control dependency.

control instructions

instructions that redirect the instruction stream. The default in instruction processing is to increment the PC to point to the next instruction. If the next instruction is other than the next sequential instruction, a control instruction is needed to provide the new PC. The LC-3b control instructions are BR, JMP, JSR(R), TRAP, and RTI.

control lines

ditto, except control bits.

control signals

signals that control the processing within the data path, the memory, and the microsequencer. The set of control signals that are effective during a single clock cycle are often characterized as a microinstruction.

control space

A term given to the 1 GB partition of the VAX vertical address space that involves control structures, mostly the privileged stacks and the user stack.

control store

a private hardware structure for storing the set of control signals needed in a particular clock cycle. In a microprogrammed machine, the job of the microsequencer is to identify the location in the control store that contains the control signals for the next clock cycle.

copy node

a node which takes a value as input and produces two copies of that value as output. A consequence of pure data flow as an abstraction wherein each input value to a data flow node is consumed when the node executes.

core

Synonym for processor.

cost

one of the three common metrics for goodness of an interconnection network. Cost is a measure of the number of connection points in the network.

CRC check

cyclic redundancy check. A mechanism for detecting burst errors, wherein the individual bit errors are not statistically independent.

critical path design

the cycle time of a processor is one of the key items that specifies the performance of the processor. Recall that performance is to a first approximation determined by

```
1/(length * CPI * cycle time)
```

Decreasing cycle time has a major impact on performance. Cycle time is the maximum of the various logic paths in the data path and its control. Critical path design instructs one to examine the longest path in the machine and attempt to reduce it, either by more clever logic design or by latching a path in the middle and turning a long single-cycle operation into a two-cycle operation.

Critical Section

name given to a segment of code that accesses a particular data structure, wherein that segment of code must complete before another thread can execute code that accesses that data structure.

cycle level simulator

A simulator that changes its state at the granularity of the clock cycle. That is, given the machine state and the control signals asserted during a clock cycle, the cycle level simulator produces the state required by those control signals.

cylinder

the set of tracks that can be read simultaneously.

DAA. Decimal Adjust
An Intel x86 instruction that allows BCD arithmetic to be performed byte by byte on a 2's complement ALU by adjusting the answer when necessary due to the fact that BCD generates a carry after 9, rather than after 15.

daisy chaining
mechanism by which all controllers with the same vertical priority receive the bus grant in order, and pass it on to the next if it does not want it.

Dark Silicon
A term used to describe the notion that since the number of transistors on a chip has grown into the billions, it is getting to be the case that it is no longer feasible to power on all the transistors. We refer to those transistors not powered on at a particularly time "dark silicon." Thus we can use transistors to specify an ASIC that when not needed can remain powered off, but when needed is powered on (presumably something that is not needed at that time will be powered off). Thus, acceleration due to the ASIC when needed, and no power consumed when not needed.

Data Cache
A cache that holds only cache lines consisting of data.

data dependency
one of the three data dependencies: flow, anti, write

data flow graph
a directed graph showing all the instructions as nodes, and the producer/consumer relationships as arcs.

data flow
the paradigm wherein instructions are executed when their data is available, rather than when they show up in program order.

data forwarding
information is sent from where it is produced to where it is needed without going first to a register. Speeds up processing.

data lines
lines containing data bits.

data movement instructions
instructions that move data. Loads/stores to/from memory and I/O instructions.

data type
a representation of information for which the ISA provides instructions that operate on that representation. For example, in the LC-3b, the ADD instruction operates on values represented as 2's complement integers. We say the 2's complement integer is a data type. There is no floating point data type in the LC-3b ISA since there is no instruction in the LC-3b that operates on values represented as floating point numbers. Note that we can still operate on floating point representations by writing a subroutine to operate on these representation. For example, if the LC-3b did have a FLOATADD instruction, the LC-3b could compute a floating point add in ONE instruction. Since it does not, one could write a procedure that isolates the exponent and fractional part of each value, compares the exponents, shifts right the smaller fraction accordingly, does the add, and renormalizes the result. However, such a procedure would take tens of LC-3b instructions.

dataflow graph
A graphical representation of instruction flow, with nodes representing operations to be performed and directed arcs representing the producer/consumer relationship of data. For example, an add operation requires two source operands; these are shown as input arrows to the node. The result of the add is shown as an outward arrow from the node to all other operations that require that result as its source operand.

dataflow node
an operation in a data flow graph.

datapath
the part of the microarchitecture that processes the information. The register file, alu, and various muxes are all part of the data path.

decode

The second phase of the instruction cycle, where the hardware identifies the task that the instruction is to carry out.

decoupled access/execute (DAE)
a relaxation of the VLIW paradigm wherein the processors do not have to execute their instructions in lock step.

dedicated bus
bus dedicated to a single purpose.

Deep pipelining
a qualitative statement about the number of stages in the pipeline. A deep pipeline is one with a relatively large number of stages.

delay slot
The number of instructions between the branch and when it executes. Also refers to the instructions that are contained in those n locations after the branch that will execute before the branch redirects the instruction stream.

delayed branch
A branch instruction whose effect does not take effect immediately. That is the semantics of the branch instruction is: execute n more instructions and then branch.

design point
a specification of the criteria to be optimized in the design process. If we are interested in performance, or reliability, or availability, or low cost, we colloquially say our design point is performance, or reliability, or ...etc.

Destructive Reading
Reading that destroys the value read, and therefore needs to be restored. For example, in a DRAM cell, "reading" discharges a capacitor. Therefore, it must be combined with subsequent logic that recharges the capacitor to the point it was at before the read was initiated.

device controller
that which interfaces the device to the bus, and manages all accesses to/from the device.

device
the I/O unit that houses all the stuff that causes the information to be stored and transformed from one form to another. Disk drive, for example

direct access storage
A term given to the storage mechanism utilized by disks. In fact, IBM has always referred to their disks as DASD (direct access storage devices). The notion is that the device needs to first find the starting address of the access, and the following locations accessed are those at the following sequential addresses. I.e., sequntial locations on the track that is rotating under the disk head. I have added another buzzword, disk terminology, to help make this clear..

direct-mapped cache
a cache characterized by the property that if a line of memory is stored in the cache, it can only be stored in one place.

dirty bit
in a write back cache, a bit that specifies whether the information in the cache line has been changed since it was read in from memory. If not, then on replacement, this line need not be written back to memory.

disk block
the unit of storage accessed from the disk. 512 bytes, for example.

disk crash
when the head actually hits the spinning platter. …and scores the surface, usually. BAD.

disk head
that part of the mechanical arm that contains the ability to read the magnetic material stored on the track, or write to the track.

disk terminology
Information is stored on disk in the following way: The disk consists of multiple rotating platters, much like an old phonograph record (as your grandmother!). If you don't like that analogy, this pizza trays. All rotate about the same vertical access at their center. Each platter consists of concentric tracks (think circles of different radii). Data is picked off the platters by locating a disk head close to but not touching (ouch!)

the platters. There is one head for each platter. Each bit is on a track. The set of tracks form what they call a cylinder. The disk head moves as a unit. At any point in time they are concurrently accessing the n bits of an n platter cylinder. (Hope this is clear. If not, google Wikipedia. I am 99% sure they probably even have a picture. Hopefully you do not need a picture, since what I put on the white board was clear!).

distributed arbitration

each device controller has enough information to know whether or not it gets the bus the next cycle.

divide by 0 exception

an exception caused by a computation that uses finite arguments and produces infinite results. Called "divide by 0" because the simplest example of this is a finite number divided by 0. Another example of this is the tangent function where the argument is 90 degrees.

DMA

direct memory access. Allows memory and the disk (for example) to transfer information without tying up the processor.

DRAM

Dynamic random access memory. A DRAM cell stores a single bit of information as the absence or presence of charge on a capacitor. The capacitor charged is one value, the capacitor uncharged is the other value. A single transistor is required to read (sample the charge) and write (store the charge) the bit. Unfortunately, the capacitor will not maintain its charge indefinitely. There are RC paths to ground that will leak the charge. Therefore, a DRAM cell requires periodic refresh. That is, periodically, capacitors must be sampled and returned to their fully charged or uncharged state.

dual fetch

Dual Fetch

An instruction fetch mechanism that can fetch two instructions each clock cycle. Generally combined with dual decode, dual rename and lots of functional units, so the dual fetch does not create a bottleneck waiting for functional units (adders, multipliers, and the like).

duty cycle

The fraction of time a piece of logic is busy. Before pipelining, the duty cycle of the fetch engine was small since it was not busy during the period after fetch until the instruction is retired. With the advent of pipelining, the duty cycle of each phase goes to 100% if the pipeline is never stalled.

Dynamic branch prediction

branch prediction based on decisions based on run-time information.

dynamic instruction stream

The *sequence* of instructions as they are executed. A single instruction (located in a single address of memory) counts once in the static instruction stream. Its count vis-a-vis the dynamic instruction stream is the number of times it is executed. The dynamic instruction stream (often called a *trace*) gives a record of the order that the intructions were carried out.

Dynamic Recompilation

Compiling that happens after run time has commenced, which enables the compiler to revisit decisions based on the code running the actual (not profiled) data.

dynamic scheduling

As you know, instructions are fetched and decoded in program order. (Program order means the order that they appear in the program.) In the old days almost all machines also executed the instructions in that same order. Then came Tomasulo and the IBM 360/91 and Thornton and the CDC 6600, where the instructions were executed in the order that their source operands became ready. We call the process of scheduling the instructions for execution on the basis of when the source operands become available: dynamic scheduling. The reason is that when that happens is not known until runtime, since for example it depends on cache misses, function unit latencies, etc. What happens at runtime we call "dynamic," what can be determined at compile time, we call "static." Ergo, dynamic scheduling.

dynamic/static interface

the interface between when the compilation process ends, and the run time process begins. The compiler translates the source program producing 0s and 1s that specify the work to be done. The hardware carries out the work of those 0s and 1s at run time. Although it is now changing, the compiler never used to have run-time behavior available to it to make its decisions. It did not know which cache accesses missed, which branches were actually taken, how long certain operate instructions took. It is true that compilers

have for a long time, as part of the compilation process, observed what happened when the program ran on some sample data. This is called *profiling.* But that data is not the *real* data, and in fact can lead to decisions that could be counterproductive, depending on how close to the sample data the real data is. We refer to the DSI as the interface between when all activity is done without real knowledge of real run-time data, and when all activity has that knowledge.

ECC

error correction code. a code which allows a single bit error to be not only detected, but also corrected without requiring the bits to be retransmitted. See my handout on the subject.

Edge-triggered flipflop, Master/Slave flipflop, Transparent Latch

Flip-flops have the property that they can be read and written in the same cycle. During the cycle, the value stored in the flipflop is what is read, while at the end of the cycle the value to be written is actually written. Transparent latches, on the other hand, do not generally allow values to be read and written during the same cycle, since values are written as soon as they get to the latch (i.e., they do not wait until the end of the cycle) which means that subsequent reads of the latch will read the value just written. Two common types of flipflops are the edge-triggered flipflop and the master/slave flipflop. The edge-triggered flipflop behaves as described above: Throughout the clock cycle, the current value is read, and nothing is written. At the end of the clock cycle (on the clock edge), the value to be written during that cycle is actually written, making it available to be read in the next clock cycle. The master/slave flipflop consists of two transparent latches. Call them A and B. During the clock cycle, B is read and A is written. That is, the combinational logic that is processing the contents of the master/slave flipflop sources the output of B, and the result of the combinational logic is written to A. To make this work, we gate A with CLK and we gate B with NOT-CLK. Thus, during the first half of the clock cycle, the combinational logic sources B (which cannot change since NOT-CLK=0) and produces a result which is written into A (since CLK=1). In the second half of the clock cycle, the value of A produced in the first half of the clock cycle is written into B (since NOT-CLK=1). This could change the output of the combinational logic, but since CLK=0 during the second half of the clock cycle, that result can not be written into A. Thus, CLK and NOT-CLK isolates the reads and writes so that we never have the case that a value written can then be read in the same clock cycle.

edges

or arcs. Shows result produced by one instruction that is sourced by the other.

