Summarized Instruction Repertory

This section is planned as a condensed description of the Orion Instruction Repertory, for quick reference. It is taken from the digitised version of the ORION Programming Manual prepared by Dr Malcolm Bigg, one of ORION 2 ‘s designers and to be found at :
http://malcolm.bigg.me.uk/Orion

First we give an overview of the ORION instruction set, in a reproduction of the original two-sided Ferranti summary sheet. For a full description of the detailed effects of the various instructions and their times see section 3 of the ORION Programming Manual quoted above.

The instruction sheets are followed by a description of ORION pseudo-registers, taken from
(see attached GIF file Order codes 2 }

00 $z' = x + y$
01 $z' = x - y$
02 $z' = y - x$
03 $z' = -y$
04 $z' = y$
05 $z' = x \& y$
06 $z' = x \vee y$
07 $z' = x \not\equiv y$

10 $z' = x + Y$
11 $z' = x - Y$
12 $z' = Y - x$
13 $z' = -Y$
14 $z' = Y$
15 $z' = x \& Y$
16 $z' = x \vee Y$
17 $z' = x \not\equiv Y$

20 $z' = x + \not{p}Y$
21 $z' = x - \not{p}Y$
22 $z' = \not{p}Y - x$
23 $z' = -\not{p}Y$
24 $z' = \not{p}Y$
25 $z' = x \& \not{p}Y$
26 $z' = x \vee \not{p}Y$
27 $z' = x \not\equiv \not{p}Y$

30 $z'_I = x_I \; y_I$
31 $z'_F = (x_F \; y_F)_r$
32 $z{:}' = xy$
33 $z{:}' = z{:} + xy$
34 $z' = xY$
35
36 Unassigned;
37 illegal

40* Unrounded: integral quotient to Z, remainder to $Z + 1$
41* Rounded: rounded integral quotient to Z
42* Rounded with d.l. mid-point quotient: integral part to Z, fractional part to $Z + 1$
43* Rounded: fractional quotient to Z
44* Unrounded with d.l. dividend: integral quotient to Z, remainder to $Z + 1$
45* Rounded with d.l. dividend; fractional quotient to Z
46
47 Unassigned; illegal.

Y is treated as signed

50 $z' = x.2^Y$
51 $z' = x.2^{-Y}$ — s.l. arithmetical
52 $z' = x$ up Y bits
53 $z' = x$ down Y bits — s.l. logical
54 $z{:}' = x{:}.2^Y$
55 $z{:}' = x{:}.2^{-Y}$ — d.l. arithmetical, result in standard form
56 $z{:}'_L = x{:}_L$ up Y bits
57 $z{:}'_L = x{:}_L$ down Y bits — d.l. logical

See other side for note on timing.

Jump to X if:-

60 $y = z$
61 $y \neq z$
62 $y > z$ ($z < y$)
63 $y \ngtr z$ ($z \geqslant y$)
64 $y < z$ ($z > y$)
65 $y \nless z$ ($z \leqslant y$)
66 $\not{p}Y = z$
67 $\not{p}Y \neq z$

For 2-address, put $z = 0$

Jump to X if:-

70 $Y = z_m$
71 $Y \neq z_m$
72 $Y > z_m$ ($z_m < Y$)
73 $Y \ngtr z_m$ ($z_m \geqslant Y$)
74 $Y < z_m$ ($z_m > Y$)
75* $Y \nless z_m$ ($z_m \leqslant Y$)
76* $\overline{y} \& z = 0$
77* $\overline{y} \& z \neq 0$

For 2-address, put $z = 0$ **(not 76,77)**

	3-address			**2-address**		
80	$z' = z + 1$	Jump	$z'_m = Y$	$y' = y + 1$	Jump	$y'_m = 0$
81	$z' = z + 1$	to	$z'_m \neq Y$	$y' = y + 1$	to	$y'_m \neq 0$
82	$z' = z - 1$	X	$z'_m = Y$	$y' = y - 1$	X	$y'_m = 0$
83	$z' = z - 1$	if	$z'_m \neq Y$	$y' = y - 1$	if	$y'_m \neq 0$
84	Jump to X		is in y	Jump to X		is in y
85	if char. Z		is not in y	if char. SP		is not in y
86	$z'_m = c + 1$, z'_s = OVR OVR′ = 0, jump to X			Enter S/R		$y'_m = c + 1$, y'_s = OVR, OVR′ = 0, jump to X
87	OVR′ = OVR v z_s Jump to $X + z_m$			Leave S/R		OVR′ = OVR v y_s Jump to $X + y_m$

90 $z'_G = x_G + y_G$
91 $z'_G = x_G - y_G$
92 $z'_G = y_G - x_G$
93 $z'_G = -y_G$
94 $z'_G = x_G . y_G$
95 $z'_G = x_G / y_G$
96 Unassigned; illegal.
97 z'_I = no. of places to standardize $x_G - y_G$.

