

ARCHIEF

**stichting
mathematisch
centrum**



AFDELING INFORMATICA
(DEPARTMENT OF COMPUTER SCIENCE)

IW 247/83

DECEMBER

J.N. AKKERHUIS
TYPESETTING AND TROFF

kruislaan 413 1098 SJ amsterdam

BIBLIOTHEEK DE WETTERINGEN
AMSTERDAM

Printed at the Mathematical Centre, Kruislaan 413, Amsterdam, The Netherlands.

The Mathematical Centre, founded 11 February 1946, is a non-profit institution for the promotion of pure and applied mathematics and computer science. It is sponsored by the Netherlands Government through the Netherlands Organization for the Advancement of Pure Research (Z.W.O.).

1980 Mathematics subject classification: 68K05

1982 CR. Categories: H.4.4., I.7.2.

Copyright © 1983, Mathematisch Centrum, Amsterdam

Typesetting and TROFF

by

Jaap Akkerhuis

ABSTRACT

This report gives a short introduction to typesetting in general, and will discuss the UNIX[†] typesetting tools and it's recent changes.

KEY WORDS & PHRASES: Typesetting, textprocessing.

[†]UNIX is a Trademark of Bell Laboratories.



In The Beginning

The Greek word 'tupos' means both 'stamp' and the 'image' of the stamp; typesetting can be defined as writing by means of stamps. In the 11th century in China ceramic stamps were used to compile text, and in the 14th century in Korea books were printed by means of metal-cast types. Although there had been books printed in Western Europe from wooden blocks, the origin of typography is still obscure. The invention is most times ascribed to Laurens Janszoon Coster from Haarlem (The Netherlands) and/or Johann Gutenberg from Mainz (Germany). Both are supposed to have developed methods to cast separated metal type-slugs. Coster probably used wet sand moulds, Gutenberg certainly used metal casting-moulds; the last method became the generally accepted method. It is still unclear whether Gutenberg knew about the activities in Holland, some primitive prints have been preserved which are possibly older than Gutenberg's first productions.

The invention did not consist of the printing process itself, or of combining pages into a book. Also, the engraving of negative moulds for casting was well known in the trade of gold and silversmiths; Gutenberg was a goldsmith himself. What was new was the combined application of these methods with the necessary adaptations and augmentations which resulted in the art of type-casting. These methods and tools made it possible to combine the type-slugs into solid printing blocks. Also new was the adaptation of the well known wine, linen, or bookbinders press to be used as a suitable pressing medium, and an oily ink, which made it possible to make a clear two-sided print. Of course there was, due to the increasing wealth, also a market for books, the cumbersome handcopying of books by scribes could no longer fulfil the increasing demand.

Gutenberg turned his methods and life long research into perfection with his 42-line Bible from 1445, which is a piece of art which has hardly ever been equaled. His methods for engraving and casting of type, typesetting and printing of texts have hardly changed for 350 years. In the last 100 years some essential other methods have been introduced, and only in the last 25 years have Gutenberg's methods become hardly recognisable in the daily printing practice.

The moulds for a type are made by cutting out in a metal bar the required size and shape of the type, this is called the stamp. If the shape of the type includes a little opening, as for instance with 'a' or 'e', this hole is made with a punch. When the stamp is the right shape, it will be hardened. A copy is stamped in a copper plate, to form an intaglio or, recessed, a matrix. This shows the type in a readable form. After making corrections so that the impression is of equal depth, the matrix was placed in an adjustable mold, the type slug is cast off. The material used for the type should not be too costly, should not have a too high melting point, and should be hard enough to withstand a high impression. It consists mainly of lead (65 to 70 %), antimony (20 to 25%) for hardening and tin.

The face can be larger than the supporting shoulder, the leftover part is called the kern. When an 'f' with a kern is to be set against an 'l', the kern will easily break. To overcome this problem, ligatures are used. A nick in the slug is made, to enable the compositor to feel the front of type, so that upside-down placement of the slug can easily be avoided.

Hot Metal Typesetting

The first automation that took place was around 1860 in the type setting machines. The real first big change was the invention of the pantographic engraving machine by Linn Boyd Benton, which eased the making of the matrices. In the 1880's Otto Mergenthaler introduced a line setting machine, for which a patent was granted in 1885. The matrices consisted of thin copper plates, in the side of which the one or two typefaces were stamped. Hitting keys on the keyboard made the matrices come from the magazine and be ordered in one line. When the spacebar was hit, a wedge-shaped spaceband was brought in position (actually, the spaceband was made from two wedge shaped pieces of metal. They taper in opposite directions, so that the outside faces were always parallel). When there were words enough to nearly fill a line, the spacebands were driven up mechanically, just enough to fill the line. From the finished line a slug was cast, and the matrices were returned again to the magazine. Machines based on this principle are called linecasters, since there were no types, but

completely assembled lines cast off.

In the early 1930's a new method of keyboarding for setting type was introduced by Walter Morey. This method is known as TTS, the initials stands for 'teletypesetting'. The most practical application for TTS is what we call the 'wire service'. The main improvement was that stories could be distributed over a telegraph or telephone in justified form, without the necessity of re-keyboarding. The actual type setting was mostly done on a linecaster. This method is more suited to straight text than for complex work.

Tolbert Lanston did not only invent an adjustable horseshoe, a mail bag lock and a adding machine, but also the Monotype method of composition. A method was patented in 1887 and a first prototype displayed at the Columbian exposition in Chicago. When four of these machines were shipped to England, they were spotted by the Earl of Dunraven, which lead to the founding of the Lanston Monotype Corporation in England in 1897. The design is inspired by the Hollerith Tabulator. The machine consists of two parts, a calculating and punching keyboard device, which delivers a 31-hole wide papertape, and a casting device. Although the first 'casting' device was a die-stamping device, which reproduced single types from cold strips of type high lead, later it was modified to become a hot metal caster. The prepared papertape is read backwards in the caster, to do the final calculations for justification, then the slugs are cast off from the matrices, which were organized in a matrix case, consisting of 132 or 225 different matrices. Some interesting features come from this approach - since every the composition is done with type-slugs, just like handsetting, it is always possible to make small improvements by hand; because the type-slug is cast off after the justification process, it is also easy to play with the inter-letter space, without the need to replace the matrices. So this introduced more flexibility in the composition then ever before.

