Micro operations microinstruction micro program micro code

Technology, Computer

Each microinstruction in a microprogram provides the bits which control the functional elements that internally compose a CPU. The advantage over a hard-wired CPU is that internal CPU control becomes a specialized form of a computer program. Microcode thus transforms a complex electronic design challenge (the control of a CPU) into a less-complex programming challenge.

Microcode is a layer of hardware-level instructions and/or data structures involved in the implementation of higher level machine code instructions in many computers and other processors; it resides in a special high-speed memory and translates machine instructions into sequences of detailed circuit-level operations. It helps separate the machine instructions from the underlying electronics so that instructions can be designed and altered more freely. It also makes it feasible to build complex multi-step instructions while still reducing the complexity of the electronic circuitry compared to other methods. Writing microcode is often called microprogramming and the microcode in a particular processor implementation is sometimes called a microprogram.

Microcode can be characterized as horizontal or vertical. This refers primarily to whether each microinstruction directly controls CPU elements (horizontal microcode), or requires subsequent decoding by combinational logic before doing so (vertical microcode). Consequently each horizontal microinstruction is wider (contains more bits) and occupies more storage space than a vertical microinstruction.

Modern microcode is normally written by an engineer during the processor design phase and stored in a ROM and/or PLA structure, although machines

exist which have some writable microcode in SRAM or flash memory. Microcode is generally not visible or changeable by a normal programmer, not even by an assembly programmer.

Some hardware vendors, especially IBM, use the term as a synonym for firmware, so that all code in a device, whether microcode or machine code, is termed microcode (such as in a hard drive for instance, which typically contain both).

Microcode was originally developed as a simpler method of developing the control logic for a computer. Initially CPU instruction sets were “ hard wired”. Each step needed to fetch, decode and execute the machine instructions (including any operand address calculations, reads and writes) was controlled directly by combinatorial logic and rather minimal sequential state machine circuitry. While very efficient, the need for powerful instruction sets with multi-step addressing and complex operations (see below) made such “ hard-wired” processors difficult to design and debug; highly encoded and varied-length instructions can contribute to this as well, especially when very irregular encodings are used.

Q. 2. How Information Technology can be used for strategic advantages in business?

Ans.: All the advantages and disadvantages of information technology, it is essential that we know what information technology is exactly, and why it has it come to play such a important role in our daily lives. Today information technology involves more than just computer literacy; it also

takes into account how computers work and how these computers can further be used not just for information processing but also for communications and problem solving tasks as well.

Globalization – IT has not only brought the world closer together, but it has allowed the world’s economy to become a single interdependent system. This means that we can not only share information quickly and efficiently, but we can also bring down barriers of linguistic and geographic boundaries. The world has developed into a global village due to the help of information technology allowing countries like Chile and Japan who are not only separated by distance but also by language to shares ideas and information with each other.

Communication – With the help of information technology, communication has also become cheaper, quicker, and more efficient. We can now communicate with anyone around the globe by simply text messaging them or sending them an email for an almost instantaneous response. The internet has also opened up face to face direct communication from different parts of the world thanks to the helps of video conferencing.

Cost effectiveness – Information technology has helped to computerize the business process thus streamlining businesses to make them extremely cost effective money making machines. This in turn increases productivity which ultimately gives rise to profits that means better pay and less strenuous working conditions.

Bridging the cultural gap – Information technology has helped to bridge the cultural gap by helping people from different cultures to communicate with one another, and allow for the exchange of views and ideas, thus increasing awareness and reducing prejudice.

More time – IT has made it possible for businesses to be open 24 x7 all over the globe. This means that a business can be open anytime anywhere, making purchases from different countries easier and more convenient. It also means that you can have your goods delivered right to your doorstep with having to move a single muscle.

Creation of new jobs – Probably the best advantage of information technology is the creation of new and interesting jobs. Computer programmers, Systems analyzers, Hardware and Software developers and Web designers are just some of the many new employment opportunities created with the help of IT.

