UNLOCKING
CONSCIOUSNESS

BRIAN MIND FORUM

Appendix 032-H

How the central processors of computers process program, With special reference to how programs can be designed to write program

In 1948 various versions of stored program computers began to work in Manchester, Cambridge and Yale. 'The key was *stored program'*. Colossus was a computer but it could only do the one task it was designed to do.

The essence of a programmable machine is that one set of hardware can do a variety of different tasks.

However, these machines could do far more. They were designed so that programs could edit programs, which laid the foundation for designing programs, or software, that could learn and even perform intelligently.

This is how that is done. Computer processors come in many different forms, but they all have the same basic architecture. All computers have a central processor and a bank of Immediate Access Memory (IAM), connected by a communications 'bus' to various banks of long term memory, input peripheral systems including a keyboard, and maybe banks of communications systems, links to the internet, or to distant terminals; a screen, and various output peripheral systems like printers, graph plotters or a huge range of machinery.

The IAM consists of banks of binary bits (transistors), divided into registers. Each register can be interpreted by the processor as a number, or as five sections.

The first section contains the reference number of the register. The second contains the code number of the instruction, the third the first operand, the fourth the second operand and the fifth the number of the register containing the next instruction.

Thus an instruction in register 100 might be:-

100, add, (the contents of) register 200, to (the contents of) register 201, (execute the next instruction in register) 101.

The instruction set might typically contain:-

001 for add;

002 for subtract;

003 [the most important instruction in all computing]; if the contents of the register in the first operand is zero go to the next instruction in the register in the second operand. If the contents of the register in the first operand is not zero *go to the next instruction in the register in the fifth section in the normal way.*

004. Place the number of this register in the register in the first operand.

005. Place the number Zero in the register in the first operand.

Then various instructions would input information from the peripheral systems like a keyboard, display information on the screen, output information to the peripheral systems like the printers, and transfer information to and from the memory systems.

There are always instructions to store the contents of IAM in the Memory systems and replace the present IAM with information from the memory systems.

These last two instructions enable, say, a master program to upload an application program then transfer control to one of the instructions in that application. Similarly, one program might edit or modify another program then download that amended program to memory and so the edited version will be uploaded the next time it is needed. To re-emphasise the point, all the registers are just strings of bits, so one program can change an instruction in another just by, say, adding the appropriate number to convert one instruction to another.

This is a simple program to multiply two numbers together, input from the keyboard, and display them and their product on the screen. :-

```
Enter program at register 001
001   enter Zero in        Register 104                              go to 002
002   enter Zero in        Register 103                              go to 003
003   enter a one in       Register 102                              go to 004
004   input number         From keyboard      To Register 100        go to 005
005   input next number    From keyboard      To Register 101        go to 006
006   Add                  Contents of R100   To Register 103        go to 007
007   Add                  Contents of R101   To Register 104        go to 008
008   Subtract             Contents of R102   from Register 103      go to 009
009   If R103 is Zero Go to Register 010      otherwise              go to 007
010   Output to Screen     Register 100                              go to 011
011   Output to Screen     Space x space                             go to 012
012   Output to Screen     Register 101                              go to 013
013   Output to Screen     Space = space                             go to 014
014   Output to Screen     Register 104                         Exit program
```

If the numbers input from the keyboard are 12 and 20

This program will loop round the three instructions in registers 007, 008 and 009 adding 20 twelve times and then display   12 x 20 = 240   on the screen.

Parallel Programming: Managing Terminals

In this example the processing program is only three instructions long, however, applications can be many thousands of instructions, however this demonstrates the point that it would be possible to insert an interrupt instruction after instruction 007 which just transferred control to another program. After this program had been executed it would transfer control back to our program which would carry on and complete the multiplication.

If this system has a number of terminals connected, whenever a terminal is activated its reference number will be flagged up in a table, or stack, attached to the terminal's management program. The interrupt program might first check if any terminal had been activated, and if so input some characters from that terminal.

Because of the speed of the processor, characters from twelve terminals could have been input while our program was multiplying its two numbers. It is readily apparent that the amount of parallel work depends on the numbers being multiplied, thus, however time consuming the original program, the processor can be interleaving many other tasks.

All programs are designed so that no one rogue program can hog the processor.

Writing program using the register and instruction numbers is fraught with problems. Programming languages use stylised English words, then a 'compiler program' replaces the numbers with subroutines of machine code instructions that are implemented by the Processor.

Programs that find errors.

A trace program makes use of a very useful instruction that can deconstruct the components of an instruction. This 'collate' instructions, could for instance, list the five components of register 007 in five registers.

Thus a trace program could load our multiplication program as a data file and deconstruct each instruction one after another. It does two things. First it carries out the instruction under the control of the trace, then prints this program step. As the trace program operates, the three instructions 007, 008, and 009 would be printed out twelve times and register 100 could be seen to reduce by one, and register 101 increase by twenty on each occasion, exposing any error.

Versatility of Processors

Thus the simple mechanism is there to allow programs to be designed so that they can control other programs. A master library of application programs can select the ones required in the appropriate sequence. This is the basis of 'operating systems'.

Programs can be designed to edit programs in the light of the experience of running those programs. Thus computers can be designed to learn.

Numbers of similar application programs can be stored and the most appropriate selected to suit particular circumstances.

Programs can be interleaved so that all the peripheral equipment, which may operate relatively slowly, can be designed to operate at their most optimal speed. For instance, typical keyboard and screen systems for operators to input text can respond instantly, however fast the 'author' can type. Nevertheless, the computer may be running many other programs in the background.

As we have seen programs can trace programs to highlight errors.

From this simple ability to select the next instruction the processor is to execute, and thus 'interrupt' any program, the most complex of systems can be constructed.

Similarity to the Brain, Mind.

One of the great advances in human cognition was the ability to interrupt the inherited or conditioned reflex and evaluate alternative courses of action. We call it *thinking*. There is very little new in the universe.

2018 // Book Final // Appendices NEW // 032-H Program Writing program