Phototypesetting

The introduction of phototypesetters was stimulated by the limited speed of the hot-metal process and, more importantly, the introduction of offset printing technology. For offset printing a sharp print is made from the complete composed page, which is photographed and etched into a zinc plate. From this plate the (ink) image is transferred to a (rubber) roll, after which the image is put on paper.

The first usable phototypesetter appeared in 1946, the (Harris) Intertype. This machine was more or less based on the Linotype. It had a magazine to hold matrices, which were brought out when necessary, and photographed on a film. The Monophoto of 1955, based on the Rototype (1949), was, not surprisingly, based on the Monotype. It had a film matrix case, from which the type was photographed on film.

The first of the modern phototypesetters was the Photon 200. According to legend, the principle was invented by two engineers who were doing research on the timing of airplane propellers by means of light strobes. When they wanted to publish the result of the study they were more or less appalled by the inconvenient and old-fashioned typesetting technology. So not surprisingly, the Photon 200 had a fast spinning disk, which contained the images of the 'types'. While the disk was continuously spinning, the types were photographed by a Xenon flashlamp on the film. The first prototype was finished in 1948, while production started around 1951.

The C/A/T, well known by UNIX-hackers, made by Graphic Systems was also designed on this principle. It has a fixed Xenon light source and filmstrips mounted on a rotating drum, 4 for the C/A/T-4 and 8 for the C/A/T-8. The filmstrips could officially contain 102 characters. There may have been machines in existence which would allow up to 108, and from a Technical Engineering Note it can be concluded that a maximum of 127 characters could be addressed in theory. This feature has never been used, addressing a non existing character would just 'hang' the machine. It was possible to mount new fonts by changing filmstrips, and (old) TROFF used this. For example, when the non-mounted font CW on position 3 was requested, TROFF loaded in a different width table, stopped after printing the string `Mount font CW on 3` and then waited for a linefeed before continuation.

After the light passed the filmstrip, it passed one of the nine lenses which were on a lens turret. The light was then fed into a fibre optic tube. The end of the tube rests on a carriage, which took care for the horizontal movement. Since the size of the tube limited the image to 18 points, another lens could be inserted between the film plane and the end of the tube. This extra lens introduced an escapement (horizontal movement) of 55 units. So this is also the explanation of the DBL or Double motion indicator and the actions performed by TROFF. The speed of the C/A/T is around 50 char/sec, but changing the lenses is quite time consuming; this explains the -p option of TROFF. TROFF does also vital optimisation, by printing an output line boustrophidionically sorted, printing one pointsize for every pass.

Although phototypesetting has a big advantage in speed, and is more suited to modern printing technology, a drawback is some loss in quality. With hot-metal composition, all the types in all the different sizes were different, even for the same typeface. This is needed, since the appearance of a type looks different at different sizes. A big sized type will look too heavy (black), when its reduced, while a small size would look too light. Also, some ink was pushed from underneath the slug, making the small details larger. This was of course more noticeable on the smaller pointsizes. With a phototypesetter, small details tend to get smaller due to light passing around them. Since the types sizes are enlarged or reduced, in photo typesetters by lenses from one single image, the original shape of the type has to be a compromise for all of the different sizes.

Cathode Ray Tubes (CRTs) first appeared in photosetters as 'index' tubes and 'printing' tubes. Although it is a bit beyond the scope of this paper to discuss the details of these techniques, basically the index tube scanned the image from a mat of matrices. The light was sent to photomultiplier tubes, and the electrical signal was then sent to another CRT, whose light made the image on the paper. Potentially this is a powerful system, it is possible to mix signals from, for instance a (colour) video source with the text information. This method is used on some (very expensive) machines. As you can see, this type of typesetter uses a combined process of scanning and 'printing'. The escapement is done by moving mirrors, or simply moving the complete CRT along the film. Some typesetters also use a big CRT, on which the complete line is output. The leading (vertical displacement), is done by moving the film. In the modern CRT typesetters, the scanning process is separated from the printing process. The digital image is stored on a magnetic medium, like f.i. a floppy disk. The source of the image is mostly still a scanned image. To have the 'image of type' in digital form allows a lot of electronic manipulation. Types can be electronically slanted, widened and enlarged, although the original image is left unchanged. The problem mentioned of compromising the shape of type for different sizes can be overcome by storing multiple images on disk for different pointsizes, which is done on certain high quality CRT typesetters. This might be visible to the user (like on the Harris) and which can then be quite a nuisance. The scanning process is quite often not a trivial process. Often the digitized image has to be processed to compensate for errors in the optical path introduced by both the scanning and printing processes.

Since typefaces are copyrighted, a lot of manufacturers do not give out details about the stored information (in the Lynotron the DES-chip for is used for the protection of the data), this copyright can be protected. Typesetting firms say they incur large costs making good, new typefaces, but often the faces look quite close to existing and popular ones. Of course, selling typefaces is also a source of income for the manufacturer of the typesetter. Some typesetting firms sell scanners as well, to let the customer make his own logo's etc. The quality of the type produced by CRT machines can be good, although the basic resolution can be a problem. You can easily see this if you look with a magnifying glass at this text (the characters are built from dots. These may not be visible, since the printing process tends to smooth this). Also, increased flexibility introduced by the CRT principle is inviting people to produce the most horrible types. Most types are based on old ones, there are hardly any new types designed, which use the new features introduced by the CRT technology and try to hide the deficiencies.

Legibility

Printed matter is a communication medium, which allows communication with hardly any technology needed at the receiving side. Also a bare minimum needed at the production side. It is sometimes surprising to see the quality of the products made with only a primitive press at hand, for instance with a silkscreen technique. The main goal of typography is legibility.

Legibility can not be defined in absolute terms, research on this always fails to produce final answers. Legibility is not showing off the capabilities you have on hand - although it may look impressive to write righthand adjusted letters, it is mostly a waste of computer time. For instance TROFF and TBL users mostly put boxes around the tables, which are in most cases unnecessary. It is worse when tbl is not tuned to the typesetter or other output device, which is the case on our system at the moment (careful users hide this fact by painting the overlap of the lines away). In general, it is very disappointing to see the horrible results produced by people who have a powerful tool like a typesetter at hand and want to show it. To have your message transported and understood is the goal of writing. Typography may help in selling products, to draw attention to posters, but for plain reading, it should be unnoticeable, just like film music is for the most part only illustrating the drama which takes place on the screen. This rule was noted by Gutenberg from the start, the types he used in his products were tailored towards the then accepted way of writing for reproduction.