Q. 3. What Characteristics of software make it different from other engineering products?

Ans.: A large number of software standards have been developed concerned with software products and processes, terminology and more general fraework standards. While the need for software standards is not disputed, it is felt that many standards fail to take into account the essential differences, and occasionally the similarities, between software products and processes and other engineering products and processes. Ideally, standards should be useful, testable and represent a consensus view. It appears that some

software standards fall significantly short of these objectives. Further progress in developing useful software standards requires a better understanding of the potential benefits which standards have, and do not have, to offer the software industry. The relevant issues are discussed and the next steps which should be taken in developing software standards suggested.

Q. 4. What are different addressing modes available?

Ans.: Addressing modes are an aspect of the instruction set architecture in most central processing unit (CPU) designs. The various addressing modes that are defined in a given instruction set architecture define how machine language instructions in that architecture identify the operand (or operands) of each instruction. An addressing mode specifies how to calculate the effective memory address of an operand by using information held in registers and/or constants contained within a machine instruction or elsewhere.

In computer programming, addressing modes are primarily of interest to compiler writers and to those who write code directly in assembly language

Absolute

+—-+————————+

| jump| address |

+—-+————————+

(Effective PC address = address)

The effective address for an absolute instruction address is the address parameter itself with no modifications.

Absolute/Direct

+——+—-+————————————-+

| load | reg | address |

+——+—-+————————————-+

(Effective address = address as given in instruction)

This requires space in an instruction for quite a large address.

Register

+——+—-+—-+—-+

| mul | reg1| reg2| reg3| reg1 := reg2 * reg3;

+——+—-+—-+—-+

This “ addressing mode” does not have an effective address and is not considered to be an addressing mode on some computers.

In this example, all the operands are in registers, and the result is placed in a register.

Immediate/literal

+——+—-+—-+———————-+

| add | reg1| reg2| constant | reg1 := reg2 + constant;

+——+—-+—-+———————-+

This “ addressing mode” does not have an effective address, and is not considered to be an addressing mode on some computers.

The constant might be signed or unsigned.

Implicit

+———————-+

| clear carry bit |

+———————-+

The implied addressing mode does not explicitly specify an effective address for either the source or the destination (or sometimes both).

Either the source (if any) or destination effective address (or sometimes both) is implied by the opcode.

Indexed absolute

+——+—-+—-+———————————-+

| load | reg | index| address |

+——+—-+—-+———————————-+

(Effective address = address + contents of specified index register)

This also requires space in an instruction for quite a large address. The address could be the start of an array or vector, and the index could select the particular array element required. The processor may scale the index register to allow for the size of each array element.

PART A? A? aˆsA¬” B

Q. 5. How will you differentiate b/w Arrays and Stacks? Explain by giving an example.

Ans:

There are two main differences between an array and a stack. Firstly, an array can be multi-dimensional, while a stack is strictly one-dimensional. Secondly, an array allows direct access to any of its elements, whereas with a stack, only the ‘ top’ element is directly accessible; to access other elements of a stack, we must go through them in order, until we get to the one we want.

Q. 6. How Assembler differs from Complier?

Ans.: An Assembler converts Assembly instructions into executable machine language. A Compiler converts higher level programming language instructions into Assembly instructions, and then those are turned into executable machine language. Most Compilers allow generation of “ object” code, which is the Assembly instruction set generated by the Compiler. Some older Compilers allow for the Assembly instructions to be fine tuned by the programmer.

Compiled programming languages typically generate many lines of Assembly instructions for each program statement. Some programming languages, such as ANSI C, are very close to Assembly, while others such as Java, result in many Assembly instructions per program statement. Most Compilers are highly optimized and it would be difficult for a human programmer to improve the efficiency of the output.

Assembly level instructions are very difficult for someone not trained on Assembly to read and comprehend.

Q. 7. Out of Linear and Binary Search , which one is preferred where and why?

Ans.: Binary Search, because in Binary Search code take less no. of execution and it save time. But, in linear search it executes full time whenever search may give result. A binary search is an algorithm for locating the position of an element in a sorted list. It inspects the middle element of the sorted list: if equal to the sought value, then the position has been found; otherwise, the upper half or lower half is chosen for further searching based on whether the sought value is greater than or less than the middle element. The method reduces the number of elements needed to be checked by a factor of two each time, and finds the sought value if it exists in the list or if not determines “ not present”, in logarithmic time. A binary search is a dichotomist divide and conquer search algorithm.