100* Convert characters to binary
101* Convert binary to characters
102* $z'_G = (x_P + \nu).2^Y$, Y signed
103 $z'_P = x_G.2^Y$, Y signed
104* Convert card 2-character data
105
106 Unassigned; illegal.
107

110 $x' = (x \& \overline{y}) \vee (z \& y)$
111 $x' = (x \& \overline{Y}) \vee (z \& Y)$
112 $x' = (x \& \overline{\not{p}Y}) \vee (z \& \not{p}Y)$ — (3) Insert field of z into x or (2) Clear field of x ($z = 0^{48}$) — wherever mask (y, Y or $\not{p}Y$) has 1-bits.
113 Unassigned; illegal
114 (3) $z' = y$, $y' = x$ (2) $x' = y$, $y' = x$ (exchange)
115 Add l.s. 6 bits of Y to m.s. 6 bits of x; add rest of Y to x_m; end-around carry
116 Add X and Y to corresponding before any replacements Modify next
117 addresses in next instruction after therein instruction.

120* Count 1-bits
121* Cyclic shift
122* Table look-up
123* Insert (append) field
124* Find end-most 1-bit
125* Standardize unpacked floating-point number
126* Justify d.l. number
127 to 137 Unassigned; illegal.

140.N 0 DEVICE
142 FROM LENGTH — Peripheral Transfer
141.N 0 DRUMADDR
142 COREADDR LENGTH — Drum Transfer — List of Modes on other side
142 TO 0
142 FROM LENGTH — Internal Transfer
143 (3) Copy z into X to $X + Y - 1$
(2) Clear to zero X to $X + Y - 1$
144 Search table starting at X for first — $\omega \geqslant y$ (ascending)
145 — $\omega \leqslant y$ (descending)
146 — $\omega_u = y_u$ or $\omega_m = 0$
and set z' = address of ω
147 Unassigned; illegal.

Standard Macro-Instructions

1000 CHAP ADDR
Load CHAP, enter at ADDR

1001 CHAP
Load CHAP, continue

1002 CHAP ADDR1 ADDR2 ADDR3
Load part of CHAP, enter at ADDR3

1003 CHAP ADDR1 ADDR2
Load part of CHAP, continue

1086 :SUB/E1
Ensure copy of Library S/R :SUB/ is in current chapter, enter S/R at E1

All addresses are working-store addresses.

Pseudo-Registers

Y	Content and Notes
0	Zero
2	Local Civil Time
4	OVR (0 if clear, 1^{48} if set); cleared if used
6	OVR (0 if clear, 1^{48} if set)
8	$1^{24}\ 0^{24}$ (upper half-word mask)
10	$1^1\ 0^{47}$ (= -1.0)
12	$0^1\ 1^1\ 0^{46}$ (= ½)
14	$0^9\ 1^{15}\ 0^{24}$ (mask for X-address)
16	$0^{26}\ 1^6\ 0^{16}$ (mask for Z-address)
18	Handswitches

If Y is even, $\not{p}(Y{+}1) = \overline{\not{p}Y}$

Note: **3-address forms given. Unless stated otherwise,** get 2-address by reading x',$x{:}'$ for z', $z{:}'$ and putting $z = 0$.

*** Individual notes overleaf.**

F E R R A N T I L T D

ORION COMPUTER
SUMMARIZED PROGRAMMING INFORMATION

COMPUTER DEPARTMENT

LONDON COMPUTER CENTRE,
68 NEWMAN ST.,
LONDON, W.1.
Telephone: MUSeum 5040

Office and Works:
West Gorton,
Manchester 12
Telephone: East 1301

List CS 299A

40 $z_I' + (z_I^{*\prime}/y_I) = x_I/y_I;\quad 0 \leqslant z_I^{*\prime}/y_I < 1$

41 $z_I' = (x_I/y_I)_r$ or $(x_P/y_P)_r;\quad -\frac{1}{2} \leqslant (x/y) - z_I' < \frac{1}{2}$

42 $z{:}_M' = (x_I/y_I)_r$ or $(x_P/y_P)_r;\quad -\frac{1}{2}\epsilon \leqslant (x/y) - z{:}_M' < \frac{1}{2}\epsilon$

43 $z_P' = (x_P/y_P)_r$ or $(x_I/y_I)_r;\quad -\frac{1}{2}\epsilon \leqslant (x/y) - z_P' < \frac{1}{2}\epsilon$