In general, discussions on whether text should be (right)-justified or not should be considered of the religious type, a general rule can not be given. One argument in favour of left- and right-justified text is the observation that people tend to read in a boustrophidonical (zig-zagging) manner. A trained reader scans a text, and can read right to left as long there is not too much problem in understanding the content (or context). Unjustified text does not force a reader's eyes to a pattern, but when used wrongly can disturb and confuse the reader. Lines which differ more than 25 percent in length are in general not nice to read. This can be avoided by making 'red zones' at the right margin, within which the lines should end. This feature is, alas, lacking in TROFF. Some formatting systems have a maximum length of the stretched paddable wordspace. When more space is needed to justify a line, extra space is added between the letters to force justification. This is a bad method when it becomes visible, since it draws too much attention to the offending line. Adding inter-letter space should only be used for special cases, for instance to emphasise a word. Means to control the inter-letter space are not provided by TROFF.

Using sans-serif typefaces may look modern, but surely has it's drawbacks. A discussion on this topic is really too religious to be within the scope of this paper.

The TROFF Programs

Around 1964 at MIT RUNOFF appeared, of which TROFF is a descendant. The first implementation of TROFF was known as ROFF, which allowed simple formatting on lineprinters (lineprinters can be considered as a primitive linecaster-like typesetter). Later ROFF was completely changed and appeared around 1973 as NROFF (New ROFF). The output device for nroff was basically the well-known Teletype 37. It also gained a cousin called troff (typesetter whichROFF) as output device. These programs were all written in PDP-11 assembler. In 1975 NROFF was recoded in C and capabilities for different types of terminals were added. TROFF now differs from NROFF in just two source files (t6.c & t10.c), some #define's, and some routines which are dummy routines in NROFF. The structure of the code is still similar to the assembler code, which makes the programs only maintainable by masochists. In 1979 Bell Labs acquired a new phototypesetter, which should have been a good opportunity to replace TROFF with something better, but no one could come up with a better design, so Brian Kernigham started to modify TROFF so it would run henceforth without change on a variety of typesetters. This package is now known as typesetter independent TROFF and is usually called DITROFF.

In 1981, the Mathematical Center obtained it's own typesetter, a Harris 7450. This machine is connected via a serial line to a portselector so it can be shared between various machines. In order to get

this machine running, a filter was hacked which took the TROFF binary output code, specific for the C/A/T and translated in to yet another binary code for the Harris. Actually it is two filters, one for the interpretation of the C/A/T code, and another one which maintains the protocol with the Harris, which protocol is arcane beyond description. Splitting this into two filters made it possible to have other programs, such as TeX, DITROFF and a plot package produce Harris code without the need to know about the ghastly protocol. Just as we were about to hack TROFF to let it use the full capabilities of the Harris, we learned of Kernighan's work and decided to wait for it. Although it took some time before DITROFF finally arrived, it took about four weeks to have the package running.

TROFF was implemented before C gained the additions of TYPEDEF's and STATIC variables. The first thing that Kernighan did was to make the code more legible - one can argue whether he succeeded or not. Most of his changes are commented to some extent in the code, which makes life a bit easier. A pencil annotated listing of TROFF, proved to be very handy; making these notes, some bugs were discovered which appeared in DITROFF again.

TROFF Internals

Within TROFF objects are passed around as 16 bit quantities. There are two fundamental objects, printable characters and motions. An object looks like this:

1	4	2	1	8
m	s	f	z	c

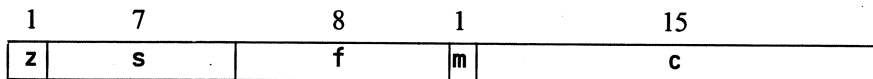
If bit *m* is set, the object is a motion, otherwise it is something to be printed. The *s* field is then an index into a table of legal sizes. The *f* field gives the number of the font, or to be more accurate, the filmstrip mounted in the C/A/T on which the character is supposed to be. The *z* field is the zero motion indicator (i.e. no space after printing) and *c* is the character; it is actually an index into a table of C/A/T-codes which is output by TROFF. If the input character is an ASCII character, its numerical value is used as an index into the code table, which contains the C/A/T flash code for the type. The index for special characters, like the hyphen (\hy, index is 0200) is looked up as soon as the sequence for a special character is recognised in the input. The code in the table itself consists of nearly legal 'flash codes' for the C/A/T. If the code has the bit 0200 set, the character resides on the 'Special' font strip. Since *ℓ* is an ascii char its index in the fontable is 0173, but the number in the code table is 0332. So, in the code generating phase of TROFF, this is captured, and action to address the right fontstrip is taken accordingly. If the 'Special' fontstrip isn't mounted, some white space will take the place of the type. Special characters, introduced by the \ℓ-sequence did not necessarily need to be on the 'Special' font and vice versa. So, although the hyphen (\hy or -), has the index 0200, its code is 040 and it was supposed to be available on every 'normal' font. Furthermore, if *s* is less than octal 040 or greater than octal 370, the character is an encoded function like a leader (001), tab (011), the undocumented mark current place in output line (\j 0374) function, the documented \x'n', (0375 or 0376) or a special character like \ℓ (035). There are myriads of places where Osanna used certain numbers for special cases and since these are not **#defined** and the octal as well as the decimal values are used, such as (32 and 040 f.i.), the reader of the source text can become very easily confused. The size of the character isn't encoded, it is calculated when necessary and the code to do the motion for the character is output, whenever possible optimised with other movements, after such a calculation. The motions are encoded as:

1	1	1	13
m	v	n	d

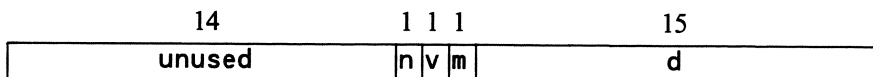
As said above, the *m* bit is 1 for a motion, *v* is 1 for a vertical motion and *n* is one for a negative motion. This leaves 13 bits for the magnitude, with a maximum of 8191. This is sufficient for the C/A/T, which has a resolution of 432 units per inch, and will mount up to 19 inches. The Harris has 1445.4 units per inch which will only give 5.6 inch, while the maximum width of the Harris is 68 Pica

or 11.2 inches.

