Instruction Set of the Deus X

Shown below are the machine language instructions available on the Deus X,[*] our hypothetical machine. Each instruction has an operation code (which says what to do) and an operand (which generally says what to do it *to* (or *with*)). In some instructions, the operand itself is used in the operation. In others, the operation takes the operand as an *address* in memory, and use the data stored at that address to carry out the instruction.

Please do not be discouraged by the tedious, painstaking detail involved in machine-level programming. You only need to grasp the basic concepts; you will probably never have to produce something in machine language yourself.

Each instruction appears below with its name, its numeric op code, and its Deus X Assembly Language mnemonic.

Control Instructions

No-op (**0**) `NOP` — Do nothing; continue by executing the next instruction.

Halt (**8**) `HALT` — Stop execution of the program.

Arithmetic Instructions

Add (**1**) `ADD` — Add data stored at operand to the A register; store result back in reg. A.

Sub (**2**) `SUB` — Subtract data at operand from the A register; store result in reg. A.

Multiply (**3**) `MULT` — Multiply data at operand by register A (producing a 16-digit result, perhaps with leading zeros); store the first 8 digits of the result in register A and the last 8 digits in register B.

Divide (**4**) `DIV` — Divide a 16-digit number (whose first 8 digits are in reg. A and whose last 8 digits are in reg. B) by data at operand; store quotient in reg. A and remainder in reg. B.

Input/Output Instructions

Input (**5**) `IN` — Read an 80-column card (or input line) into 20 words of memory starting at operand.

Output (**6**) `OUT` — Print 30 words of memory starting at operand as a 120-character line.

Storage Instructions

Move (**9**) `MOVE` — Copy a block of memory elsewhere: Reg. A holds the number of words of memory to move; reg. B holds their starting location; the operand is the first word of the destination.

Load into A (**10**) `LDA` — Copy to reg. A the data at the address specified by the operand.

Load into B (**11**) `LDB` — Copy to reg. B the data at the address specified by the operand.

Store from A (**20**) `STA` — Store contents of reg. A to the address specified by the operand.

Store from B (**21**) `STB` — Store contents of reg. B to the address specified by the operand.

Increment A (**30**) `INCA` — Add the operand itself to reg. A.

Increment B (**31**) `INCB` — Add the operand itself to reg. B.

Enter into A (**40**) `ENTA` — Copy the operand itself to reg. A.

Enter into B (**41**) `ENTB` — Copy the operand itself to reg. B.

[*] The Deus X machine was named by David Smallberg.

Comparison Instructions

Compare reg. A (**50**) `CMPA` — Compare reg. A with data at operand and set the appropriate comparison indicator (Equal, Less Than, or Greater Than).

Compare reg. B (**51**) `CMPB` — Compare reg. B with data at operand and set the appropriate comparison indicator (Equal, Less Than, or Greater Than).

Branching Instructions

For each of the following instructions, the operand is used as the address in memory where the processor will find the next instruction to execute. This allows the machine to choose different paths through the program. Since the machine keeps track of which instruction to execute in the Program Counter, these instructions work by copying the operand into that register if the specified condition is met.

Jump (**7**) `JUMP` — Transfer control unconditionally.

Jump if Equal (**60**) `JE` — Transfer control if the Equal Indicator is on.

Jump if Not Equal (**61**) `JNE` — Transfer control if the Equal Indicator is off.

Jump if Greater Than (**62**) `JG` — Transfer control if the Greater Than Indicator is on.

Jump if Less Than (**63**) `JL` — Transfer control if the Less Than Indicator is on.