44 $z_I' + (z_I^{*\prime}/y_I) = x{:}_I/y_I;\quad 0 \leqslant z_I^{*\prime}/y_I < 1$

45 $z_P' = (x{:}_P/y_P)_r$ or $(x{:}_M/y_I)_r;\quad -\frac{1}{2}\epsilon \leqslant (x/y) - z_P' < \frac{1}{2}\epsilon$

$$\left.\begin{array}{l}\textbf{50 to 53}\\ \textbf{54 to 57}\end{array}\right\}\left\{\begin{array}{l}\text{The}\\ \text{last}\end{array}\right\}\left\{\begin{array}{l}24\\ 48\end{array}\right\}\left\{\begin{array}{l}\text{places shifted take no time}\\ \text{if shift number} \geqslant \text{stated number.}\end{array}\right.$$

75 2-address is recommended unconditional jump.

76 2-address: Jump to X if $Y = 0$

77 2-address: Jump to X if $Y \neq 0$

100 $z' = y_7(y_6(\ldots(y_1(y_0 z + x_0) + x_1) + \ldots) + x_7$

$$\left.\begin{array}{l}\text{If m.s.}\\ \text{2 bits}\\ \text{of } y_i\\ \text{are}\end{array}\right\}\left\{\begin{array}{lll}00 & & \text{check that m.s. bit of } x_i \text{ is } 0\\ 01 & (+16) & \text{check m.s. 2 bits of } x_i \text{ are } 01\\ 10 & (+32) & \text{make no check}\\ 11 & (+48) & \text{treat } x_i \text{ as zero}\end{array}\right.$$

$$\textbf{101}\quad\left.\begin{array}{l}(3)\\ (2)\end{array}\right\}\left\{\begin{array}{l}x' = \text{characters resulting}\\ \text{from conversion of}\end{array}\right\}\left\{\begin{array}{l}z.\\ x.\end{array}\right\}\left\{\begin{array}{l}y^* \text{ contains radices}\\ \text{and } y \text{ their product}\end{array}\right.$$

$$\left.\begin{array}{l}\text{To convert}\\ \text{non-significant}\\ \text{zeros}\\ \text{as}\end{array}\right\}\left\{\begin{array}{l}\textbf{SP}\\ 0\ \text{(digit)}\\ \text{full stop}\\ \textbf{UC}\end{array}\right\}\text{add}\left\{\begin{array}{r}0\\ 16\\ 32\\ 48\end{array}\right\}\text{to radix}$$

102 $\nu = 0$ if OVR clear

$\nu = \pm 2$ if OVR set, sign of ν opposite

to that of x_P. OVR$'$ = 0 unless z_G' overflows

104 Convert 2-character data in x to 4 characters

according to code table starting in Y

(3) $z_u' = z_m$, z_m' = new characters

(2) $x_u' = 0$, x_m' = new characters

$$\textbf{120}\quad\text{If}\left\{\begin{array}{l}Y \geqslant 0\\ Y < 0\end{array}\right\}\left\{\begin{array}{l}z_m' = \text{number}\\ \text{of 1-bits in}\end{array}\right\}\left\{\begin{array}{l}\text{first}\\ \text{last}\end{array}\right\}\left\{\begin{array}{l}|Y| \text{ bits of } x.\ z_s' = \text{inverse}\\ \text{of next bit of } x\end{array}\right.$$

121 $z' = x$ shifted cyclically right Y bits (Y signed)

If $|Y| \geqslant 24$, last 24 places shifted take no time.

$$\textbf{122}\quad\left.\begin{array}{l}(3)\\ (2)\end{array}\right\}\left\{\begin{array}{l}x' = y \text{ shifted}\\ \text{cyclically left}\end{array}\right\}\left\{\begin{array}{l}Z\\ z_c\end{array}\right\}\text{characters.}$$

$$\textbf{123}\quad\left.\begin{array}{l}(3)\\ (2)\end{array}\right\}\left\{\begin{array}{l}\text{Leave unchanged}\\ \text{the first}\end{array}\right\}\left\{\begin{array}{l}Z\\ z_c\end{array}\right\}\left\{\begin{array}{l}\text{characters of } x{:}_L,\ \text{follow them}\\ \text{by all } y,\ \text{clear rest of } x{:}_L\end{array}\right.$$