In DITROFF the objects are widened to 32-bit quantities. The current representation of a character is:



and for a motion:



With this representation 225 fonts can be addressed, up to 32767 different characters in a font and 127 point sizes. The actual maximum number of characters is 200 and due to an implementation detail, limited to 255. Values of the *c* field are still used for special internal functions or for f.i. *\n*, *\t* etc. The maximal motion on a Harris is now 20.8 inches, which is sufficient on even an expanded Harris (100 Pica). When typesetters with higher resolution come on the market, it will likely not be enough. All the fields in the internal objects can now be accessed by macro's, so raising the maximal motion to 29 bits, (and swapping the *z* and *m* fields for the character representation) can now probably be easily done. Note that the objects described above are internal representations, and are never output by ditroff (there is one exception, some bugs in ditroff can cause some of the internal objects to be output, with the *-a* option).

Dynamic Machine Parameters

Every time DITROFF is run, it loads tables describing the capabilities of the required typesetter. It uses the commandline *-T* flag to get the name of the typesetter, as in *NROFF*. The tables consist of two parts, one part describing the typesetter, the *DESC* file, and one or more describing the fonts used. Each of these files has the same name as the font used. Here is the table for the C/A/T Roman font, the *R* file:

```
# Graphic Systems C/A/T-4
res 432
hor 1
ver 3
unitwidth 6
sizes 6 7 8 9 10 11 12 14 16 18 20 22 24 28 36 0
fonts 4 R I B S
charset
\l \^ \- \_
hy bu sq em ru 14 12 34 mi fi fl ff Fi FL de dg sc fm aa ga
ul sl *a *b *g *d *e *z *y *h *i *k *l *m *n *c *o *p *r *s
*t *u *f *x *q *w *A *B *G *D *E *Z *Y *H *I *K *L *M *N *C
*0 *P *R *S *T *U *F *X *Q *W sr ts rn >= <= == ~= ap != ->
<- ua da eq mu di +- cu ca sb sp ib ip if pd gr no is pt es
mo pl rg co br ct dd rh lh ** bs or ci lt lb rt rb lk rk bv
lf rf lc rc
```

A *#* introduces a comment. The machine resolution is *res* units per inch. The minimum number of machine units that it is possible to move are given by *hor* and *ver*. The point size at which the character widths map directly into machine units is given by *unitwidth*. The list of *sizes* is the set of legal point sizes, terminated by a zero. The number and names of the default set of *fonts* are given. There is an optional *paperwidth* keyword, with an argument which gives the width of the paper in basic units, overriding the 7 $\frac{3}{4}$ ". Undocumented keywords are, *paperlength*, *spare1* and *spare2*. The set of legitimate character names (names of the form *\(xx*, including some special

cases like \l) are introduced by the keyword **charset**.

There is one file per font, which lists the properties of a font. Here is a part of Times Roman for the C/A/T:

```
# Times Roman for C/A/T-4
name R
internalname 1
ligatures ff fi fl ffi ffl 0
charset
\l 6 0 0
\^ 3 0 0
a 17 0 025
b 20 2 012
c 16 0 027
d 20 2 011
e 18 0 031
f 13 2 014
...
! 12 2 0145
& 28 2 050
( 16 2 0132
) 16 2 0133
* 16 0 0122
+ 36 0 0143
, 12 0 047
hy 13 0 040 hyphen
- " =hy
\~ 36 0 0123
. 10 0 044
de 15 0 0136 degree
dg 20 0 0137 dagger
fm 8 0 0150
rg 20 0 0141
co 20 0 0153
ct 19 0 0127
...
```

The name is the external TROFF name, one or two characters, as used in the **.ft** and **\f** commands. The internal name is not needed in DITROFF but used by the postprocessor. **Ligatures** give the ligatures on the fonts, if any; only the ligatures in the form as listed above are scanned for in the input of DITROFF. There is an optional keyword **special** to denote a special font, a font that has to be searched when a character is not on the regular font. There is no limit on the number of special fonts, but they should be listed last in the fonts part of the DESC file. The first column is the character name, the second the width in machine units when the pointsize is unitwidth. If the name is a single character, it is taken to be the ASCII character, otherwise it is of the form **\(xx** (there are some historic exceptions like **\~**). The third column gives what is called the kerning information. This information is very primitive, and can not be compared to what is usually understood by kerning. A 1 means that the character has an descender, a 2 an ascender and 3 is used for both. A width in the form of a " indicates that the character is a synonym for the immediately preceding character. The fourth column is the actual typesetter code required to print it. The rest of the line is ignored and can be used for comments. These tables are in ASCII format to allow easy editing, and compiled in binary form by a separate program **makedev**. The original **makedev** has been improved a lot. It will now check that the list of sizes is terminated properly (it used to just generate a core dump), whether the maximum number of funny character names will not be exceeded, or all of the funny

names are actually used in one font, and whether characters in a font are not redefined. These added checks improve the consistency of the tables. A number of almost incomprehensible actions of DITROFF when loaded with faulty tables are thus avoided. The tables are not only read in by DITROFF but also by the postprocessor. Although it makes the tables larger than necessary, it simplifies the maintenance of the tables. DITROFF just skips the information that it doesn't need. A program `dumpdev` has been written which restores the original ASCII file.