Efficiency of a multiprocessor

A multiprocessor keeps p processors on call for Tp units of time. It may not use all p every cycle, but it ties them up nonetheless. A uniprocessor ties up one processor for T1 units of time. Thus, Efficiency is the degree to which the multiprocessor EFFECTIVELY utilizes its processors. That is, T1/(pTp).

The elevator mechanism in Disk accesses

Moving the disk head to the track one wishes to access is very time consuming. Usually, there are several disk accesses awaiting service. The elevator mechanism says to move the disk head from outside track to inside track, performing the disk access as you go. For example, suppose the disk has 10 tracks, numbered 1 to 10 from the outside in. Suppose the pending requests are to tracks 3,9,6,1,5,8,2. The elevator mechanism would direct the disk to handle the accesses in the order 9,8,6,5,3,2,1, which would minimize seek time (the time to move the head to the desired track).

Embarrassingly Parallel

A term given to applications with such regular structure that it is easy to keep as many processors as one has in the system busy most of the time working on an application. The most common set of problems that have this property are scientific problems and graphics problems. For example a program that contains the statement: for i=1 to infinity, ... will take the branch backward a humongous number of times, and branch prediction becomes trivial. If the loop body of each iteration are independent of each other, then we can do as many loop bodies concurrently as we have processors available to do so. Inverting a large matrix is an example this. The more processors, the less time to do the entire job.

endianness (big-endian/little-endian)

Suppose a 32-bit value is to be stored in a memory having byte-addressability. Four memory locations are required. We store the 32-bit value in four consecutive memory locations, say those haveing addresses A, A+1, A+2, and A+3. The question still remains: which 8 bits of the value get stored in which location. If we store bits 31:24 into location A, 23:16 into A+1, etc., we say the memory system is big-endian. If we

store bits 7:0 into location A, 15:8 into A+1, etc., we say the memory system is little-endian. That is, if the number of bits required to represent a value is a multiple of the addressability of the memory, we store the value in contiguous locations of memory, starting with the low-numbered address. If we store the most significant bits at that address, we say we are storing according to the big-endian convention. If we store the least significant bits there, we say we are storing according to the little-endian convention.

EPIC (explicitly parallel instruction computer)
a modern variation of VLIW, wherein additional bits (called template bits) are provided to specify which instructions in a very long word can be operated in lock step, and which have interinstruction dependencies, and therefore, can not be.

error due to inexact
amount of error caused by rounding.

error due to underflow
amount of error caused by representing a value by 0.

exception vector
see interrupt vector

exception
an event internal to the program that is running that must be handled to properly deal with the instruction. Page fault, access control violation are two examples of exceptions.

exceptions
events that are internal to the process running, such as illegal opcode, page fault, floating point underflow, etc., that cause the processor to stop processing the current program and deal with the exceptional event. Like interrupts, it is necessary to first put the machine in a consistent state and then vector to the starting address of the exception service routine. The service routine that is executed depends on the exception which has occurred. Each exception (or set of exceptions) has its own vector to tell the machine which service routine.

excess-code (bias)
A code that uses an excess or bias to compute its code words. To obtain a code word, we add the bias to the value to be encoded. For example, with an excess-7 code, the value -1 would be expressed in eight bits (for example) as 00000110.

exponent
the bits used to describe the size of the datum. See floating point.

fall through path
The branch not taken path.

fall through problem

Fall-through Path
For conditional branches, the path that follows a branch not taken.

fault tolerance
the ability of a processor to compute correctly even though there are some faults in the hardware.

Fault vs. trap
Traps and faults are both exceptions. We stop the normal flow of execution, put the machine in a consistent state and execute the corresponding exception service routine. The difference between traps and faults is based on what to do when the exception is detected. In the case of a fault, the instruction can not be completed, so the computer returns to the state just before execution of the faulting instruction happens, and executes the exception service routine. Page faults and illegal opcode faults are examples of exceptions that are faults. In the case of traps, the instruction can be completed, so the hardware completes it and then executes the trap service machine. Trace traps and integer overflow are examples of traps.

fetch
the process of accessing an instruction from the instruction memory.

fine-grain dataflow
a dataflow graph wherein each node performs a small granularity operation, like a single multiply or a single add.

finiteness (property of algorithm)
For a procedure to be an algorithm, it must terminate so one can read the answer. The fact that it terminates means the length of time it takes to execute is finite.

fire when ready
: schedule the instruction for execution (that is, send it to a functional unit) when all data that it needs is available.

firing rule
: the rule for when a data flow node can be scheduled for execution. Safe vs. queues.

fixed length vs. variable length
: used to describe the number of bits used to encode an entity, usually used to refer to instructions, although data can also be fixed or variable length. If fixed length, the entity is always the same size. If variable length, the size is determined by the specific entity involved. LC-3b instructions are fixed length; every one consists of 16 bits. The x86 instruction is variable length. Depending on the instruction, there may be anywhere from one to 15 bytes constituting a single instruction.

Fixed point arithmetic
: the generalization of integer arithmetic. The decimal point (or binary point) is always in the same place, in the case of integer arithmetic, at the far right. No bits of a data element are wasted identifying where the binary point is.

floating point coprocessor
: A large piece of circuitry that is used to perform floating point operations. In the old days, the number of transistors on a chip was sufficiently small that to perform floating point arithmetic in hardware required a separate chip. Separate from the processor chip --> co-processor chip. By the time the 486 was introduced (late 1980s), there were enough transistors on the chip to have the floating point operations performed on the same chip and not require a separate chip. From that point on, people have more and more forsaken the name floating point coprocessor in favor of "floating point unit."

floating point exception
: there are five things that can cause an exception :underflow, overflow, inexact, divide by 0, and invalid. Exceptions can be quiet, in which case a sticky bit is set which can only be cleared by testing for it (usually used for inexact) or signalling, in which case the exception is taken immediately (always used for invalid).

floating point
: a data type supported by the ISAs of most computers. It allows larger and tinier numbers than is possible with an equal number of bits using a fixed point representation. This is accomplished by using some bits to express the size of the value (exponent bits) at the expense of the number of bits of precision. A magnitude is represented by a fractional part and an exponent part, where the fractional part contains the significant bits of precision, and the exponent part specifies the size. A value can be expressed in floating point as either a normalized number or, if it is too small to be expressed as a normalized number, it can be expressed as a “subnormal number.” Normalized numbers are of the form $\mathbf{(-1)^{\mathit{sign}} \times 1.\mathit{fraction} \times \mathit{radix}^{\mathit{exp}}}$, where the 64 bit format in IEEE arithmetic specifies the bits as follows: [63:63] is the sign bit, [62:52] is the exponent field, and [51:0] is the fraction field. If the value is too small to be represented as a normalized number, it can still be represented in the form $\mathbf{(-1)^{\mathit{sign}} \times 0.\mathit{fraction} \times \mathit{radix}^{\mathit{smallest\_exp}}}$, where [63:63] is the sign bit, [62:52] contains the smallest exponent, and [51:0] is the fraction field.

flow dependency
: often called read after write hazard. Example $\boldsymbol{a + (b \times c)}$. Before we can do the add, we must do the multiply. That is the result of the multiply must be **written** before it can be **read** as a source for the add.

flow dependency
: the dependency between a result produced by one instruction that is needed by another instruction. For example the expression A(B+C). We can not do the multiply until we do the add. We say there is a flow dependency between the add and the multiply. A popular well-meaning textbook refers to this as a RAW hazard, meaning that we can not read the source of the multiply until after we write the result of the add. I personally do not like this terminology because it totally distorts the meaning of the word hazard. We will deal with this after Spring break, so you can ignore this for now.

Flynn's bottleneck
: As long as we fetch one instruction at a time into our pipeline, we can never get an IPC at the other end of the pipeline greater than one.

Flynn's bottleneck

Michael J. Flynn's observation that as long as one fetches instructions one per cycle into the pipeline, the machine will never obtain an IPC greater than 1.

FPGA

Field Programmable Gate Array. Name given to a chip consisting of an array of interconnected logic blocks, such that the function of each logic block and the interconnection of the inputs and outputs of the blocks is programmable after the chip has been installed in a system. Programming consists of specifying and then loading SRAM cells on the chip that determine the functions of each logic block and their connectivity.

frame

the physical space that a virtual page occupies in physical memory.

free page (frame) list

An alternative to using reference bits to determine the candidate frames to be replaced on a page fault, the operating system maintains a list of frames that are available for servicing page faults.

full cross-bar switch

a set of buses, such that there are multiple transactions that can occur, one on each bus, at the same time. See my handout on interconnection networks.

Full Window Stall

in an out-of order processor, the inability to add more instructions to the reservation stations because every slot is occupied by an instruction that has been fetched, decoded and renamed, but not retired.

fully-associative cache

a cache characterized by the property that if a line of memory is stored in the cache, it can be stored in anyplace in the cache.

Functional languages

Non-procedural languages. Chacterized NOT by programs that execute step by step sequences. Characterized by single assignment. Variables can be assigned a value once and keeps that value for the duration. Thus, x=x+1 has no place in a single-assignment language, for example.

fundamental mode

asynchronous operation of a sequential machine wherein at most one input signal changes its value at a time.

g-share predictor

A variation of the GAs two-level predictor proposed by Scott McFarling, as a way to have a history encountered by different branches index into different entries of the pattern history tables. Instead of indexing based on the history in the branch history register, each bit of the BHR is exclusive-ORed with a selected bit of the address of the specific branch instruction being encountered. In that way, the index into the PHT would be different for different branches, even though the history was the same. Since different branches are updating different 2-bit counters, they would not interfere with each other, which is the objecive of g-share.

gradual underflow

the result of introducing subnormal numbers into the representation scheme so that the error due to underflow is equal to the error due to rounding as values approach 0. Subnormal numbers are numbers too small to represent as normalized numbers. Consider the smallest normalized number, we call it $\mathbf{1.000\ldots000 \times 2^{n}}$. The binade delimited by $\mathbf{2^{n}}$ and $\mathbf{2^{(n+1)}}$ contains $\mathbf{2^{k}}$ exact values, where k is the number of bits of fraction. This binade is exactly the same size as the interval between 0 and $\mathbf{2^{n}}$. The subnormal numbers are those values $\mathbf{0.}\boldsymbol{fraction} \times \mathbf{2^{n}}$. Since the fraction consists of k bits, there are exactly $\mathbf{2^{k}}$ exact values in this interval. Since this interval is the same size as the smallest binade, the error due to underflow is the same as the error due to rounding. Professor W. Kahan (the inventor of IEEE Floating Point Arithmetic) gave this notion the name “gradual underflow.”

granularity of concurrency

the scope of what is being processed simultaneously. Granularity can be at the intrainstruction level, as in pipelining, or at the interinstruction level, as in VLIW, or at the processor level, as in a multiprocessor.

Hamming Code

the mechanism for performing ECC correction. See my handout.

hammock

In a control flow graph, the result of a branch introduces two paths, the taken path and the not taken path. At some point these two paths come together and execute the same instruction. We call the point they do this "the merge point." Example: A common high level statement is IF predicate THEN instruction1 ELSE instruction2. The compiler generates ISA instructions which includes a branch and two paths, one executing instruction1, the other executing instruction2. After the code representing the IF statement is executed, the two paths merge and the instruction executed next (at the merge point) is the instruction resulting from compiling the statement after the IF statement. The branch instruction, instruction1, instruction2, and the fact that the processing then merges is referred to as a hammock.

handshaking

the interaction that starts off each bus cycle that is required since there is no clock.

hardware interlock

a hardware signal that prevents improper behavior from occurring in the execution of instructions. For example, consider the pipelined processing of a MUL that stores to R3, immediately followed by an ADD that sources R3. At the time the ADD wishes to source R3, the value stored there is the old value, not the value that will be produced in a few cycles by the MUL instruction. A hardware interlock (called a scoreboard bit) prevents the ADD from sourcing R3 until after the MUL stores its result there.