$$\textbf{124}\quad(3)\ \text{If}\left\{\begin{array}{l}Y \geqslant 0\\ \\ Y < 0\end{array}\right\}\left\{\begin{array}{l}x' = x\\ \text{shifted}\\ \text{logically}\end{array}\right\}\left\{\begin{array}{l}\text{left}\\ \\ \text{right}\end{array}\right\}\text{until}\left\{\begin{array}{l}\text{left-most}\\ \\ \text{right-most}\end{array}\right\}\left\{\begin{array}{l}\text{1-bit shift}\\ \text{off but noted}\\ \text{more than }|\end{array}\right.$$

z' = shift number. If no 1-bit removed then $z_m' = |Y|$ and $z_s' = Y|$ places

(2) x' = shift number, etc.. x and its shifted form overwritten. : 1

125 (3) $x{:}_P' = (x{:}_P + \nu)2^m\ [x^* \geqslant 0];\quad z_I' = z_I - m;\quad$ OVR$'$ = 0

(a) $\nu = 0$ if OVR clear, $\nu = \pm 2$ if set (sign opposite

to that of x). OVR left clear.

(b) m is an integer such that

either (i) $\frac{1}{2} \leqslant x{:}_P' < 1$ and $-1 \leqslant m \leqslant Y$

or (ii) $-1 \leqslant x{:}_P' < -\frac{1}{2}$ and $-1 \leqslant m \leqslant Y$

or (iii) $-\frac{1}{2} \leqslant x{:}_P' < \frac{1}{2}$ and $m = Y$

(c) If $m < 0$, shifting down is unrounded;

only happens if $\nu \neq 0$ (i.e. OVR set on entry)

(2) Illegal.

126 (2) $x' + \epsilon y' = x + \epsilon y + \epsilon\nu;\quad y' \geqslant 0;\quad$ OVR$'$ = 0

$\nu = 0$ if OVR clear, $\nu = \pm 2$ if set (sign opposite

to that of y). OVR left clear unless x' overflows

(3) As (2) but result is $z' + \epsilon y'$ = justified form of $x + \epsilon y$.

MODES IN 140 AND 141

Y denotes Y-address in the 142-instruction

Paper Tape

1 Read 7-tr. up to NL, at most Y chars.
2 Read 7-tr., Y chars.
8 Read 5-tr., Y chars.
16 Disengage
21 Punch 7-tr. up to NL, at most Y chars.
22 Punch 7-tr., Y chars.
28 Punch 5-tr., Y chars.

Magnetic Tape

1 Read forward
2 Read backward
14 Rewind
21 Write, long gap
22 Write, short gap
28 Erase

Punched Cards

1 Read, IBM
2 Read, ICT
4 Read, BULL
7 Read, IBM and binary
8 Read, ICT and binary
11 Read, BULL and binary
14 Read, binary
16 Disengage
19 Read normal and interstage
21 Fill data buffer, punch (coded)
22 Fill data & code buffers, punch (binary)
26 Fill code buffer

Xeronic Printer

16 Disengage
21 Print up to NL, at most Y chars.
22 Print Y chars.

Line Printers

16 Disengage

$$\left.\begin{array}{l}21\\ 22\end{array}\right\}\left\{\begin{array}{l}\text{Fill data buffer}\\ \text{and print a line}\end{array}\right\}\begin{array}{l}\text{with full character set}\\ \text{with part character set}\end{array}$$

26 Fill code buffer

Drum

1 Read Y words from drum-store
21 Write Y words into drum-store

All Devices

13 Interrogate

SOME SPECIAL 150-INSTRUCTIONS

Summary of events associated with the various values of Z
in the instruction 150 X Y Z

1 Timing flag
2 Branch interlock

10 Stop (Halt or Suspend)
11 Abolish
12 Date and time
13 Message to Flexowriter
14 Question to Flexowriter
15 Read Directory

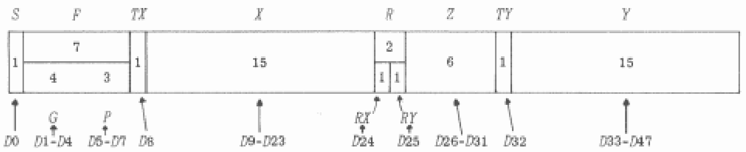
20 Set Monitoring style
21 Set peripheral incident
22 Set Monitoring peripheral
23 Return from private Monitoring
24 Start New Branch
25 Restore Branch Conditions

30 Reserve peripheral
31 Relinquish peripheral
32 Get geographical name
33 Load document
34 Get document if loaded
36 Change name of peripheral

$$\left.\begin{array}{ll}\textbf{40} & \text{Read block 0}\\ \textbf{41} & \text{Write block 0}\\ \textbf{42} & \text{Get current block number}\\ \textbf{43} & \text{Write non-sequential block}\end{array}\right\}\begin{array}{l}\text{Magnetic}\\ \text{tape}\end{array}$$

50 Chapter change
51 Load chapter of semi-built-in program
52 Change drum reservations
53 Change core reservations

Instruction Format



If X or Y is replaced in an internally modified instruction, the replacement occurs **before** the modification.