Although these tables make it possible to use different typesetters, some enhancements had to be made for the Harris typesetter. For the sans-serif fonts on the Harris, there are no italic fonts, they have to be made by giving a slanting command, which can slant the images of types by 9 12 or 15°. So the keyword `slant` can now be used in the font tables. Also the notion of different fonts is not as one would expect. Each font on the Harris can have at most 128 characters, but some logical fonts have different physical fonts. For instance, the roman and italic ligatures of Times Roman are both on one physical font. This introduced the `fonttab` keyword in the font table, which denotes a fifth field giving the number of the internal Harris font on which the symbol resides. Also the `internalname` loses its function. The `DESC` file for the Harris looks like:

```
#
# Harris 7500 series
#
# The true resolution is 1445.4000 goobies (half decipoints) an inch
#
res 1445
hor 1
vert 1
unitwidth 10
sizes 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 0
#
fonts 10 R I B BI VL vL V v S Sm
#
# The complete charset of older versions of troff for a start.
# And more...
#
charset
\l \^ \- \_
hy bu sq rü em 12 14 34 13 23 18 38 58 78 ff fi fl Fi FL de
dg fm ct rg co pl mi eq ** sc aa ga ul sl *a *b *g *d *e *z
*y *h *i *k *l *m *n *c *o *p *r *s *t *u *f *x *q *w *A
*B *G *D *E *Z *Y *H *I *K *L *M *N *C *O *P *R *S *T *U *F
*X *Q *W sr rn >= <= == ~= ap != -> <- ua da mu di +- cu ca
sb sp ib ip if pd gr no is pt es mo br dd rh lh or ci lt lb
rt rb lk rk bv lf rf lc rc
-h
AE OE ae oe a^ a. ad ab ac a~ ah a, ce ho "a -D /O /L -d /o
/l .i .j r! r? sz Fo Fc fo fc
oA oE c+ c. c* pp nm as ic ll cp ll Dr <> DL c> => =< c< |>
cs =~ ~~ Cl Cr ~> l- l= an Ah Vh Gl bl OR AN
>> << cn nc D> BP
```

The tables for a font looks like:

```
#
# Vega light italics for Harris 7400 (italics by slanting)
#
# table of font 1418 with master 14 and unitwidth 10
# Note: ascender and descender info generated automatically
#
name vl
slant 12
fonttab
internalname 6
ligatures ff fi fl ffi ffl 0
charset
\| 33 0 0 0 spacewidth is 66 decip.
\^ 17 0 0 0 this isn't really a quarter space
@ 166 3 01 1418 AT SIGN
dg 111 3 04 1418 DAGGER
de 111 2 05 1418 DEGREE
J 69 3 010 1418 CLO BRAC
= 166 0 011 1418 EQUALS CEN
ct 111 2 014 1418 CENT SGN
fm 55 2 015 1418 MINUTE S
13 166 2 017 1418 1/3
23 166 2 020 1418 2/3
[ 69 3 023 1418 OPN BRAC
" 111 2 024 1418 SEC SIGN
+ 166 2 025 1418 PLUS BA
# 166 2 031 1418 NO SIGN BA
...
A 130 2 0101 1418 UC A
B 130 2 0102 1418 UC B
C 144 2 0103 1418 UC C
D 136 2 0104 1418 UC D
E 119 2 0105 1418 UC E
F 111 2 0106 1418 UC F
G 150 2 0107 1418 UC G
H 136 2 0110 1418 UC H
...
AE 194 2 01 4418 UC DIPH
OE 225 2 02 4418 UC OE DIP
ae 175 2 03 4418 LC DIPH
oe 186 2 04 4418 LC OE DIP
ff 111 2 05 4418 LC FF
fi 100 2 06 4418 LC FI
fl 100 2 07 4418 LC FL
Fi 155 2 010 4418 LC FFI
Fl 155 2 011 4418 LC FFL
aa 0 2 012 4418 LC ACC ACUTE
ga 0 2 013 4418 LC ACC GRAVE
a^ 0 2 014 4418 LC ACC CIRCUM
ab 0 2 016 4418 LC ACC BREVE
...
```

Output Language

The output of DITROFF is ASCII, so it is possible to use standard UNIX tools, in case it ever has to be processed. The output language is quite simple:

```

sn      size in points
fn      font as number from 1 to n
cx      ASCII character x
Cxy     character \(\(xy; terminate xy by white space
Hn      go to absolute horizontal position n. (n > 0)
Vn      go to absolute vertical position n (down is positive)
hn      go n units horizontally (to the right; n > 0)
vn      go n units vertically (down; n > 0)
nmc     move right nn, then print c (nn is exactly 2 digits!)
nb a    end of line (information only -- no action needed)
        b = space before line, a = after
w       paddable word space (information only -- no action needed)
pn      new page n begins -- set V to 0
x ... \n device control functions
D ... \n drawing functions (graphics)

```

The *nmc* encoding shrinks the output file about 35% and run time 15%.

The device control and graphics commands can be expanded as needed, and are currently:

```

x i      init
x T s    name of typesetter is s
x r n h v resolution is n/inch, h = minimum horizontal motion, v = min vert
x p      pause (can restart)
x s      stop -- done forever
x t      generate trailer
x f n s  font position n contains font s
x H n    set character height to n
x S n    set slant to n

```

Subcommands such as “i” are often spelled out like “stop”. The drawing functions are:

```

DL dh dv  draw line from current position by dh dv
Dc d      draw circle of diameter d with left side here
De d1 d2  draw ellipse of diameters d1 d2
Da dh1 dv1 dh2 dv2
           draw arc from current position to dh1+dh2 dv1+dv2,
           center at dh1 dv1 from current position
D~ dh1 dv1 dh2 dv2 ...
           draw B-spline from current position to dh1 dv1,
           then to dh2 dv2, then to ...

```

Everywhere, *dh* and *dv* are increments on the current horizontal and vertical positions, and, as usual in the typographic world, down and right positive. Blanks, tabs and newlines may, and sometimes have to, be used to prevent confusion, as separators. As an illustration, from the input

```

types in
.sp 2
.ps 20
.ft V1
print

```

the output using -Tcat is:

```

x T cat
x res 432 1 3
x init
x font 1 R
x font 2 I
x font 3 B
x font 4 S
V0
p1
s10
f1
H416
V72
ct
20y30p32e30sw45i17nn72 0
x font 0 VL
f1
H416
f0
s20
V288
cp
73r47i33n73tn72 0
x trailer
V4752
x stop

```

And for the Harris the output is:

```

x T har
x res 1445 1 1
x init
x font 1 R
x font 2 I
x font 3 B
x font 4 BI
x font 5 VL
x font 6 vL
x font 7 V
x font 8 v
x font 9 S
x font 10 Sm
V0
p1
s10
f1
H1392
V240
ct
66yh100cp
h111ce
88swh143ci
55nn240 0
H1392

```

```

f5
s20
V960
cp
h222cr
h138ci
88nh222ct
n240 0
x trailer
V15895
x stop

```

Note that the figures in the output has nearly tripled, and also that the font VL is mounted by default on the Harris.