Hashing functions

If we have 2^k addresses, we can index into a storage structure of size 2^k very quickly. But, what if only a small fraction of those 2^k addresses are relevant. Most agree that it would be a great waste of storage to allocate space for 2^k locations in this situation. Example: one billion addresses, but only 100 relevant addresses. A simple technique that allows us to keep the notion of indexing, but does not unnecessarily waste so much storage is to transform each 30 bit address into a 7 bit index (since 2^7 > 100). We call that transformation a hash function, and the process: hashing. The hash function can be something as simple as selecting 7 of the 30 address bits. We would store the 30 bits in each entry, so the 7 bit hash function would index into the 100 element storage, and the contents compared with the full 30 bits to determine if one is accessing the correct location. A problem occurs when two different 30 bit addresses hash to the same 7 bit index. We call this a collision. There are several ways to resolve a collision, all beyond what we want to do in 460N.

high availability

a more recent term for fault tolerance. Availability refers to the fraction of time a computer is processing correctly, and not “down.” A high availability machine is one that is “down” for no more than a very few minutes a year. To accomplish such, the computer system must be able to operate in the presence of faults; therefore, the term: fault tolerant.

history register

See above.

hit ratio

the percentage of accesses that are hits. That is

```
cache hit ratio = hits/(hits + misses)
```

Horizontal Microcode

Microcode in which the fields in a microinstruction are mostly independent. Microinstructions are generally made up of many more bits than is the case for vertical microinstructions. The independence of the fields generally allows for a richer set of control options for an individual microinstruction, plus the ability to specify more than one concurrent operation in a single microinstruction. This generally results in far fewer microinstructions to implement a specific task than would be required for vertical microcode.

horizontal priority

priority determined by proximity to the PAU, among all controllers having the same vertical priority.

hypercube (cosmic cube)

An example of an interconnection network. The structure is a boolean hypercube. That is if n is the number of dimensions, and each dimension has two values (0 and 1), there are $\mathbf{2^n}$ connection points in the network. Each connection point is connected to n other connection points. Each connection point is at most n hops from every other connection point. See my handout on interconnection networks.

I Buffer

An internal pipeline register at the output of the fetch phase of an instruction, which holds the instructions fetched from the Instruction memory.

I/O processor
a special purpose processor that manages I/O activity, including doing some processing on that information which is being transferred.

immediate/literal
An operand that is available in the instruction itself. That is, an operand that does not have to be fetched from memory or register.

in-flight instructions
The set of instructions that have been fetched but not yet retired.

in-order execution
a microarchitecture that executes instructions in program order.

in-order retirement
instructions complete their execution (as far as the software knows) in program order – that is, the same order that the program dictates.

in-order retirement
retirement of instructions occur in the order of their location in the program. Necessary to obey the sequent programming model.

inclusion property
In a shared memory multiprocessor, it is important to have cache coherency across all the caches. That could mean each processor's caches monitoring the traffic associated with all the other processors' caches. It would degrade performance to have the L1 cache involved in this monitoring activity. One solution is to obey the inclusion property, which states that any location present in a processor's L1 cache is also present in that processor's L2 cache. In such cases, the L2 caches can do the monitoring for cache coherency, freeing the L1 caches from that task.

indirect addressing
An addressing mode, where the address (A) of the source data is contained in a memory location (B), the address of B is computed by the addressing mode. We say that B is a pointer variable because its contents is not data but rather the address A of the data.

Inexact Numbers
a number that can not be represented exactly in the data type being used. For example, with 2's complement integers, the number 2.34 can not be represented exactly. With 64-bit floating point, the number 3.6 can not be represented exactly.

Instruction Cache
A cache that holds only cache lines from the instruction stream.

Instruction Cycle
The time it takes to process one instruction, from fetching the instruction to storing the result.

instruction level simulator
A simulator that changes its state at the granularity of the instruction. That is, given the machine state and an instruction, the instruction level simulator produces the state required by the instruction. No within-instruction states are simulated.

Instruction prefixes of x86
The x86 has a variable length instruction, from 1 to 16 bytes in all. The opcode is usually the first byte, except the ISA has introduced several one-byte prefixes to alter how the instruction will be performed which when used, precede the opcode. For example, the Repeat prefix requires the instruction to be executed many times, depending on the contents of the ECX register. There is a prefix that alters the default uses of the segment registers. There is a prefix that changes the data size from 32 bits to 16 bits if the data being operated on is 16 bit data. This is part of what makes decoding x86 instructions such a pain.

Intel variable page sizes
Intel's 48 bits of relevant address is partitioned into fields of 9-9-9-9-12, giving us multiple levels of indirection. CR3 contains the starting address of the main directory, which is a 4K page of 8-byte descriptors. Bits 47:39 of the address is a 9 bit offset into that Page Directory. The descriptor points to the base address of a second directory, and bits 38:30 index into that directory. ...etc. until bits 20:12 provide the PTE of the 4K page. At each step along the way, if we prefer, the descriptor would not point you to

another directory but rather to a larger size page. i.e., 12+9 bits give you a 2MB page, 12+9+9 give you a 1GB page.

Intelligent I/O Device
An I/O device controller that can operate independent of the processor, once the processor has specified the actions for the I/O controller to carry out. For example, a DMA access requires the I/O controller to carry out the transfer without being instructed by the processor on an instruction by instruction basis.

interconnection network
a structure that interconnects various elements (usually processors and units of memory) of a computer system.

Interference in branch predictor (positive and negative)
the prediction of one branch based on history information acquired for other branches. If the behavior of the two branches are correlated, then the prediction generally succeeds. The interference is positive. If the behavior of the two branches are not correlated, the prediction is usually erroneous. The interference is said to be negative.

interleaving
allocating addresses to memory locations "horizontally," rather than "vertically." That is, if address n is in bank 0, address n+k is in bank 1, address n+2k in bank 2, address n+3k in bank 3, etc. where k is the size of the access in a bank in bytes (assuming byte-addressable memory).

interrupt driven I/O
I/O activity controlled by the device requesting the processor interrupt what it is doing.

interrupt vector
the bit pattern identifying the particular interrupt, which is used to initiate the correct service routine.

interrupt/exception priority level
priority at which an event requests interruption of the running program. Usually, the priority at which the service routine runs. Priority depends on urgency of the event.

interrupt
an external event more urgent than the program running that causes the machine to stop execution of the program that is running and turn its attention to service the external event. Keyboard interrupt is an example of a low priority interrupt. Machine check is an example of a high priority interrupt.

interrupts
external events that cause the processor to stop processing the current process, and after putting the machine in a consistent state, vector to the starting address of the service routine specified by the specific vector.

invalid exception
an exception caused by an operation that produces nonsense results. Examples include 0/0, infinity/infinity, sqrt(-*n*), arcsin(2).

IPC (instructions per cycle)
a common metric of performance. The number of instructions completing execution on average per cycle.

IRD
An acronym: Instruction Register Decode. In microarchitecture parlance, it designates that in this cycle instruction decode is being performed.

Irregular Parallelsim
To differentiate it from regular parallelism, regular parallelism is when the code can be executed in a very orderly parallel way which is obvious when looking at the algorithm. Simplest example I know of is taking the inner product of two vectors: One multiplies a[i] times b[i] for all i. Very "regular". Irregular parallelism is the case where, if you draw the data flow graph of the algorithm, it is not obvious where the parallelism is. I appreciate that this definition is not precise, reflecting the notion that the concept is not precise.

ISA
Instruction Set Architecture. The specification of all aspects of the interface between the software and the hardware; for example, instruction formats, opcodes, data types, addressing modes, memory address space, memory addressibility, etc. This allows the software to specify a sequence of operations for the hardware to carry out (the program) and for the hardware to know what each of those instructions is asking it (the hardware) to do. For example: x86, Power-PC, SPARC, MIPS, VAX, Alpha are all ISAs.

issue width

The maximum number of instructions that can be fetched and decoded each cycle. In the old days that number was 1. Today, for the most part it is four.

kernel, executive, supervisor, user privileges

If an ISA is to support multiple processes in the balance set, managed under the control of an operating system, it is normally the case that processes run with different levels of privileges. Many ISAs specify two levels of privilege: privileged (usually referred to as Kernel mode) and unprivileged (usually referred to as User Mode). Some ISAs have been known to identify a finer granularity of privilege, which I have tongue in cheek referred to as unpriviledged, privileged, really privileged, and REALLY privileged. The VAX ISA is an example of an ISA with four privileged levels, identified as user, supervisor, executive, and kernel. More levels of privilege gives the operating system the ability to differentiate different processes with different degrees of access rights

key transformation

a fancy synonym for hashing.

label

a symbol that identifies the location of an instruction. Necessary if that location contains the targe of a branch or a location that contains data that will be accessed by a load or a store instruction.

last time predictor

predict whatever the branch did last time it will do this time.

latency

how long it takes something to happen. memory latency -- the time it takes from when the address is provided until the information stored at that address is available. We will talk about interrupt latency after the spring break -- the time it takes from the time some external device requests service until the time that device starts getting serviced.

latency

one of the three common metrics for goodness of an interconnection network. Latency is a measure of the number of hops through the network a transfer has to make from source to destination.

length register

A register that is part of the virtual memory management structure, identifying how many pages are described by the corresponding page table. This is needed to protect against false translations wherein the page number computed from the virtual address may correspond to a page that is not part of the virtual address space that is mapped. For example, if an executable image consists of 6 pages, the page table will contain 6 PTEs, and the length register will contain the value 5 (pages 0, 1, ...5). Any virtual address whose bits that identify a page greater than 5 must therefore be a bad address. This is easy to detect by simply comparing the page- number bits to the contents of the length register.

levels of transformation

my term for the transformations that must take place in order to get a problem stated in some natural language like English, Spanish, Gujarati or Tagalog to be solved by the electrons that do the work. Problem --> algorithm --> high level or assembly language program --> object code in the ISA of the machine --> control signals that drive the microarchitecture --> logic --> circuits --> electrons.

limit register

the number of PTEs in the page table. Used to determine if a virtual address actually corresponds to an address in the virtual image.

linear address

Intel's name for virtual address. That is, a segment register provides a base address which is combined with the address provided in the program to form the linear address. The linear address is further translated to a physical address using PTEs.

live-in

a value that exists prior to the entity under consideration. With respect to a unit of code, a live-in is a value that is a source for one of the instructions in the unit of code.

live-out

a value that is produced in the unit of code and is required as a source by code subsequent to the unit of code.

Load/Store architecture

The term refers to ISAs that require separate load and store instructions to move data to and from memory. That is a load/store ISA does not allow operate instructions to operate on data loaded from memory in the same instruction. Nor, does it allow the result of an operate instruction to be written to memory in the same instruction. So-called RISC ISAs were in general Load/Store. So-called CISC ISAs were in general not.

load/store ISA

An ISA, where operate instructions can only operate on operands that are stored in the registers. That is, one can not load a value from memory and operate on that value in the SAME instruction, for example. The LC-3b is an example of a load/store ISA. The x86 is not.

load/store pipe

lock step

a characteristic of a VLIW machine is that the *n* instructions are executed by *n* processing elements in lock step, that is, at the same time. Further, if one processor finishes before the others, it can not fetch its next instructions until all have finished and the next fetch is done at the same time for all n processing elements.

logically complete

a set of logical primitives that allows one to specify all logical functions. NAND, for example is by itself logically complete. AND is not by itself logically complete. The set {AND, NOT}, however, is logically complete.

long integer

integers of size that is a multiple of the word length. Not supported by the ISA. However, software can be written to deal with long integers. Example in class: with a word length of 32 bits, the x86 can operate on 320 bit integers by means of a procedure that processes the values 32 bits at a time. Takes a loop that will iterate 10 times.

look ahead carry

A logic design mechanism that recognizes the recurrence relation in the process of addition, and uses that relation to compute the sum of two source operands without the need to wait for the carry signal to ripple through all the bits of the source operands. We will discuss this further when we talk about arithmetic after the break.

Loop buffers

in the Cray machines, an instruction buffer containing enough storage to accommodate an entire loop.

loop unrolling (software pipelining)

The name given to the process of executing multiple loop interations before encountering a branch. For example, a loop that iterates ten times could be unrolled once, such that each iteration now consists of what was before two iterations. That is, the code would do iteration one, iteration two, and then test to see whether to branch back. The technique is used in those cases where results produced in one iteration are needed by the next iteration, wherein if the loop is enrolled the whole second iteration does not have to wait for the completion of the whole first iteration.

loosely coupled MP

see multiprocessor.

machine check

the highest priority interrupt, indicating that the processor is generating bogus information. Cause: a hardware malfunction.

mantissa (aka significand or fraction)

the part of the representation that contains the significant bits of the value.

mapping virtual to physical addresses

the transformation of a virtual address to the physical location in memory where the information at that virtual address is actually stored.

maskable

the ability to ignore an event so as to defer handling, or even deny handling altogether.

medium

the material containing the information – the hole in a punched card, the magnetic material on the disk, etc.

memory bank

a unit of memory that can process one memory access at a time.

memory contention

the result of more than one processor attempting to access the same memory bank at the same time.

memory mapped IO
the characteristic of an ISA wherein there are no separate input and output instructions, but rather all I/O devices are assigned addresses and input and output is accomplished by Loads (input) and Stores (output) to those addresses

memory order buffer (MOB)
The structure that keeps track of the memory accesses so as to be able to quickly tell if there is a disambiguation problem.

