Version 2, March 2011. **CCS-M2X3**

EMIDEC 2400 instruction set and instruction times.

The following description comes from references 1 and 2 (see below).

The format of a 36-bit word when used as an instruction may be effectively visualised as follows:

Function	Modifier	Address A	*(spare)*	Address N
6	6	6	6	12

Notation for the EMIDEC 2400 instructions:

N	main memory address: locations 0 – 63 refer to the high-speed diode-capacitor store (also called the cache); locations 64 – 4095 refer to core storage.
n	contents of a main memory address before the operation.
n’	contents of a main memory address after the operation.
A	high-speed memory address (cache).
a	contents of a cache address before the operation.
a’	contents of a cache address after the operation.
M	cache address to be used as a modifier.
q	contents of cache address 16.
PC	program counter (called *control number* in the original EMIDEC 2400 manuals).

The significance of the addresses may vary with different types of instruction – for example when N signifies a constant (literal). The following guidelines apply. The complete list of functions, with their explanation and minimum/maximum execution times, is given later.

Simple arithmetic instructions (functions 0 – 3, 6 – 9):
These functions operate on the binary numbers a and n and return the result to address A and N. Negative numbers are held in two’s complement form.

Data-transfer instructions (functions 4, 5, 10 and 11):
These instructions transfer a brick or a multi-length word from one store to another. Functions 5 and 11 transfer one brick only. The instructions 4 and 10 are used to transfer alphanumeric data from one store to another. The number of bricks transferred is stored in the most significant character of the multi-length word. If the source address in these instructions holds a binary brick, only one brick is transferred.

Literal (constant number) instructions (functions 12 – 16)**:**
These functions were known as *Program Number Entry* in the original manual - see ref. 2). They treat N as a constant (literal), in the range (2^{12} – 1) to (1 – 2^{12}).

Multiplication instructions (functions 17, 18):
The result of the multiplication is stored in the two special cache (high-speed store) addresses 14 and 15, where address 14 holds the most-significant half of the result.

Division instructions (function 19 (with remainder) and 20 (with fractional quotient)):
The integral quotient in both instructions is stored in the special cache address 16. For instruction 19 an integral remainder is obtained and placed in address 17. For instruction 20 a fractional quotient is obtained and placed in address 17.

Logical instructions (functions 21, 22 and 23).
Instructions 22 and 23 are, respectively, the normal logical AND and OR operations. For instruction 21, the contents of cache address 16 (denoted by q) is used as a mask. The result of function 21 is that those bits of n which are specified by 1's in q are replaced by the corresponding bits in a.

Shift instructions (functions 35 – 39):
N specifies the number of places that a is to be shifted. Instructions 35 and 36 are 36-bit arithmetic shifts. Instructions 37 and 38 are 34-bit logical shifts. Instruction 39 is a 36-bit circular shift.

Magnetic tape instructions (functions 24 – 30):
For these instructions, a specifies the tape unit number and the buffer number to be used. Instructions 24 and 25 both read one block from tape into core store. Instructions 28 and 29 both write one block from core store to tape. For more information on different forms of reading and writing, refer to reference 2. Instructions 26, 27 and 29 respectively concern skipping forwards/backwards on tape and write/erasing. Again, for more information see ref. 2.

Comparison and indicator instructions (functions 32 – 34 and 42):
Instructions 32 – 34 compare a with, respectively: n in binary, n in left-justified alphanumeric data, n in right-justified alphanumeric data. The results of comparisons are recorded by setting appropriate values on bits 1 – 6 of the 16 special indicators in a *Conditions Register,* as listed below. Instruction 42 jumps to N if indicator A is on. The indicators in the *Conditions Register* are numbered as follows:

Indicator	*meaning if set*
0	overflow
1	n > a
2	n < a
3	n = a
4	n ≤ a
5	n ≥ a

6	n ≠ a
7	carry
8	parity error
9	parity error interrupt program in progress
10	non parity error interrupt program in progress
11	orientation of tape unit connected to buffer 1
12	orientation of tape unit connected to buffer 2
13	orientation of tape unit connected to buffer 3
14	orientation of tape unit connected to buffer 4
15	operators' console switch.

Switch-setting instructions (functions 40, 41):
The computer has 64 single-bit, addressable, switches which are physically part of the diode-capacitor cache. Instructions 40 and 41 cause, respectively, switch a to be turned off or on. Some of the 64 switches are assigned special meaning for the control of the magnetic tape buffers.

Jump instructions (functions 43 to 47; see also function 42 described previously):
See the main list given below for the actions of these control-transfer instructions.