Pseudo-Registers

The following description is taken from section 2.6.0 of the ORION Programming Manual.

2.6.0

The pseudo-registers are a number of auxiliary registers containing useful constants or the state of switches inside or outside the machine. They can be read from but it is not possible to write to them. They are referred to by the instructions 20-27, 66, 67 and 112 (see sections 3.2, 3.6 and 3.11) - these instructions take the same time as the corresponding instructions (00-07, 60, 61 and 110) for ordinary registers. Pseudo-registers are available to all programs impartially - they are not subject to lockout or reservation checking. The Y^{th} pseudo-register is referred to as PY, and its contents as pY.

2.6.1 Contents of Pseudo-registers

The contents of even-numbered pseudo-registers from P0 to P18 are given below; for odd number registers p(2n+1) = ¬p(2n), i.e. the contents of each odd numbered pseudo-register is the inverse of the contents of the preceding even numbered register.

p0 = 0

p2 = Local civil time - see section 2.6.3 for details of the code.

p4 = Zero if overflow clear, all ones if overflow set.

Any instruction referring to p4 or p5 clears overflow.

p6 is as p4 except it is not cleared when used.

p8 = upper half word mask - i.e. 24 ones followed by 24 zeros.

p10 = -1.0 i.e. one followed by 47 zeros.

p12 = ½, i.e. a zero, a one followed by 46 zeros.

p14 = Mask for X address, i.e. 9 zeros, 15 ones, 24 zeros.

p16 = Mask for Z address, i.e. 26 zeros, 6 ones, 16 zeros.

p18 = Handswitches. The handswitches are a set of 48 keys which are provided for the use of the engineers; their use by programmers is not recommended except in very special circumstances - e.g. Pegasus simulator. It is the operator’s responsibility to ensure that only one program using the handswitches is in the machine at once.

2.6.2 Unallocated Pseudo-Registers

At present pseudo-registers 20-31 inclusive contain zero for even numbers and all ones for odd numbers. Pseudo-register numbers above 31 are taken modulo 32, i.e. only the least significant five bits are decoded. Programmers are strongly recommended not to use either of these facts as they may be changed if it is decided to add further pseudo-registers to the machine.

2.6.3 Local Civil Time

The digital clock in Orion is a 24-hour clock in hours, minutes and seconds. It is stored in a one out of n code, i.e. in each field which has n possible values one and only one of n bits is a 1. In each case the m.s. bit of the field represents 0, the next 1, the next 2 and so on. The fields are as follows:

D0	to	D2	tens of hours
D3	to	D12	hours
D13	to	D18	tens of minutes
D19	to	D28	minutes
D29	to	D34	tens of seconds
D35	to	D44	seconds
D45	to	D47	not used (always zero)

e.g. the time 17.08.23 is represented by 1-bits in digits:

1, 10, 13, 27, 31, 38

and 0-bits elsewhere.

Third Character			Fourth Character		
(**Twelves of Hours**)			(**Hours**)		
Value	**Binary Code**	**Decimal Code**	**Value**	**Binary Code**	**Decimal Code**
0	000000	0	0	000000	0
1	000001	1	1	000001	1
			2	000011	3
			3	000010	2
			4	000110	6
			5	000111	7
			6	001111	15
			7	001110	14
			8	001010	10
			9	001011	11
			10	001001	9

11	001000	8
0	001000	8
1	001001	9
2	001011	11
3	001010	10
4	001110	14
5	001111	15
6	000111	7
7	000110	6
8	000010	2
9	000011	3
10	000001	1
11	000000	0

Fifth Character
(**Tens of Minutes**)
Seventh Character
(**Tens of Seconds**)

Value	**Binary**	**Decimal**
	Code	**Code**
0	000000	0
1	000001	1
2	000011	3
3	000111	7
4	000101	5
5	000100	4
0	000100	4
1	000101	5
2	000111	7
3	000011	3
4	000001	1
5	000000	0

Sixth Character
(**Minutes**)
Eighth Character
(**Seconds**)

Value	**Binary**	**Decimal**
	Code	**Code**
0	000000	0
1	000001	1
2	000011	3
3	000111	7
4	000101	5
5	001101	13
6	001111	15
7	001011	11
8	001001	9
9	001000	8
0	001000	8
1	001001	9
2	001011	11
3	001111	15
4	001101	13
5	000101	5
6	000111	7
7	000011	3
8	000001	1
9	000000	0