Memory Rank
A set of memory chips whose addresses all agree in the high bit positions. For example, consider one GB of physical memory, made up of 2MB chips. Assume it is 16 way interleaved, and accessible via a 64 bit bus. The 30 address bits address the memory chips, as follows: Bits[2:0] don't address the memory chips. Bits[6:3] determine the bank of memory. They can enable the tri-state drivers. Bits[27:7] identify the address of each individual chip. That leaves bits[29:28] to Chip Enable the appropriate "row" of memory chips. In this case there are 4 such rows. We call the "row" in this context the "rank" of the memory. Again, all locations in the same rank have the same value for address bits [29:28].

memory-mapped I/O
I/O activity can be specified by special instructions in the ISA or by using the load and store instructions, but assigning each I/O device register an address from the memory address space. The latter is referred to as memory-mapped I/O. For example, with memory-mapped I/O, keyboard input is specified by a load instruction where the address is the one assigned to the keyboard data register.

merge point

Merging nodes
In HPS, each instruction is broken into multiple micro-ops. The producer/consumer relationship of these micro-ops can be described as a data flow graph, with the nodes as micro-ops and the arcs as the data flowing from producer node to consumer node. All instructions in flight are considered as a combined single data flow graph, where the connections between nodes of the various instructions are those resulting from nodes producing results that are needed by subsequent nodes. After an instruction is decoded into its data flow graph, that data flow graph is merged into the combined single data flow graph of all instructions in flight. This is done by constructing arcs from nodes in the combined data flow graph that produces results that are consumed by nodes in the data flow graph of the instruction just decoded.

mesh
a connection network that forms a two dimensional array. See my handout on interconnection networks.

message passing
see multiprocessor.

microarchitecture
a particular data path and its control that carries out the specification of an ISA. For example, the x86 ISA has been implemented by Intel by the 8086, 386, Pentium, Pentium-Pro, Pentium 4, Pentium M, etc. and by AMD by the Athlon, Opteron, etc.

Microcode vs Micro-op
Microcode is a generic term to describe all control signals at the microarchitecture level. A micro-op is usually a single control signal. Sometimes called micro-command. A microinstruction is the set of micro-ops that are active during a clock cycle.

microcommand
An individual encoding of a field in a microinstruction. For example, LD.REG is a microcommand that either loads or does not load a register in the register file. ALUK/ADD is a two bit microcommand that instructions the ALU to perform addition this cycle.

microinstruction
the set of control signals needed in one clock cycle. In a microarchitecture with a control store: the contents of one location in the control store.

microsequencer
the mechanism in the microarchitecture that figures out what control signals will be needed to process during the next clock cycle.

MIMD

multiple instruction streams each operating on its own data set. The classic form of a multiprocessor or multicomputer.

mirroring

aka RAID-1. Two disks to store the information of one. Each bit has its mirror bit.

modify bit

a bit in the PTE which is set when a page is written to. If the Modify bit is clear, the page need not be written back to disk on replacement since it has not changed since being read from the disk.

modulus

In a modulo scheme of any sort, the number of distinct values. In practical matters, if I say "*A* modulo *p*," I am asking what is the remainder when I divide the value *A* by the modulus *p*. Example: **10 modulus 3 = 1**.

mops/rops

mops = micro-ops, reflecting the fact that most ISA instructions require multiple microoperations in order to carry out the work of the ISA instruction. For example, a load register + offset requires at least an address arithmetic mop, and a memory access mop. Due to the fact that these mops resemble in many ways the instructions of a class of ISAs known as RISC processors, AMD dubbed the mop rop, or RISC op.

MSYN

signal used by the bus master in synchronizing the transaction.

MTTR (mean time to repair)

see TMR

multi-dimensional array

a classical mathematical structure, having multiple dimensions.

multicore

More than one core on a chip.

multiple levels of caches

A vast disparity exists between the fast access needed by the cache that is dealing with the processor (say, one or two cycles) and the very slow access that is available from memory (say, hundreds of cycles). It is also the case that in order for the cache to be accessible in one or two cycles, it must necessarily be small. Caches are either fast or large, they can not be both. This model would result in too many misses, resulting in too many long accesses to memory. The solution is to have multiple levels of cache: L1 is small, but takes one or two cycles, L2 might take 6 or 8 cycles but can be much, much larger. Sometimes there is an L3 cache, which is slower still and larger still. Finally, there is memory -- largest, but slowest. The notion is to extend the notion of the memory hierarchy to within the levels of cache.

Multiplexed Address and Data Bus

A bus that uses the same wires to transmit the address, and then subsequently the data.

multiplexed bus

a bus that is used to store address bits at some times and data at other times.

multiprocessor

more than one processor operating concurrently. The pure definition of a multiprocessor requires that each processor carry on substantial work by itself (to differentiate it from a co-processor), and that all the processors share the same memory space. Sometimes this form of multiprocessor is referred to as a tightly-coupled multiprocessor …as compared to a loosely-coupled multiprocessor where there is no shared memory. The more proper term for a loosely-coupled multiprocessor is multicomputer network. But most people refer to the two paradigms as tightly and loosely coupled multiprocessors. In a shared memory multiprocessor, values can be passed from one to the other via shared memory locations. In a loosely-coupled multiprocessor, they are passed via message passing.

multithreading

a processor that allows multiple threads to be in various stages of execution at the same time. A thread is another word for an instruction stream. The HEP (circa 1978) is the first machine to implement multithreading. Each cycle an instruction is fetched from a different thread in a round-robin fashion. See my handout on the HEP.

NaN (not a number)

the result of an operation that creates nonsense. see invalid exception.

nanocode

The result of two-level microprogramming, where a very wide horizontal microinstruction would be replicated a sufficiently large number of times that it is more economical to specify a sequence of smaller length microinstructions which point to the much wider ones. Thus n instances of the wide microinstruction is replaced by a single instance of the wide microinstruction and n instances of a much narrow microinstruction. The wide microinstructions have been dubbed nanocode.

natural word length

the generic size of elements processed in the machine. Generally, the size of the registers. Generally, the size of values processed by the ALU in a single cycle.

NMI(Non Maskable Interrupt)

Corresponding to every interrupt level is a mask bit that can mask the interrupt so that code can run with that interrupt turned off. A non-maskable interrupt is one that we always want it to interrupt the processor if it is present. Typically, the only two interrupts that are non-maskable are machine check and power failure.

nodes

instructions

Node table

My term for the reservation stations. In an out-of-order execution machine, we need a place to store micro-ops waiting for their dependencies to be resolved so they (the micro-ops) can be executed. Tomasulo called this storage "reservation stations" and almost everyone agrees with this term. I think of these micro-ops as nodes in a data flow graph so I want to store nodes in a "node table." My term for it. Since no one seems to agree with my choice of a term, you can conveniently forget it and call the storage locations reservation stations.

non-deterministic

The opposite of deterministic. A deterministic process states that given an input and a current state, the output and subsequent state are explicitly known. A nondeterministic process allows for more than one possibility for output and subsequent state, given a specific current state and input. That is, given a current state and current input, you may go to one next state or another and there is no way to know which.

normalize

to put into a standard (i.e., “normal”) form. In the case of IEEE Floating Point arithmetic, that form is $\mathbf{1}.\mathit{fraction} \times \mathbf{2}^{\mathit{exponent}}$.

Northbridge, Southbridge

name given to the buses that connect the processor chip to memory (northbridge) and I/O peripherals (southbridge)

.

NUMA (non-uniform memory access)

a distributed shared memory multiprocessor, wherein processors can access all of memory, but the access time varies depending on which part of the distributed memory a processor is accessing.

Omega network

an interconnection network that is a "half-way house" in some sense between a bus which has maximum contention and minimum cost and a full crossbar which has minimum contention and maximum cost. The omega network consists of an array of k by k crossbar switches. cost is proportional to the number of switch connections. Ergo, a k by k crossbar has a cost = k2. To implement an n by n omega network with k by k crossbars, the array has n/k rows of log-base-k(n) columns. Each column decodes log-base-2(k) bits of the address, and forwards the access request to the next column of k by k crossbars.

one level/two level microcode

Two level microcode is the nanocode described above. One level is the absence of two level!

opcode

the part of an instruction that specifies the operation(s) that the hardware is expected to perform in carrying out this instruction.

operand

An argument of the opcode to be executed (source operand) or the result of the execution (destination operand)

operand specifier

A code (part of the instruction) that specifies the addressing mode and a register to use in the address computation to identify the location of an operand. For example, in the PDP-11, an operand specifier of 010101 specifies 010 as the addressing mode, and 101 as R5, the register to use in the computation. The addressing mode 010 in the PDP-11 ISA instructs the computer to use the contents of the register specified (in this case R5) as a pointer to the memory location containing the data, afterwhich to increment the register by the size of the data.

operate instructions

instructions that operate on data: ADD,SUB,XOR,SHR, etc.

out-of-order execution

a microarchitecture that executes instructions when they are ready for execution, regardless of their order in the program.

output dependency

values output from the same register must be output in program order.

overflow

an exception caused by a computed value being too big to be expressed by any normalized finite value. Rounding will result in the value being represented by infinity or by the largest number to be represented exactly, depending on the rounding mode used.

Overlay

With n bits, one can uniquely identify 2^n items. Suppose one wants to uniquely identify more than 2^n things. One could resort to overlays, as follows: One could identify 2^n items whose overlay vaule had a value of 0, an additional 2^n items whose overlay is 1, a third 2^n items whose overlay value is 2, etc. That is, if the overlay register can have k distinct values, then with overlays, one can identify k * 2^n items with n bits.

page fault (TNV)

Translation not valid. the virtual address being translated is not resident in physical memory.

page in/page out

the act of moving a virtual page from the disk to physical memory (page in), or from physical memory to the disk (page out).

page mode

Typically DRAM chips are addressed in two steps. Consider the DRAM chip as containing 2^n addresses. The high order n/2 bits (Row) come in first, using RAS. This effectively preselects the contents of all addresses having the same row address bits. Then the low order n/2 bits (column) of the address comes in, using CAS. This selects the exact address from the 2^(n/2) that have identical row address bits. Access time is the sum of RAS + CAS times. If the next address has the same row address bits as the previous one, the memory controller, which keeps track of this, only needs to send the column address bits to the DRAM chip, thereby saving the time needed to latch the row address bits. We say that all addresses having the same row address bits are on the same "page," and the act of the memory controller exploiting this by only sending subsequent column address bits, we refer to as "page mode."

page table base register

the starting address of the page table. Used to compute the address of a PTE.

page table

a data structure containing the PTEs of all pages in the virtual image of the process.

page

the unit of virtual memory that is transferred between disk and physical memory. That is, the whole page is brought in or out, or none of it.

paging

the process the o/s virtual memory management system goes through in bringing a page from the disk into a frame of physical memory (and perhaps sending the page that previously occupied that frame back to the disk).

pair and spare

another scheme for maintaining error-free operation in the presence of single processor failures. Two pairs of processors, all four executing. Each pair compares results with its partner. As long as all four give the same answer, we are fine. When the partners of one pair disagree, we enter vulnerable mode, where we use

the other pair for computation while we fix the failing pair. Again, MTTF and MTTR tell us how good our scheme is.

pairwise relatively prime

A number of values are pairwise relatively prime if the greatest common diviser of every pair equals 1. For example, 7,8, and 9 are relatively prime since the gcd of 7,8 is 1, gcd of 7,9 is 1, and gcd of 8,9 is 1. Note that 8 and 9 are not prime.

parallelizable

see Amdahl's Law. That part that can be executed in parallel.