This is what the true output looks like:

types in

print

New Things



As you may have noticed, it is possible to use an arbitrary number of fonts on one page. If the font requested with the `.ft` request is not already mounted, it will be loaded into the otherwise inaccessible position 0, which is called the font cache. The font remains in or will be mounted again in position 0 when the line is printed. Since it is not possible to access this position directly, or with the request, it is likely to cause problems. So if you use a font regularly, it is better to mount it with the `.fp` request. There are some plans to change this. According to the description of DITROFF, the `.fp` command takes an optional third argument for the directory to load the fonts from, but according to the source it does not. Fonts with two character names can also be used with the `\f` command, addressing them in the style used to refer to two character strings and registers, i.e by `\f(xx`.

A new facility is the addition of graphics commands:

```

\D'l dh dv'    draw line from current position by dh, dv
\D'c d'        draw circle of diameter d with left side at current position
\D'e d1 d2'    draw ellipse of diameters d1 d2
\D'a dh1 dv1 dh2 dv2'
                draw arc from current position to dh1+dh2 dv1+dv2,
                with center at dh1 dv1 from current position
\D'~ dh1 dv1 dh2 dv2 ...'
                draw B-spline from current position by dh1 dv1
                then by dh2 dv2, then by dh2, dv2, then ...

```

For example, the input `\D'e0.2i 0.1i'` draws the ellipse , and the input `\D'l.2i -.1i'\D'l.1i .1i'` will draw the line . The position after a graphical object has been drawn is at its "end", where for circles and ellipses, the end is at the right side. As with other commands, default units are ems horizontally and line spaces vertically.

These commands are not processed by DITROFF, but by the postprocessors, and are intended for preprocessors like PIC, IDEAL, and PLTROFF.

Other commands are `\H'n'`, which sets the character height to *n* points, and `\S'n'`, which will slant the character *n* points. The function of *n* is like in the `\s'n'` command, but *n* is absolute, while $\pm n$ is relative to the current value and 0 restores the previous value. Due to some optimisation in DITROFF, `\H` is not foolproof. There are plans to change these two things in the style of the `.bd`

request.

The `.sy commandline` command executes the commandline and returns, the output of the command is not captured by DITROFF, but this can be done by redirecting the output and reading it with the `.so` request.

The `.pi` command now also works in DITROFF, as it used to do in NROFF.

The `.cf file` copies *file* into the DITROFF output stream, without processing. This request is not new in (NROFF) but was undocumented before.

The string `*.T` contains the name of the current typesetter.

The transparent mode has been fixed, so that transparent output actually can appear in the output. This makes it possible to give commands directly to the postprocessors such as

```
.if "\*(.T"har" x ...
```

For a more detailed description of the new features, read the new *Addendum* to the **TROFF USER'S MANUAL**.

As a final remark, I have to note that Kernighan DITROFF is based on the Berkeley TROFF, which had some bugs/features changed to support the `-me` macro's.

Preprocessors, Postprocessors and other loose ends

Well, this is a paper in itself, as are probably all the other things not touched upon, such as

- Comparisons of TROFF with other text-processing systems, and the future directions of text-processing systems,
- Implications of new technology in text processing such as laser-printing,
- Problems associated with previewing.

The appendices of this report will show the characters and fonts currently (August 1983) available with DITROFF. This will likely change in the near future.

Consulted Literature

Ossana, J. F. *NROFF/TROFF User's manual*, Comp. Sci. Tech. Rep. 54, Bell Laboratories, Murray Hill, NJ, October 1981.

Kernighan, Brian W. *A typesetter-indepent TROFF*, Comp. Sci. Tech. Rep. 54, Murray Hill, NJ, Revised March 1982.

McLean, Ruari *Typography*, Thames and Hudson Ltd, London 1980.

Seybold, John W. *Fundamentals of Modern Composition*, Seybold Publications Inc, Media, Penna. 1977.

Knuth, Donald E, *T_EX and METAFONT, New directions in Typesetting*, American Mathematical Society and Digital Press, Bedford Mass. 1979.

Reid, Brian K. *Scribe: A Document Specification Language and its Compiler*, Ph.D. disertation, Computer Science Department, Carnegie Mellon Univ., Pittsburgh, Pa, October 1980.

Treebus, K. F. *Tekstwijzer, Een gids voor het grafisch verwerken van tekst*, Staatsuitgeverij, 's-Gravenhage, 1982.

Linden, Fons, van der *De grafische technieken*, Canteleer bv, de Bilt 1979.

Times Bold Italic

Internal name: BI, Internal number: 4

ABCDEFGHIJKLMNOPQRSTUVWXYZ

abcdefghijklmnopqrstuvwxyz

0 1 2 3 4 5 6 7 8 9

' % * [] = " + # ! \$ & () ' , - . : ; ? / @

1/3 2/3 1/8 1/4 3/8 1/2 5/8 3/4 7/8

† ° ¢ ' • - — — ^

ff fi fl ffi ffl

Æ CE æ æ ^ ~ ~ ~ ~ ~ " Ø ø ı j i ; ß « » ‹ › Ð đ Ł ł

Special Mathematical Font #1

This font is compatible with older versions of troff. It has a greek character set, so this will be used if the font that is used doesn't have its own greek characters.

Internal name: S, Internal number: 9.

" ' \ ^ ~ / < > { } # @ + - = * _ ® © □ †

A B Γ Δ E Z H Θ I K Λ M N Ξ O Π P Σ T T Φ X Ψ Ω

α β γ δ ε ζ η θ ι κ λ μ ν ξ ο π ρ σ τ υ φ χ ψ ω

ħ √ ⁻ ≥ ≤ ≡ ~ ≅ ≠ → ← ↑ ↓ × ÷ ± ∪ ∩ ⊂ ⊃ ⊆ ⊇ ∞ ∂

§ ∇ ∩ ∫ α ∅ ∈ ‡ ☞ ☛ | ○ [()] { } | [] [] []

Special Mathematical Font #2

This font is incompatible with older versions of troff. A lot of characters used to be available with complicated defines in eqn and /usr/pub/eqnchar. On the Harris there are a lot available as a single character. Symbols used often by the different departments of the Mathematical Centre are included as well.

Internal name: Sm, Internal number: 10.