Data conversion instructions (functions 48 – 51):
See the main list given below for the actions of these instructions.

Operators' console instruction (function 52):
As well as giving a program the ability to display general information, this function allows specific instructions to be given to the operator to rewind tape units, change reels of tape, etc. See also section M2/X2.

Stop instruction (function 31):
This causes the computer to halt. Functions numbered 53 to 63, which are unused (spare), also cause the machine to halt.

List of EMIDEC 2400 instructions and their execution times.
Instruction times are in two parts: *fetch* (called 'set-up' in the original documents) and *execute.* The *fetch* times depend upon whether an instruction resides in the main core store or in the fast diode-capacitor cache. Under certain circumstances an instruction in the core store may be fetched in parallel with the execution of the previous instruction. The *fetch* times for the three cases are as follows:

Instruction in main core store:	19 microseconds.
Instruction in fast cache:	11 microseconds.
Parallel fetching for an instruction in core store:	10 microseconds.

In addition to the *fetch* times, the minimum and maximum *execution* times, in microseconds, for each instruction are shown in the Table below. The times vary between minimum and maximum according to two factors: (a) the location of the operand(s); (b) the number of bricks, m, in alphanumeric operands.

Function	***Action***	***Minimum execution time***	***Maximum execution time***
0	a’ = a + n	14	16
1	a’ = a - n	14	16
2	a’ = n – a	14	16
3	a’ = - n	9	17
4	a’ = n *(word)*	8.5 x m	16.5 x m
5	a’ = n *(brick)*	8.5	16.5
6	n’ = n + a	14	21
7	n’ = n - a	14	21
8	n’ = a - n	14	21
9	n’ = - a	11	21
10	n’ = a *(word)*	8.5 x m	19 x m
11	n’ = a *(brick)*	8.5	19
12	a’ = a + N	9	9
13	a’ = a - N	9	9
14	a’ = N - a	9	9
15	a’ = - N	7	7
16	a’ = N	7	7
17	a x n	92	100
18	a x N	92	92
19	n / a *(remainder)*	33	290
20	n / a *(fractional quotient)*	39	440
21	n’ = (a AND q) OR (NOT q AND n)	24	26
22	q’ = n AND a	14	17
23	q’ = n OR a	14	17
24	Read one block	42	42
25	Read one block by records	42	42
26	Skip N blocks	29	42
27	Move tape to control point	42	42
28	Write one block	42	42
29	Write one block by records	42	42
30	Write erase	42	42
31	Stop	--	--
32	Compare a with n *(binary)*	16	18
33	Compare a with n *(left justified)*	21 x m	23 x m
34	Compare a with n *(right justified)*	16 x m	18 x m
35	a’ = a x 2^{-N} *(arithmetic shift down)*	11 + 2N	11 + 2N
36	a’ = a x 2^{+N} *(arithmetic shift up)*	11 + 2N	11 + 2N
37	a’ = a x 2^{-N} *(logical shift down)*	11 + 2N	11 + 2N

38	a’ = a x 2^{+N} *(logical shift up)*	11 + 2N	11 + 2N
39	a’ = a x 2^{+N} *(circular shift)*	11 + 2N	11 + 2N
40	Set switch A off	4	4
41	Set switch A on	4	4
42	Jump to N if indicator A is on	2 (off)	4 (on)
43	Jump to N if switch A is on and switch off	4 (off)	8 (on)
44	Jump to N if switch A is on	4 (off)	8 (on)
45	Jump to N; a’ = PC + 1	10	10
46	Jump to N if a ≠ 0; make a’ = a + 1	4, (a = 0)	9, (a≠ 0)
47	Jump to N if a ≠ 0; make a’ = a - 1	4, (a = 0)	9, (a≠ 0)
48	a’ = n *(decimal to binary conversion)*	61	408
49	a’ = n *(sterling to binary conversion)*	61	436
50	a’ = n *(binary to decimal conversion)*	137	343
51	a’ = n *(binary to sterling conversion)*	238	390
52	Operators’ console instruction	17	17

(Functions 53 – 63 are unassigned. If used by a programmer, they cause the same action as function 31, ie halting the program).

References.

1. Emidec 2400 Data Processing System: general description. Undated 8-page brochure issued by EMI Electronics Ltd. Computer Division, Hayes, Middlesex. (Printer’s reference code: ES17/1064/5M/360/BP).

2. Emidec 2400 Data Processing System: technical specification. Undated 10-page brochure issued by EMI Electronics Ltd. Computer Division, Hayes, Middlesex. See: http://www.vintage-icl-computers.com/icl44aaa