Parallelizibility

the fraction of the problem that can be assigned to as many processors as are available, such that all processors are active. If the number of processors is p, then that part of the problem can be executed in 1/p of the time it would have taken with a single processor. Also called vectorizability.

parity

a coding scheme that requires only one extra bit to detect a single bit error, provided that all single bit errors are statistically independent. "Even" parity dictates that this extra bit must be set to 1 or 0 such that the entire n+1 bit transfer contains exactly an even number of 1s. In that way, if the received n+1 bits contain a single error, the number of 1s will be odd, so the fact that an error has occurred will be detected. Statistical independence of the bit errors is necessary, since if two bits are in error, parity will not detect it.

pattern history table

The table of 2-bit counters. If the history register contains n bits, the table contains 2^n 2-bit counters.

pending bus

a bus with the property that once the bus is grabbed for a transaction, the bus is tied up (often twiddling its thumbs) until the transaction has completed.

Pentium / RISC frequency

Pentium was designed to get as much work done per cycle as possible. Ergo, 66 MHz frequency. The Alpha 21064 (first Alpha chip and very much a poster child for RISC) made its design lean, opting for the highest frequency possible. Ergo, 200 MHz. Recall the performance equation that Performance is inversely proportional to cycle time. So, Alpha 21064 wins, 3 to 1 over Pentium. But there are other figures of merit, CPI and L. Performance is also inversely proportional to them. Pentium wins big on L (the number of dynamic instructions executed in the program). So, it is not just frequency that matters but CPI and L as well. Still frequency is the dominant one of the three figures of merit. ...which is why Intel's next chip (Pentium Pro) opted for a 12 stage pipeline. By dividing the work into more pieces, it could do each piece more quickly, resulting in a smaller cycle time, which meant a higher frequency.

performance metric

The basic equation for performance is :

$$\mathbf{1 \div (}\boldsymbol{N \times CPI \times t}\mathbf{)}$$

N is the number of instructions in the program being measured. *CPI* is the average number of cycles it takes to complete (retire) each instruction. *t* is the cycle time of the clock. Note that if make the clock smaller, I probably increase *CPI*. *CPI* vs. *t* is a fundamental tradeoff. Note also that *t* is 1/*freq*, and *CPI* is 1/*IPC*. Most machines quote their *IPC* values (bigger is better), although the microarchitect is more interested in *CPI*, since the microarchitect is interested in eliminating some of the cycles that contribute the *CPI* of an instruction. Recall, when we talked about virtual memory, I mentioned that the VAX 11-780 had a *CPI* of 10.5, that each VAX instruction on average made one memory reference, and that the page table translation mechanism took 22 cycles. The reason the *CPI* is only 10.5 is because there is a TLB on that machine which takes no extra time when it hits. Since it hits 95% of the time, only 5% of the time does the 22 cycle translation take place. 5% of 22 cycles is approximately one cycle. That is, approximately one cycle of the 10.5 *CPI* is due to virtual to physical memory translation.

performance metric

The basic equation for performance is :

$$\mathbf{1 \div (}\boldsymbol{N \times CPI \times t}\mathbf{)}$$

N is the number of instructions in the program being measured. *CPI* is the average number of cycles it takes to complete (retire) each instruction. *t* is the cycle time of the clock. Note that if make the clock smaller, I probably increase *CPI*. *CPI* vs. *t* is a fundamental tradeoff. Note also that *t* is 1/*freq*, and *CPI* is 1/*IPC*. Most machines quote their *IPC* values (bigger is better), although the microarchitect is more interested in *CPI*, since the microarchitect is interested in eliminating some of the cycles that contribute the *CPI* of an instruction. Recall, when we talked about virtual memory, I mentioned that the VAX 11-780 had a *CPI* of 10.5, that each VAX instruction on average made one memory reference, and that the page table translation mechanism took 22 cycles. The reason the *CPI* is only 10.5 is because there is a TLB on that machine which takes no extra time when it hits. Since it hits 95% of the time, only 5% of the time does the 22 cycle translation take place. 5% of 22 cycles is approximately one cycle. That is, approximately one cycle of the 10.5 *CPI* is due to virtual to physical memory translation.

PFN

page frame number. The high bits of a physical address; it specifies the frame number within physcial memory.

physical memory

memory that actually exists.

physically-indexed, physically-tagged cache

a physically addressed cache.

pipeline bubble

a hole in pipeline, caused by the inability of instructions to flow through the pipeline. For example, following a conditional branch, it may not be possible to know which instruction to fetch next. This causes a a pipeline bubble. …or a load followed by the use of that load can cause a pipeline bubble.

pipeline stages

the set of steps each instruction goes through in the pipeline.

pipelined bus

A special case of a split transacton bus, wherein if there are more than one transaction in progress, the sequence of the second part of the transactions is the same as the sequence of the first part. That is, the order of the bus transactions is maintained. (See also, split transaction bus.)

pipelining

the assembly lining of instruction processing. Each instruction goes through multiple steps (stages), from Fetch to Retirement/completion of the instruction.

pipelining

the process of carrying out instructions in an assembly line fashion. Suppose instructions are fetched one per cycle. Then in a pipelined implementation, in one cycle the fetch logic fetches an instruction, and hands it off to the decode unit to decode in the next cycle. In that next cycle, the fetch logic will be fetching the next instruction.

PLA

Programming Logic Array. A combinational structure consisting of an array of AND gates all producing outputs which are fed to an array of OR gates. The general structure of a PLA sources all external inputs to each AND gate, making each AND gate able to identify one input combination (row of a truth table). Also, in the general structure, all AND gate outputs are available as input to every OR gate. Those actually connected as inputs to an OR gate are those that produce a 1 in the output column of the truth table of the relevant logic function. Thus, in principal, any arbitrary logic function of the external inputs can be implemented as the output of one of the OR gates. In practical structures, the fan-in (maximum number of inputs) to an OR gate is fixed, thereby limiting the ability of the PLA to implement all combinational logic functions.

platter

colloquial word to describe one of the surfaces of a disk.

polling

I/O activity controlled by the processor executing its program.

post increment (auto increment)

This is an addressing mode that is common on many older machines and a few more recent machines. The address of the memory location containing the operand is in a register. After the address is read, the register is incremented, so the next time that instruction is executed, the register will point to the next data

element. Side note: since data usually occupies some number of bytes, the register is incremented by the number of bytes of the operand.

power failure

loss of power causing a very high priority interrupt.

pragma

A pragma is a non-executable augmentation to a high-level language program that provides useful information to the compiler, similar to the pseudo-ops we have encountered in the LC-3b assembly language.

precise exceptions

if an instruction causes an exception, it is often important to recover the machine state that corresponds to the situation where all instructions preceding the exception-causing instruction have executed correctly and no instruction following the exception-causing instruction has executed. It is also useful to identify the instruction that caused the exception. A microarchitecture that can do both is said to support precise exceptions. The Tomasulo Algorithm running on the IBM 360/91 did not support precise exceptions.

precision

the number of bits of significance in the representation of a value. For 52 bits of fraction, the value has 53 bits of precision.

Predicated execution

execution of an instruction that is conditioned on some predicate. The normal form of a predicated instruction is op Ri,Rj,Rk,p where p is a predicate (true or false). Ri is loaded with the result of op Rj,Rk if p is true. If p is false, the instruction is treated as a no-op.

Prediction of technology - alpha 21164 example

Also in the introduction, I told you that one of the jobs of the architect is to predict where technology will be when your design is ready to be turned into silicon. ...and if you guess too agreesively (i.e., incorrectly), you may end up with not being able to implement your design. The Alpha 21164 was an example of too aggressive a prediction, since the architect thought he would have enough transistors for a 128KB cache.

prefetching

Fetching instructions or data early so that they are available quickly when they are needed. For example, most microprocessors have units of storage called L1 caches that can be accessed in one or two cycles. It is useful during instruction fetch or data fetch to be able to access this element in one or two cycles. This can be accomplished by obtaining the element from a much slower unit of storage before it is really needed. We call that access in advance of when needed *prefetching.*

prefix property

the property that a bit pattern is a prefix of a longer bit pattern. For example, 1001 is a prefix of 1001000001111.

priority arbitration unit (PAU)

the central arbiter.

priority/privilege

Priority is about urgency, privilege is about the right to do something. They are orthogonal issues. We sometimes confuse them in our human experiences because sometimes they go hand-in-hand. Example, the fire truck runs a red light to get to the fire. Running the red light is about privilege. We give it that privilege because of the urgency of getting to the fire. There are examples where they do not go hand-in-hand. Recall my example in class of the mail clerk who had to go to the bath room while waiting for the elevator on the 55th floor. In computer technology, they really are orthogonal. Real-time experiments may run at higher priority, because they need to sample the sensors at a precise moment in time. But they can not execute RTI or turn off the computer, or access certain files. System administrators can do that, even though they may run at a low priority level because it is not urgent when they do their job, just that they do it.

privilege level

every ISA defines a set of privilege levels, from unprivileged to most privileged to distinguish what level of privilege is required to make certain accesses. The simplest model has two levels of privileged: privileged and unprivileged.

Process Control Block

A data structure that identifies all the state information about a single process. This data structure is stored by the operating sytem, and loaded when a process gets control of the processor, and saved when a process loses control of the processor - "monotonically non-decreasing level of access" "monotonically" means always in the same directcion. "non-decreasing" means never getting smaller. Ergo, monitonically nondecreasing mean "increasing or staying the same." In the context of the usage in class, as privilege goes from lowest privilege level to highest privilege level, the allowed accesses never get lessened.

producer

the instruction producing the result that is sent to various consumers.

profiling

the act, usually as part of the compilation process, of executing a program on a sample input data set to get a feel for the behavior of branches and other run-time activities. The intent is to have the compiler generate a better object file as a result of this information. Whether profiling provides value or not is highly dependent on how close the sample input data set resembles the actual input data set when the program is executed for real. We call this "closeness" representativeness.

Program Order

the order of instructions in a program as specified by the programmer or by the compiler.

program phases

Programs executing usually exhibit phase behavior, which is to say the algorithmfunctions as a sequence of phases. A good real life example is the game of chess which has a beginning game, a middle game, and an end game. Close to computing is sorting. Quicksort has a major phase which is executed recursively while the number of elements is large. When the number of elements becomes small, the algorithm moves to a different sorting algorithm. The point of noting phase behavior is that at runtime, we are able to use history information to predict certain things, like branches, for example. But when the phase changes, all the history is meaningless and we need to collect new information. The good news, of course, is that we can collect new information, because we are operating at run time. Phases, however, are a problem for compile time activity since it is very difficult for the compiler to adequately accumulate effects and differentiate things done in one phase or another. Simple example: in phase 1, a branch is taken 10 times, not taken 0 times. In phase 2, a branch is taken 0 times, not taken 8 times. What should a compile time scheduler predict? A run time predictor can keep track of the phases and get it right most of the time.

Propagation Delays

A propagation delay of a structure is the time it takes the output of a structure to reflect a change in values at the inputs of that structure. The term can be applied to an individual gate. The term can also be applied to the collection of gates that operate during a clock cycle. In fact, logic design involves keeping track of the propagation delay of each gate in order to ensure that the propagation delay of the entire circuit is less than the clock cycle time.

protection

the concept of protecting a computer resource from being utilized by a process that does not have the desired level of privilege. For example, a page can be protected so that only the highest privileged processes can write to that page. Another page may be protected so that any process can write to it. Some instructions can only be executed by processes that have a certain level of privilege. For example, on machines that have a HALT instruction, unprivileged processes are generally not allowed to execute the HALT instruction ...and allowed to remotely stop the computer.

protocol

Taken straight from our embassies. The sequence of actions that must take place in order to transact business. In the case of an asynchronous bus, business is the transmission of a value from A to B. Also called handshaking.

Pseudo-LRU replacement policy

one of the cache replacement policies that approximates replacing that cache line from the set that was least recently accessed. We discussed in class two such policies for 4-way set associative caches: (1) the victim, next victim policy, and (2) the three-bit per set policy that combines an LRU bit for way 1 and way 2, an LRU bit for way 3 and way 4, and a third LRU bit for deciding between the other two LRU bits.

pseudo-op

a directive provided by the assembly language programmer to the assembler to help the assembler translate the assembly language program to the 0s and 1s of the machine's ISA. In LC-3b assembly language,

.ORIG, .FILL, and .STRINGZ are examples of pseudo-ops. Pseudo-ops are not executable at run-time. They are strictly messages to help the assembler assemble the program.