∀ ∃ ⊕ ⊙ ⊗ ⊥ ∉ ¶ ∅ // ∏ ∥ ⇒ ⇔ ⇐ > ≥ ≤ < ∴ ∘ ≡

≈ ⇔ ⇨ ~ ∙ ∓ ∠ ∞ ∩ ∪ ■ ħ ∇ ∧ ∃ ∄ ≫ ≪ ⇔

∏ £

SMALL CAPS FONTS.

The small caps fonts will automatically map lower case input characters to upper case. They are special designed to look nice in combinations with lower-case as in f.i.

Input

\ f(Rstroff \ fP-manual

instead of the usual

\ s-2TROFF \ s0-manual

Output

TROFF-manual

TROFF-manual

TIMES ROMAN SMALL CAPS

INTERNAL NAME: Rs, INTERNAL NUMBER:

ABCDEFGHIJKLMNOPQRSTUVWXYZ

& Æ Œ

TIMES BOLD SMALL CAPS

INTERNAL NAME: Bs, INTERNAL NUMBER:

ABCDEFGHIJKLMNOPQRSTUVWXYZ

& Æ Œ

The next fonts are sans-serif fonts. On the harris are no sans-serif italics fonts, so these are made by slanting the 'roman' fonts by 12°.

Vega Light

Internal name: VI, Internal number: 5

ABCDEFGHIJKLMNOPQRSTUVWXYZ

abcdefghijklmnopqrstuvwxyz

0 1 2 3 4 5 6 7 8 9

' % * [] = " + # ! \$ & () ' , - . : ; ? / @

1/3 2/3 1/8 1/4 3/8 1/2 5/8 3/4 7/8

† ° ¢ ' • - — — ^

ff fi fl ffi ffl

Æ Œ æ œ ~~~~~, ~~~~~ Ø ø | j i ç ß « » ‹ › Ð ð Ł ł

Vega Light Italic

Internal name: vl, Internal number: 6

ABCDEFGHIJKLMNOPQRSTUVWXYZ

abcdefghijklmnopqrstuvwxyz

0 1 2 3 4 5 6 7 8 9

' % * [] = " + # ! \$ & () ' , - . : ; ? / @

1/3 2/3 1/8 1/4 3/8 1/2 5/8 3/4 7/8

† ° ¢ ' • - — — ^

ff fi fl ffi ffl

Æ Œ æ œ ~~~~~, ~~~~~ Ø ø | j i ç ß « » ‹ › Ð ð Ł ł

Vega

Internal name V, Internal number: 7

ABCDEFGHIJKLMNOPQRSTUVWXYZ

abcdefghijklmnopqrstuvwxyz

0 1 2 3 4 5 6 7 8 9

' % * [] = " + # ! \$ & () ' , - . : ; ? / @

1/3 2/3 1/8 1/4 3/8 1/2 5/8 3/4 7/8

† ° ¢ ' • - — — ^

ff fi fl ffi ffl

Æ Œ æ œ ~ ~ ~ ~ ~ Ø ø | j | ÷ ß « » < > Đ đ Ł ł

Vega Italic

Internal name v, Internal number: 8

ABCDEFGHIJKLMNOPQRSTUVWXYZ

abcdefghijklmnopqrstuvwxyz

0 1 2 3 4 5 6 7 8 9

' % * [] = " + # ! \$ & () ' , - . : ; ? / @

1/3 2/3 1/8 1/4 3/8 1/2 5/8 3/4 7/8

† ° ¢ ' • - — — ^

ff fi fl ffi ffl

Æ Œ æ œ ~ ~ ~ ~ ~

Ø ø | j | ÷ ß « » < > Đ đ Ł ł

Vega Medium

Internal name Vm, Internal number:

ABCDEFGHIJKLMNOPQRSTUVWXYZ

abcdefghijklmnopqrstuvwxyz

0 1 2 3 4 5 6 7 8 9

' % * [] = " + # ! \$ & () ' , - . : ; ? / @

1/3 2/3 1/8 1/4 3/8 1/2 5/8 3/4 7/8

† ° ¢ ' • - — — ^

ff fi fl ffi ffl

Æ Œ æ œ ~ ~ ~ ~ ~ Ø ø | j | ÷ ß « » < > Đ đ Ł ł

Baskerville bold italics

Internal name: bl, Internal number:

ABCDEFGHIJKLMNOPQRSTUVWXYZ

abcdefghijklmnopqrstuvwxyz

0 1 2 3 4 5 6 7 8 9

' % * [] = " + # ! \$ & () ' , - . : ; ? / @

1/3 2/3 1/8 1/4 3/8 1/2 5/8 3/4 7/8

† ° ¢ ' • - — — ^

ff fi fl ffi ffl

Æ Œ œ æ ~ ~ ~ " Ø ø ı j i ç ß « » ‹ › Ð ð Ĺ ĺ

Baskerville small caps

Internal name: bs, Internal number:

ABCDEFGHIJKLMNOPQRSTUVWXYZ

& Æ Œ

Baskerville bold small caps

Internal name: bS, Internal number:

ABCDEFGHIJKLMNOPQRSTUVWXYZ

& Æ Œ

Century Schoolbook

Internal name: cr, Internal number:

ABCDEFGHIJKLMNOPQRSTUVWXYZ

abcdefghijklmnopqrstuvwxyz

0 1 2 3 4 5 6 7 8 9

' % * [] = " + # ! \$ & () ' , - . : ; ? / @

1/3 2/3 1/8 1/4 3/8 1/2 5/8 3/4 7/8

† ° ¢ ' • - — — ^

ff fi fl ffi ffl

Æ Œ œ æ ~ ~ ~ " Ø ø ı j i ç ß « » ‹ › Ð ð Ĺ ĺ

Century Schoolbook italics

Internal name: ci, Internal number:

ABCDEFGHIJKLMNOPQRSTUVWXYZ

abcdefghijklmnopqrstuvwxyz

0 1 2 3 4 5 6 7 8 9

'%*[] = " + # !\$ & () ', - . : ; ? / @

1/3 2/3 1/8 1/4 3/8 1/2 5/8 3/4 7/8

† ° ¢ ' • - — — ^

ff fi fl ffi ffl

Æ CE æ œ ~~~~~ ~ Ø ø ı j i ð ß « » ‹ › Đ đ Ł ł

Century Schoolbook bold

Internal name: cb, Internal number:

