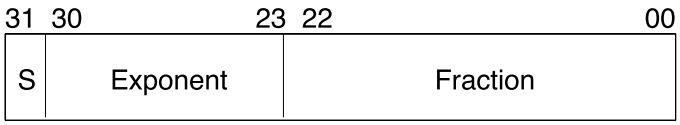
## 2.12 The IEEE formats

Read *Foundations of CS* pp200-201 section 4.12. There are in fact three formats and these differ subtly from the vax formats.

1. Single precision (32bit)
2. double precision (64bit)
3. quadruple precision (128bit)

We take a quick look at the single precision format:




Differs from the VAX G_Format in several significant ways:

- the fraction is interpretted to be of the form $1.M\ldots$

  $$\text{range is } 1.0 \leq f < 2.0$$

- exponent is *excess-127*

  $$(-1)^S 2^{(E-127)}(1.M)$$

  This is clearly different from the vax since

  $$= 1.M \times 2^{(E-127)} = 0.1M \times 2^{(E-126)}$$

  is not evenly biased.

- exception is when $E = 0$, then $M$ is interpreted without extra 1 (but the point is now shifted)

  $$(-1)^S 2^{-126}(0.M)$$

## 2.13 Precision & range

- It is useful to explore how the points are spaced on the number line.
  ...—1–0–1–2—4———8

**Exercise 2.13.1** *How many significant digits does the binary representation correspond to in Decimal?*

## 2.14 Alphanumeric representations /strings

### 2.14.1 ASCII

is the American Standard Code for Information Interchange (see table overleaf).

Note: the conversion to the character occurs at the perpheral device. The machine itself has no perception of what the bits stand for.

In ASCII, the *control characters* have special significance only in respect of the display (cursor control) of characters.

**Other obsolete codes**

1. Baudot
2. EBCDIC Extended BCD code 8-bit IBM code
3. Hollerith code on punched cards

### 2.14.2 Strings

A string has length and location.

Note that in the "Pascal" notation of ASCII strings the variable is generally a pointer to an array, and the length information appears in the first byte og the array. "C" uses the convention of null terminated strings, which does not limit the length of the string but precludes the use of the null as a string character.

A number of string formats are possible.

Finite runs of characters.
- Binary Strings,
- ASCII strings, up to 65535 chars
- Decimal strings
  - two sorts of decimal supported
  - numerical
  - packed decimal

| BITS b7 b6 b5 / b4 b3 b2 b1 | 0 0 0 | 0 0 1 | 0 1 0 | 0 1 1 | 1 0 0 | 1 0 1 | 1 1 0 | 1 1 1 |
|---|---|---|---|---|---|---|---|---|
| | CONTROL | | SYMBOLS NUMBERS | | UPPER CASE | | LOWER CASE | |
| 0 0 0 0 | 0 NUL 0 | 16 DLE 20 | 32 SP 40 | 48 0 60 | 64 @ 100 | 80 P 120 | 96 ‘ 140 | 112 p 160 |
| 0 0 0 1 | 1 SOH 1 | 17 DC1 21 | 33 ! 41 | 49 1 61 | 65 A 101 | 81 Q 121 | 97 a 141 | 113 q 161 |
| 0 0 1 0 | 2 STX 2 | 18 DC2 22 | 34 ” 42 | 50 2 62 | 66 B 102 | 82 R 122 | 98 b 142 | 114 r 162 |
| 0 0 1 1 | 3 ETX 3 | 19 DC3 23 | 35 # 43 | 51 3 63 | 67 C 103 | 83 S 123 | 99 c 143 | 115 s 163 |
| 0 1 0 0 | 4 EOT 4 | 20 DC4 24 | 36 $ 44 | 52 4 64 | 68 D 104 | 84 T 124 | 100 d 144 | 116 t 164 |
| 0 1 0 1 | 5 ENQ 5 | 21 NAK 25 | 37 % 45 | 53 5 65 | 69 E 105 | 85 U 125 | 101 e 145 | 117 u 165 |
| 0 1 1 0 | 6 ACK 6 | 22 SYN 26 | 38 & 46 | 54 6 66 | 70 F 106 | 86 V 126 | 102 f 146 | 118 v 166 |
| 0 1 1 1 | 7 BEL 7 | 23 ETB 27 | 39 ’ 47 | 55 7 67 | 71 G 107 | 87 W 127 | 103 g 147 | 119 w 167 |
| 1 0 0 0 | 8 BS 10 | 24 CAN 30 | 40 ( 50 | 56 8 70 | 72 H 110 | 88 X 130 | 104 h 150 | 120 x 170 |
| 1 0 0 1 | 9 HT 11 | 25 EM 31 | 41 ) 51 | 57 9 71 | 73 I 111 | 89 Y 131 | 105 i 151 | 121 y 171 |
| 1 0 1 0 | 10 LF 12 | 26 SUB 32 | 42 * 52 | 58 : 72 | 74 J 112 | 90 Z 132 | 106 j 152 | 122 z 172 |
| 1 0 1 1 | 11 VT 13 | 27 ESC 33 | 43 + 53 | 59 ; 73 | 75 K 113 | 91 [ 133 | 107 k 153 | 123 { 173 |
| 1 1 0 0 | 12 FF 14 | 28 FS 34 | 44 , 54 | 60 < 74 | 76 L 114 | 92 \ 134 | 108 l 154 | 124 \| 174 |
| 1 1 0 1 | 13 CR 15 | 29 GS 35 | 45 – 55 | 61 = 75 | 77 M 115 | 93 ] 135 | 109 m 155 | 125 } 175 |
| 1 1 1 0 | 14 SO 16 | 30 RS 36 | 46 . 56 | 62 > 76 | 78 N 116 | 94 ^ 136 | 110 n 156 | 126 ~ 176 |
| 1 1 1 1 | 15 SI 17 | 31 US 37 | 47 / 57 | 63 ? 77 | 79 O 117 | 95 _ 137 | 111 o 157 | 127 DEL 177 |

LEGEND: dec CHAR hex oct

Figure 2.1: American Standard Code for Information Interchange (ASCII)