PSR

processor status register. A processor register that contains status information about the running process, including its privilege level and priority.

PTE

page table entry. A descriptor for each page of virtual memory, containing information, including the frame of physical memory where it resides, whether it has been modified since it has been loaded from the disk, and protection information as to what levels of privilege are required to access all information stored on that page. Some ISAs include other information in each page's PTE.

queue dataflow

wherein a node fires when all inputs have tokens. The outputs produced are queued, waiting for their subsequent use.

race condition

A condition where the result depends on the inadvertent propagation delay of signals from one device to another.

radix

the base, as in $\mathbf{2}^{\boldsymbol{exponent}}$ ---> 2 is the radix.

RAID-0

striped disk array with no redundancy.

RAID-1

see mirroring

RAID-2

array with ECC coding across the disks

RAID-3

array with parity coding. All parity bits in the same disk.

RAID-4

array with larger units on interleaving, so an entire file can be accessed on the same disk.

RAID-5

RAID-3, except the parity bits are stored across all the disks, getting rid of the hot spot of a parity disk.

RAM

Random Access storage. Given an access, the time to perform the next access is independent of the location of that access relative to the preceding access.

re-order buffer (result buffer)

a structure in the microarchitecture that can hold results that are executed out-of-order so these results are still returned to the software in program order.

ready bit (Out-of-Order)

In an out-of-order execution machine, an operation can execute when all its source operands are available. To keep track of the readiness of source operands, a single ready bit is maintained, indicating whether the corresponding source operand has been produced. When all the ready bits are 1, the operation can be scheduled to a function unit for execution.

reciprocal approximate

a quick way to do division. *a* divided by *b* is replaced by *a* times 1 divided by *b*. The reciprocal approximate unit takes *b* as input and produces 1 divided by *b*.

redundancy

the concept of using more than *n* bits to store *n* bits of information in order to recover in case one or more of the bits get corrupted. Parity provides one extra bit of redundancy. ECCL provides $\mathbf{log}\ \boldsymbol{n} + \mathbf{1}$ extra bits of redundancy.

reference bit

a mechanism used for page replacement. If the reference bit mechanism is used for page replacement, each PTE contains a reference bit. Periodically, all reference bits in the page table are set to 1. When a page is accessed, its reference bit is cleared. When a page fault occurs, the page chosen to be kicked out to make room is taken from the set of pages whose reference bits are still 1.

refresh

See DRAM cell.

register alias table (RAT)

the table of aliases used by the Tomasulo algorithm to assign tags. Each register has a valid bit, a tag and a value. Depending on the valid bit, either the tag or the value is correct.

register allocation problem

programs have variables that are assigned values. The program would run faster if each of these variables could be assigned a register. For example, R1 is assigned to VELOCITY, R2 is assigned to TIME, and R3 is assigned to DISTANCE. Then computing DISTANCE would be a single multiply instruction in computers with MUL opcodes. That is, MUL R3,R1,R2. Unfortunately, most programs have far more variables than registers. So, one of the jobs of the compiler is to optimally utilize the registers by moving variables between memory and registers so when you need a variable, most of the time it is in a register and does not require a Load instruction first. We call the problem that the compiler has of managing which variables use which registers and when, the register allocation problem.

register renaming (aliasing)

Recall that because we do not have an infinite number of registers, we need to use the same register to hold different values at different times in the flow of the program. Due to the nature of out-of-order execution processing, it is likely that two instruction that write to the same register could be active at the same time. Thus it is important to separately identify each INSTANCE of each destination write. We do this by assigning a tag (usually a physical register number) to each destination operand. In Tomasulo's time, that tag was referred to as an alias. Recall our figure in class – the Greek letters represented tags. Recall the data flow example – the tags correspond to arcs.

register renaming

also called aliasing. Assigning unique register names to independent uses of the same register.

register spill code

that part of an object program generated by a compiler that writes values back to memory because there are not enough registers in the microarchitecture to accomodate all the temporaries required of the high level program.

register spill

Registers are useful of storing values that will need to be sourced, rather than write them to memory and have to read them from memory again. More registers mean more values can be kept in registers. Unfortunately there are not enough registers, so at some point we need to write the values of registers to memory in order to make them available to hold new temporary values. We call that process of writing the registers to memory to temporarily get them out of the way and make room: register spill, and the compiler output that does this “register spill code.”

relational nodes

a node having two inputs and one output, the output being a boolean. The node performs the relation operation on its two inputs and produces the output token TRUE if the relation is true, or FALSE if the relation is false.

reorder buffer (ROB)

A convenient register file that allows destination writes to take place in any order, but reads for the purpose of retirement must be done in program order.

replacement policy (LRU, FIFO, random, victim/next-victim)

algorithm to determine which line within a set is to be evicted if a cache miss occurs, and one of the lines in the set must go. The line to be evicted is often called the victim. The line that will become the victim after the victim is evicted is the next-victim.

representativeness (of data)

see profiling.

reservation station

storage location for an instruction awaiting operands before it can execute.

resident

a term used to describe the fact that a virtual page is actually occupying a frame of physical memory.

residue

another word for remainder

restricted data flow

data flow in the Tomasulo sense, wherein only that subset of the data flow graph that corresponds to instructions that have been decoded but not retired is contained. I coined the term in 1984 to differentiate it from the data flow graph of the entire program

restricted data flow

my word for the data flow graph produced by generating it (a la Tomasulo) at run time. A compile time data flow representation of a program would be an enormous data flow graph. Generation at run time restricts the data flow graph to only those instructions in the active window. The active window is the set of instructions that have been fetched but not retired yet.

retirement (commit, completion)

Important in an out-of-order execution machine, where precise exceptions must be maintained. Instructions execute when they are ready, but they must commit their results in the order of the program (in program order) – hence the term "in-order retirement." We use the word retire or commit to designate that act of permanently changing the destination register.

rich instruction set

An ISA with many opcodes, many addressing modes, and supporting many data types.

ROM

Read Only Memory. Memory that can not be written into. It can only be read. Each bit has been hard-wired to the value 0 or 1.

rotation time

the time it takes the correct starting point on the track to come under the head where it can be read/written.

round down

one of the four rounding modes. Rounding is accomplished by rounding toward the next smaller value. All values are rounded toward -infinity.

round to unbiased nearest (beautiful rounding)

one of the four rounding modes. Rounding is accomplished by rounding to the next value (greater or smaller) that is closer in magnitude to the value being rounded. If equidistant, then we round to the value (of the two) that has a 0 in its ULP.

round up

one of the four rounding modes. Rounding is accomplished by rounding toward the next larger value. All values are rounded toward +infinity.

row address strobe (RAS)

see page mode.

row address register

An address is made up of many parts: channel bits, bank bits, rank bits, row address bits, and column address bits. Corresponding to the row address is a row buffer, which identifies all bytes having the same channel bits, bank bits, rank bits, and row address bits. When we access a DRAM chip, we load the row address register with the row address bits. Subsequent accesses to other locations having the same channel bits, bank bits, rank bits, and row address bits (i.e., only the column address bits are different will hit in the row buffer, and allow the access to occur much faster than would otherwise be the case since the byte can be obtained from the row buffer. We call such an access, where the required byte is obtained from the row buffer, "page mode."

Row Buffer Hit

An access that hits in the row buffer. That is, an access to the same row as the previous access.

Row Buffer

We access a DRAM with the address of the datum we want. We generally do this in two steps, first latching the high order bits of the desired address (accompanied by the RAS signal), then latching the low order bits of the desired address (accompanied by the CAS signal). As a result of latching the high order address bits, data at all addresses having these high order address bits are latched into a structure. That structure is called a row buffer.

row-major order

an array stored in memory in the order: first the components of the first row, then the components of the second row, etc.

run-time predictor

the guess is determined by the hardware on the basis of what has been happening with the running of the program.

runtime (dynamic)

that which is done while the program is running. For example, a run-time branch predictor makes its prediction based on information that is collected while the program is being executed. It is important to note that in the old days, there was a clear distinction between compile time and run time. That is, the program was compiled. Then the program was executed. Today, more and more research has gone into the notion that the program should be amenable to re-compiling while it is running. That is, once the hardware knows that a decision that was made at compile time was a bad decision, the hardware ought to be able to reinvoke the compiler with this information and generate new object code that takes advantage of this new information. This concept is still in its infancy.

SACK

signal output by the bus grantee signifying it will be the next bus master.

safe dataflow

wherein a node fires when all inputs have tokens, **and** there is an available slot for the result. Pro: Queues are not required. Minus: the node is kept from firing until an output slot is available, even though it possibly could have fired earlier.

saturating arithmetic

arithmetic which does not wrap around. That is, the highest value serves the role of infinity. i.e., **infinity + 5 = infinity**, therefore, **highest value + 5 = highest value**.

scalability

a measure of the ability of a design to produce linearly improving benefits as the number of elements in the design continues to increase linearly. For example, if we add *k* ALUs, do we get *k* times the performance. If we increase the frequency by *x*, do we compute the answer *x* times faster.

scalar computer

what you are used to. No vector registers, no vector loads, stores, adds, etc.

Scalar Registers

registers that contain one value. As compared to vector registers, which contain multiple values, each of which is a component of the vector.

scalar

An operand that has a single value. For example, the integer 5. Scalar operations are performed by scalar functional units on scalar values.

scheduling algorithm

an algorithm used to decide which of several ready instructions should be executed next.

science of tradeoff

My tongue in cheek phrase to emphasize the importance of tradeoffs to the discipline of computer architecture. Clearly, computer architecture is more art than science. Science, we like to think, involves a coherent body of knowledge, even though we have yet to figure out all the connections. Art, on the other hand, is the result of individual expressions of the various artists. Since each computer architecture is the result of the individual(s) who specified it, there is no such completely coherent structure. So, I opined if computer architecture is a science at all, it is a science of tradeoffs. In class, we keep coming up with design choices that involve tradeoffs. In my view, "tradeoffs" is at the heart of computer architecture.

Scoreboard bit

a bit indicating whether the resource associated with that bit contains valid data.

scoreboard

a hardware interlock that prevent an instruction from sourcing stale data.

SECDED

A code (often referred to as Hamming Codes or ECC codes) that will identify single bit errors, thereby allowing them to be corrected, and will also allow two bit errors to be detected, although not corrected. The acronym stands for single error correct, double error detect.

Second level (L2) cache

a larger and slower cache that is accessed if the access to the first level cache results in a miss.

sector cache

the characteristic of a cache whereby each cache line is partitioned into sectors, whereby each sector is individually valid. Thus, if a cache line is partitioned into four sectors, there are four valid bits in the tag store entry for that line. Useful in those cases where one wishes to allocate on a write miss, but not read in the entire line from memory because it is likely that the entire line will be completely overwritten before it is ever read.

SED

Single error detect. Applies to a code that can only detect single bit errors. For example, parity protection is SED.

seek time

the time it takes the head to get to the desired track to read or write the information.

segment register

a register used in conjunction with the address provided by the program to produce an address which is either a physical address that can be accessed directly or a virtual address which needs to be further translated to a physical address.

segmentation

a protection mechanism to protect regions of memory for different kinds of accesses.

selective mircocode

semantic gap

an old term, used to describe the closeness of fit between the instructions in a high level language and the opcodes in an ISA. For example, the VAX had an opcode AOBLEQ which added 1 to an operand, and branched if the result of the add was less than or equal to some limit operand. This instruction was clearly in the ISA to accomodate "for" loops. The semantic gap between the "for" and AOBLEQ was tiny. In the old days, there was strong interest in specifying opcodes that had a small semantic gap relative to the high level languages commonly in use at that time. That interest has waned during the past 20 years, but there is good reason why we may see it return in the future.

sequential (serial) bottleneck

see Amdahl's Law.

sequential access storage

access is always sequential to the next location. Most common type is a tape.

sequential programming model

instructions are expected to execute in the order specified by the programmer.

service call/system call

An operating system function performed on behalf of the running program, invoked by a trap instruction. The trap vector identifies the operating system function to be performed.

service routine (handler)

see above.

set-associative cache

a cache characterized by the property that a line of memory can be stored in any of k entries in the cache. We say such a cache is k-way set-associative, and the set size is k.

setup time/ hold time

A cycle time is specified by the time it takes to propogate signals through all the gates to arrive at results that need to be latched at the clock edge identifying the next clock cycle. Set-up time refers to the time needed for all signals to be stable before the next clock edge occurs in order to guarantee that the correct results will be written correctly. Hold time refers to the amount of time after the next clock edge where the new inputs (outputs from the previous clock cycle) need to remain stable before being allowed to change during that clock cycle. In our work with the LC-3b, we are assuming you can not change the inputs until the end of the clock cycle.

shared global memory

see multiprocessor.

shift and add multiplication algorithm

Recall your experience with the multiplication of decimal numbers. We systematically multiply the multiplicand by each digit of the multiplier and write each corresponding temporary result right-aligned with the corresponding multiplier digit. Then we add up these temporary results, producing the product. In multiplying binary numbers, the only digits are 1 and 0 so the multiplication step consists of simply

copying the multiplicand or writing all zeroes as each temporary result. If we are multiplying n bit numbers, we get n temporary results. To add these temporary results properly, we need to take into account their relative alignment. This can be done by n-1 steps where each step includes the sum of all temporary steps before it. That is, after step 1, we have correctly added the temporary results for bits 1 and 2. After step i, we have correctly added the temporary results for bits 1, 2, ... i+1. The (i+1)th step consists of SHIFTING right the (i+2)th temporary result one bit so as to align it with the sum of the first i steps and then ADDING the two quantities. Thus, the name shift and add to describe binary multiplication.

Shift and Add multiplier

classical way to do binary multiplication. Since multiplying by 1 simply means adding in the multiplicand, and multiplying by 0 means adding zero, both after aligning (shifting) properly, multiplication is usually described as a series of shifts and adds, one for each bit in the multiplier.

short integer

an integer of the size of the word length.

SIB Byte

A byte in the x86 ISA that extends the ISA's addressing mode capability. You know that you can compute an effective address of an operand by adding an offset to a register. With an SIB byte, one can augment that address computation by adding the contents of a Base register and/or the contents of an Index register (after multiplying the contents of the index register by a scaled amount). The modR/M byte specifies whether an SIB (Scale-Index-Base) byte is included in the x86 instruction. If yes, the effective address computation includes the augmentation described above.

signed magnitude

the representation of a value that uses one bit for sign and the remaining bits for magnitude. Used in floating point data type.

signed-magnitude

ditto

SIMD (data parallel)

processing that involves a single instruction stream such that each instruction operates on multiple data sets. Two forms of SIMD processing exist: array processors and vector processors.

SIMD

single instruction stream, multiple data sets. Sometimes called data parallel. Two forms: vector processors and array processors.

simultaneous multithreading (SMT)

a variation of multithreading in which the execution stage of the pipeline executes out-of-order. The result is that multiple functional units can be simultaneously executing instructions from multiple threads.

SISD

single instruction stream operating on a single data set.

slave

the other controller that participates in the bus transaction, under the control of the bus master.

SMP (Symmetric multiprocessor)

A multiprocessor, wherein any of the processors is capable of taking control of the kernel of the operating system. Co-processor: also called an attached processor. What we might call today an accelerator. A processor that can not do substantial work on its own. No PC, generally can not even do full decode. Usually just does the operation required by the instruction after being supplied with the source operands.

snoopy cache

a mechanism for maintaining cache coherency. This is done by allowing each cache to monitor the accesses of other processors to memory so that it knows whether a particular cache line is exclusive to itself or shared with other caches. …or is being written by another processor and so it (this cache) must either invalidate its copy or update it with the value being written by the other processor.

soft error

A soft error is an error that is not hard. Duh.

So, what does that mean? We distinguish errors as transients errors and permanent errors. A transient error is one that occurs due to some event that for purposes of the computer is random. Someone started up a

washing machine in the same room, and there is not enough noise margin. Or, some alpha particle got in the way, which gets more and more common as the clock gets more and more into the GHz range. The idea is that if you replay the program, the error will not show up the second time.

The other kind of error is the permanent error. Here the device is broken and always give you the bad result.

Permanent errors are called hard errors. Transient errors are called soft errors.

Software managed cache
a tongue in cheek description of Cray T registers.

spatial locality
the property that if a location in memory is accessed, it is likely that locations close by will be accessed soon.

speculative execution
activity that is carried out before the hardware knows it must be (mandatory) carried out. For example, instructions fetched as a result of a branch prediction.

speedup
a measure of the improvement in performance achieved by adding processing capability to a problem. In its pure form,

$$\textbf{Speedup}(p) = \textbf{Time}(1) \div \textbf{Time}(p)$$

where Time(1) is the time a single processor requires to compute the answer using the best available algorithm for a single processor, and Time(p) is the time the processor requires to compute the answer using p processors.

split-transaction bus
A bus that breaks its transaction into two parts, the first part being the request from the master to the slave, and the second part being the completion of the transaction (i.e., either the data supplied by the slave, or the data accepted by the slave from the master). While the request of the first part of the bus transaction is being serviced, instead of twiddling its thumbs, the split-transaction bus releases the bus to allow other transactions to use it.

SPMD (single procedure multiple data set)
a variation of SIMD, wherein the level of synchronization is a procedure rather than each instruction.

SRAM
Static random access memory. An SRAM cell stores a single bit of information with two cross-coupled inverters. The feedback path maintains the state of the bit. As long as power is supplied to the two inverters, the SRAM cell maintains its value. The SRAM cell is also referred to as volatile memory, since if power is removed from the inverters, the bit that is stored is lost. [Cross-reference, non-volatile, where the bits retain their values even when power is removed, as in a disk or tape.] The minimum number of transistors for an SRAM cell is 4 -- two inverters, with 2 transistors per inverter. However, most SRAM cells use more transistors to provide for other features, such as switching speed, gating, noise immunity, for example.

SSYN
signal used by the slave in synchronizing the transaction.

stale data
data that was stored in a location but the location has since been reassigned so that the old data is no longer valid for new instructions that come along.

Standard Performance Equation
Performance = 1/T = 1/(l x cpi x cycle_time).

starvation
A condition in which a controller may never get the bus it is requesting. Recall the example of the daisy chain grant of the BG signal.

state machine

the structure that specifies, for each state of the machine, and each set of relevant values (opcode, interrupt present, memory ready, etc.) what set of control signals should be invoked during the next clock cycle.

state of machine

Computer processing is all about systematically changing the state of the machine in response to each instruction in the dynamic execution of the program. That is a program executes by the machine being in an initial state, and one by one each instruction execution changes the state in accordance with the semantics (specification) of the instruction. The machine state consists of the PC, the PSR, the general purpose registers, the condition codes (if the ISA specifies condition codes), the contents of each memory location, and the mapping registers that enable the memory management process. More sophisticated ISAs provide additional state.

static instruction stream

The program as it sits in memory, or rather out on the disk. Each instruction generated by the compiler exists once in the static instruction stream.

steering bit

A bit in an instruction that specifies the interpretation of other bits in the instruction, depending on the value of the steering bit. For example, bit[5] in the LC-3b is a steering bit, since it determines the usage of bits[4:0] of the instruction.

sticky bit

a condition code that once set retains its condition until interrogated under program control.

store result/write back

The final stage of the instruction cycle where the result is written, completing execution of the instruction.

Streaming Workload

Streaming generally refers to large units of data that are loaded sequentially from memory, then operated on, after which they are typically sequentially stored back to memory.

stride

the distance in memory between locations to be accessed vis-a-vis successive registers.

striping

interleaving, when applied to disk.

Stripmining

The process of handling more than 64 iterations of a loop, 64 at a time.

subnormal

see gradual underflow.

Superlinear speedup

On a plot of speedup vs number of processors, the 45 degree line reflects all points such that with k processors, we get speedup = k. Superlinear speedup refers to points above the 45 degree line -- that is points such that with k processors we get speedup greater than k. ...which begs the question: what are you forgetting or what are you hiding.

superpipelined

a term used to describe a pipeline wherein the instruction fetch rate is a multiple of the rate at which instructions can be processed by one of the functional units in the pipeline.

superscalar processor

A processor that fetches more than one instruction at a time and by virtue of the logic it entails (multiple decode, branch prediction, out-of-order execution) is able to obtain (hopefully) an IPC greater than 1. Note: it does not have to obtain an IPC greater than 1. Rather it has the enablements to accomplish that. Some have argued that as long as more than one instruction is being fetched and decoded each cycle, and that number is determined at run-time, it is a superscalar processor. The name derives from the fact that before these multiple fetch processors came around, the only processors that had an IPC greater than one were vector processors. Ergo this scalar processor that provides in some sense capability previously reserved for vector processors were not simply scalar processors, they were super scalar processors.

superscalar

a microarchitecture paradigm where more than one instruction is fetched and decode each cycle, the exact number determined by the fetch/decode engine, that number dependent on the actual instructions fetched.

Supervisor Stack, Kernel Stack, Interrupt Stack

Many ISAs provide a system stack for use of the operating system, and per process stacks for each privilege level that a process can run in. The system stack is sometimes referred to as the Interrupt Stack since its most important function is push/pop state information on the system stack during the process of handling interrupts. If the ISA specifies two privilege levels (like Unix), there is often implemented a Supervisor stack and a User stack. If the ISA specifies four levels of privilege (like VAX), then the microarchitecture often implements four separate stacks. One reason for the multiple stacks is to prevent unauthorized access to data remaining on a higher level stack after that data has been popped, but not removed.

Supervisor/User Cache
A method of partitioning a cache so that lines of memory accessed by the processor when running in supervisor mode compete for cache lines in one partition of the cache, and lines of memory accessed by the processor when running in user mode compete for cache lines in the other partition of the cache.

swap in/swap out
the act of bringing in or out the working set of a process. If the working set of a process is stored back to disk, we say the process is swapped out.

synchronous bus
a clocked bus.

synonym problem
Suppose two processes have different virtual addresses for the same physical address. In a virtually-indexed, physically tagged cache, since the virtual address is used to index into the tag store, the physical address could be present in two locations in the cache. This is known as the synonym problem. It is a particular headache since updating one location in the cache requires updating the other location in the cache since both locations correspond to the same address in memory.

system space
virtual address space that is common to all processes, and is used to house the data structures and code of system related functions.

tag, index, offset bits
A memory address is partitioned into three parts: its tag bits, its index bits and its offset bits. The offset bits define which location within the cache line is being accessed. The index bits are used to index into the tag store, and specify the set of locations in the cache available for storing the particular line. The tag bits are stored at the corresponding index in the tag store to verify that the actual line stored in the cache is the one desired. That is, all blocks/lines from memory having identical index bits in their addresses compete for the same entries in the cache. The tag bits are used to specify which of these many entries are actually stored in the cache.

tag/data store
The tag store is the part of the cache that keeps all the bookkeeping information associated with the blocks of memory that are actually stored in the cache. Most important of these is the identity of the actual block. The data store is the storage in the cache allocated for storing the actual information that is contained in the corresponding lines of memory. When we say a cache is of a certain size, we are referring to the size of the data store. [A system having 16MB of memory and a 64KB cache stores 64KB out of the 16MB in the cache's data store. We do not say how much tag store is required. The tag store is just management overhead to allow the cache to function properly.]

tag
another word for the alias or distinct identifier assigned to each instance of a register.

template bits
see EPIC.

temporal locality
the property that if a location in memory is accessed, it is likely that location will be accessed again soon.

thrashing
the condition whereby there is not enough frames of physical memory to hold all the virtual pages that a process needs, resulting in an inordinate amount of time being wasted by pages coming in and going out, only to come in again, etc.

tightly coupled MP
see multiprocessor.

TLB

translation lookaside buffer. It is a cache of PTEs. It allows translation to take place without going through the involved process of "walking" through the page tables. That is, within a fraction of a cycle instead of tens of cycles.

TMR (triple modular redundancy)

one mechanism for achieving fault tolerance. Three machines execute the program in lock step, and continually vote on the results they produce. The probability that two are both wrong and give the same wrong answer is sufficiently remote that processing continues correctly as long as two give the same answer. When one of the three gives a different answer from the other two, it is assumed to be broken. It is replaced with a known good machine, and the system continues. The system is continually available as long as the other two machines give identical answers during the period where the broken machine is replaced. We call the frequency of faults in a processor the MTTF (mean time to failure), and the time required to replace the faulty machine with a good one MTTR (mean time to repair). As long as MTTF is greater than MTTR, we should do just fine.

tokens

another name for values that are input to data flow nodes or results that are output from data flow nodes.

Tomasulo's algorithm

the algorithm which assigns register aliases that allows an instruction to move to a reservation station, while maintaining all producer/consumer relationships. This maintenance of the producer/ consumer relationships is the most important single thing needed to allow instructions to actually execute out of order.

Touch instructions

It would be nice if data that is needed in the next instruction is contained in the first level cache. The "touch" instruction tries to accomplish that by accessing the memory location containing that data. If this results in a hit in the first level cache, execution of the Touch instruction terminates. Otherwise, the data is copied from memory or the second (third, etc.) level cache into the first level cache. This is all done in the background, not affecting appreciably the processing of instructions. The resulting effect is that when the data is actually needed, it can be obtained from the first level cache immediately.

trace scheduling

a profiling technique used with VLIW machines to find branches that are highly skewed so as to schedule some of the instructions in a vliw word by violating control dependencies. Since 98% taken (a highly skewed branch) is not the same as 100% taken, additional instructions have to be provided to prepare the machine to undo instructions that should not have been executed if the branch behaved according to the other 2%, and still additional instructions to actually do the undo when that “bad” branch behavior occurs. The benefit of this technique is that most of the time the branch behaves according to the 98% behavior and so additional paralllelism can be exploited. The idea is that the undo time is small and is more than compensated for the extra parallelism 98% of the time.

track

on a platter, a complete circle, containing bits of information.

transducer

the thing that tranduces from one medium to another. From light passing through a punched card to the electrical signals. From the magnetized material on the disk track to the electrical signals.

tree

a connection network that forms a binary tree. See my handout on interconnection networks.

triad

An instruction with three source operands. Most common example is: MulAcc A,B,C,D which multiplies C times D, adds the result to B and store the result of the addition in location A. B,C, and D are source operands. Ergo, a triad.

ULP (unit in the last place)

the value of $\mathbf{0.000\ldots001 \times 2^{exponent}}$. For example, with 4 bits of fraction, ***exponent* = -2**, one ULP is 1/64.

unaligned access

an access to memory that is not aligned on an access boundary, thereby requiring two accesses to perform the desired operation. See my handout on unaligned accesses.

underflow

an exception caused by a computed value being too small in magnitude to be expressed other than by 0.

Unified L2 cache
A single L2 cache that contains both code (from the L1 instruction cache) and data (from the L1 data cache)

.

uniform vs. non-uniform decode
fixed length instructions may have the property that the various fields of the instruction always occur in the same place, simplifying the decode process. We call such an encoding of the instruction format: uniform decode. The LC-3b for example has the opcode ALWAYS in bits 15:12 and the result operand always in bits 11:9.

uniprocessor cache consistency
We probably also ought to note here that the rest of the world would call this “coherency” not “consistency.” Normally, we think of cache coherency only when we have more than one processor that can write to its own private cache. We will deal with this more after the exam. It is worth noting however that in a uniprocessor environment with intelligent I/O devices, the I/O device (e.g., DMA controller) can independently write to memory, in which case data in the cache may no longer be valid. Ergo, in an environment with intelligent I/O devices, the cache must monitor independent accesses to memory from the I/O subsystem in order to be sure that data in the cache has not been invalidated by what the I/O device wrote to memory.

uniprocessor
One processor. Also called "one core," as differentiated from multi-core, more than one processor on a single silicon die. The processor can be any one of several different kinds of processors, but usually we think of a single core as able to execute a single instruction stream, one PC, one set of gprs, etc.

user space
virtual address space specific to a process. Sometimes called process space. Each process has its own version of process space.

Utilization of a multiprocessor
A multiprocessor ties up p processors for Tp units of time. Utilization is a measure of the fraction of time that the processors are actually doing useful work. That is, Op/pTp.

valid bit
a bit determining whether other bits are accurate. In the case of the PTE, the valid bit tells whether the hardware can believe the PFN bits, which really says whether the page is resident or not. If V=0, the page is not resident, forcing the hardware to take a page fault.

vector chaining
instruction processing wherein the results of one vector operation can be forwarded to the next vector operation without waiting for all components of a vector instruction to be completed.

vector length register
identifies how many of the components of a vector register, or vector operation, are operative.

vector load
multiple values loaded as a result of a single load instruction.

vector operation
an operation that is performed on a set (or sets) of values (referred to as vectors). For example, a vector A consists of n values A1, A2, A3, ..An. And a vector B consists of n values B1, B2, ...Bn. A vector add of vectors A and B would produce the vector consisting of A1+B1, A2+B2, ... An+Bn. We often refer to this mode as data-parallel, since there are no dependences between the component adds, and they can all be executed in parallel.

vector processing
the paradigm we discussed in class for handling vectorizable loops.

vector processor
a processor containing vector instructions, i.e., instructions that operate, in turn, on multiple values.

vector register
a register consisting of multiple components, useful for storing the result of a vector load, vector add, etc.

vector stride register
register containing the current stride value.

vector

An operand that has more than one component value. For example the sequence 5,3,7,9,7. If this was the value of a vector V, then V1=5, V2=3, V3=7, etc. Vector operations are performed by vector functional units on vector values. For example, the vector add of 5,3,7,9,7 and 4,-3,2,-7,1 in a vector adder is 9,0,9,2,8.

vectorizable

a loop instruction whereby each iteration of the loop is completely independent of the other iterations.

Vertical Microcode

Microcode in which the fields in a microinstruction are mostly interdependent and combine to perform a single micro-op. This generally results in many more microinstructions to implement a specific task. Vertical microinstructions resemble machine instructions in the sense that each microinstruction generally performs a single micro-command.

vertical priority

priority determined by the particular BR line.

virtual machine (emulation)

An ISA is implemented in hardware. Each instruction in that ISA is fetched, decoded, etc. and at each step of the instruction cycle, control signals are generated to carry out the phases of that instruction. What if you wanted a program compiled into the ISA of machine A to be executed on a machine that has a different ISA, say ISA B. Clearly, the 0s and 1s of the ISA of A have no meaning with respect to the machine of ISA B. We call the ISA of machine A a virtual machine because it does not correspond to real hardware in our situation. What is needed is a program (usually called an emulator) that translates each instruction of ISA A into a sequence of instructions from the ISA B that can then be carried out on the machine of ISA B and produce the same result as it would have if the ISA A instruction had been carried out on a machine of ISA A. We often refer to ISA B as the host machine since that is where the results are actually being produced. And ISA A the target machine since that is the code produced by the compiler from the original program. The concept has been around for a very long time, although it is lately gaining a lot of popularity both in research universities and product development centers. An early example was the Burroughs 1700 of the 1970s. Programs were compiled to three different virtual machines, depending on whether the nature of the code was scientific, business, or system level in nature. The resulting three different ISAs were then emulated on the host B1700 machine. More recently, there have been plenty of Java programs that have been compiled into an ISA referred to as the Java Virtual Machine. The result is that other companies have produced software emulator to translate each JVM instruction into the ISA of its own host machine. So, for example, a Pentium M can run JVM instructions if it wishes by emulating each JVM instruction into a sequence of x86 instructions that does that job.

An analogy to virtual memory is in order. Like virtual memory, a virtual machine exists in the eye of the beholder. Recall a virtual address has to be translated to a physical address before it can be accessed. A virtual machine instruction has to be translated to a sequence of instructions in the host ISA before it can be carried out.

virtual memory

memory that a process thinks it has. Generally, the size of the virtual address space.

virtually-indexed, physically-tagged cache

a cache wherein the index bits used to access the tag store are obtained from the virtual address, but the tag bits actually stored in the tag store are physical address bits. This allows TLB and tag store accesses to be made concurrently.

virtually-indexed, virtually-tagged cache

a virtually addressed cache.

VLIW (very large instruction word)

a paradigm wherein the fetch/decode mechanism fetches *n* instructions from the instruction memory each cycle, and executes the *n* instructions concurrently. The value *n* is a fixed value for the machine. The set of *n* instructions fetched in one cycle, that is the contents of one VLIW word, is specified by the compiler. In order for the *n* instructions to execute concurrently, they must all be independent with respect to flow

dependencies. It is the job of the compiler to identify *n* instructions that can be packaged together as a single VLIW word.

VLIW

Very Long Instruction Word. A processing paradigm wherein the compiler generates a long word, consisting of more than one instruction, all stored at the same PC address. At instruction fetch, all instructions in that long word are fetched, decoded, and executed concurrently in lock step. To work, it is necessary that there be no flow dependencies among the instructions in a single long word. This paradigm provides benefit if the compiler can find at compile time multiple instructions that have no flow dependencies, and therefore, can execute concurrently. The number of instructions n in the long word (VLIW instruction) is fixed. Therefore, if the compiler can not find n independent instructions to put into one VLIW instruction, it must fill the remaining slots with no-ops.

volatile/non-volatile

Volatile storage loses its data when power is turned off. Memory is volatile. Non-volatile storage retains its data when power is turned off. CDROMS, disks, tapes are all non-volatile.

VPN

virtual page number. The high bits of a virtual address; it specifies the page number.

wilkes diode matrices

Maurice Wilkes 1951 paper on the best way to design a calculating machine included a figure showing each microinstruction as a horizontal wire and each vertical wire as a control signal. The vertical wires were partitioned into two parts, the vertical wires on the left were signals to control the data path. The vertical wires on the right formed an address which was used to access the next microinstruction. The left part was referred to as Diode Matrix A because diodes were place at some of the intersections of the vertical and horizontal wires. These diodes were forward biased from the horizontal wire to the vertical wire so that a voltage carried by a horizontal wire (microinstruction) would assert a subset of the control signals. The right part was referred to as Diode Matrix B because the corresponding diodes produced the address (0s and 1s) for the next control store address.

wish branch

an invention of Professor Hyesoon Kim during her PhD research. A new opcode for the compiler to use instead of a normal branch if the nature of the branch is such that predication would make sense if the runtime prediction accuracy of that branch turns out to be low. The compiler can not know how good the prediction accuracy of that branch will be at compile time. So the compiler generates a wish branch rather than a conventional branch. At run time when the hardware knows what the branch prediction accuracy is, it can either convert the branch dynamically to predicated code or to a conventional branch.

wobble

the ratio of error introduced by rounding when the result of rounding is a power of the radix. For radix 2, one ULP larger than $\mathbf{2^n}$ is twice as large as one ULP smaller than $\mathbf{2^n}$. Therefore the wobble is 2.

word length

the size of the ALU, the size of the registers, the default value of sizes.

Word Line

In memory, a word line consists of all bits that make up the contents of a word of memoy.

working set

the set of pages belonging to a process that is resident in memory at an instant of time is called the process' working set at that instant of time.

write back buffer (store buffer)

Loads must be serviced as soon as possible, since subsequent instructions are usually dependent on the data being loaded. Stores, on the other hand, can take place when convenient. [Note: in a multiprocessor, that may not be quite so simplistically stated, but that is an issue we will deal with later this semester.] A store buffer allows the processor to put values to be written to the side and not tie up bus bandwidth unnecessarily. For example, on a load miss to a write back cache, put aside the dirty cache line to write to memory until after the new cache line is read in. For example, on a write through cache hit, we may want to buffer multiple writes to the same cache line, so as to only have to do one memory write and tie up bus bandwidth once. The write buffer is an extension of the memory system and needs to be considered as such. That is, an access to the cache also accesses the write buffer.

write enable

A signal that must be asserted if a storage structure is to be written. In the technology we use in 360N, if the signal is asserted throughout the cycle, the storage device is written at the end of the cycle.

writeback/write through

a property of caches. A write through cache is one in which every write to the cache is also written to memory. In this case, memory and the cache are always consistent. On replacing a block, it is unnecessary to write that line back to memory. A write back cache is one in which writes to the cache do NOT also include writes to memory. Therefore, if a cache line is to replaced, its dirty bit is checked to see whether any writes to that line have occurred since it was brought in from memory. If the dirty bit is set, indicating that writes have occurred, then before the line can be removed from the cache. it has to be written back to memory first.