ABCDEFGHIJKLMNOPQRSTUVWXYZ

abcdefghijklmnopqrstuvwxyz

0 1 2 3 4 5 6 7 8 9

'%*[] = " + # !\$ & () ', - . : ; ? / @

1/3 2/3 1/8 1/4 3/8 1/2 5/8 3/4 7/8

† ° ¢ ' • - — — ^

ff fi fl ffi ffl

Æ CE æ œ ~~~~~ ~ Ø ø ı j i ð ß ‹ › « » Đ đ Ł ł

Century Schoolbook bold italics

Internal name: ci, Internal number:

ABCDEFGHIJKLMNOPQRSTUVWXYZ

abcdefghijklmnopqrstuvwxyz

0 1 2 3 4 5 6 7 8 9

'%*[] = " + # !\$ & () ', - . : ; ? / @

1/3 2/3 1/8 1/4 3/8 1/2 5/8 3/4 7/8

† ° ¢ ' • - — — ^

ff fi fl ffi ffl

Æ CE æ œ ~~~~~ ~ Ø ø ı j i ð ß ‹ › « » Đ đ Ł ł

Laurel

Internal name: lr, Internal number:

ABCDEFGHIJKLMNOPQRSTUVWXYZ

abcdefghijklmnopqrstuvwxyz

0 1 2 3 4 5 6 7 8 9

‘% * [] = " + # ! \$ & () ’ , - . : ; ? / @

1/3 2/3 1/8 1/4 3/8 1/2 5/8 3/4 7/8

† ° ¢ ' • - — — ^

ff fi fl ffi ffl

Æ CE æ œ ~ ~ ~ ~ ~ " Ø ø ı j Ĩ Ĳ ß « » ‹ › Đ đ Ł ł

Laurel italics

Internal name: li, Internal number:

ABCDEFGHIJKLMNOPQRSTUVWXYZ

abcdefghijklmnopqrstuvwxyz

0 1 2 3 4 5 6 7 8 9

‘% * [] = " + # ! \$ ¢ () ’ , - . : ; ? / @

1/3 2/3 1/8 1/4 3/8 1/2 5/8 3/4 7/8

† ° ¢ ' • - — — ^

ff fi fl ffi ffl

Æ CE æ œ ~ ~ ~ ~ ~ " Ø ø ı j Ĩ Ĳ ß « » ‹ › Đ đ Ł ł

Laurel bold

Internal name: lb, Internal number:

ABCDEFGHIJKLMNOPQRSTUVWXYZ

abcdefghijklmnopqrstuvwxyz

0 1 2 3 4 5 6 7 8 9

‘% * [] = " + # ! \$ & () ’ , - . : ; ? / @

1/3 2/3 1/8 1/4 3/8 1/2 5/8 3/4 7/8

† ° ¢ ' • - — — ^

ff fi fl ffi ffl

Æ CE æ œ ~ ~ ~ ~ ~ " Ø ø ı j Ĩ Ĳ ß « » ‹ › Đ đ Ł ł

MC-fontcatalog

Laurel bold italics

Internal name: II, Internal number:

ABCDEFGHIJKLMNOPQRSTUVWXYZ

abcdefghijklmnopqrstuvwxyz

0 1 2 3 4 5 6 7 8 9

' % * [] = " + # ! \$ % & ' () , - . : ; ? / @

1/3 2/3 1/8 1/4 3/8 1/2 5/8 3/4 7/8

† ° ¢ ' • - — — ^

ff fi fl ff fl

Æ Œ œ æ ~ ~ ~ ~ ~ " Ø ø ı ı ı ı ß « » ¸ ð ð ð ð ð

Appendix II

Input Naming Conventions for ‘,`, and — and for Non-ASCII Special Characters

Non-ASCII characters and *minus* on the standard fonts.

Input Character			Input Character		
Char	Name	Name	Char	Name	Name
'		close quote	œ	\(oe	œ
'		open quote	Ð	\(-D	Icelandic D
<	\(fo	french ‘	ð	\(-d	Icelandic d
>	\(fc	french ’	Ł	\(/L	Polish L
«	\(Fo	double french ‘	ł	\(/l	Polish l
»	\(Fc	double french ’	Ø	\(/O	Scandinavian OE
—	\(em	3 / 4 Em dash	ø	\(/o	Scandinavian oe
-	—	hyphen or	ß	\(sz	German sz
-	\(hy	hyphen	ı	\(r!	reversed !
—	\(—	current font minus	ı	\(r?	reversed ?
●	\(bu	bullet	ı	\(i	dotless i
½	\(12	1 / 2	ı	\(j	dotless j
⅓	\(13	1 / 3	’	\(aa	acute accent
⅔	\(23	2 / 3	`	\(ga	grave accent
¼	\(14	1 / 4	^	\(a^	accent circonflexe
¾	\(34	3 / 4	ˇ	\(ah	háček
⅛	\(18	1 / 8	¨	\(ad	diëresis
⅜	\(38	3 / 8	~	\(a~	accent tilde
⅝	\(58	5 / 8	°	\(ac	corona
⅞	\(78	7 / 8	¸	\(ce	cedille
fi	\(fi	fi	¸	\(ho	hook
fl	\(fl	fl	”	\(”a	accent ”
ff	\(ff	ff	,	\(a,	underline comma
ffi	\(Fi	ffi	˘	\(ab	accent breve
ffl	\(Fl	ffl	°	\(de	degree
Æ	\(AE	Æ	†	\(dg	dagger
æ	\(ae	æ	’	\(fm	foot mark
Œ	\(OE	Œ	¢	\(ct	cent sign

Non-ASCII characters and $_{-}$, $+$, $-$, $=$, and $*$ on the special font.

The ASCII characters |, \, {, } and _ exist *only* on the special font and are printed as a 1-em space if that font is not mounted.

Note that the characters `^` and `~` are really diacritical mark, so they will have an effective width of 0. If you want to use them as a character, you have to surround them with some white, `\ |` is enough.

If a character is not on the current font, there will be searched for on all the other mounted fonts, (including the font as a result of the last `.ft` or `\f` command), starting at the position of the first special fonts in a wrap around manner.

The following characters exist on the special fonts. The special math plus, minus, and equals are provided to insulate the appearance of equations from the choice of standard fonts.

Char	Input Name	Character Name
------	------------	----------------

The next characters are on font S

+	\(pl	math plus
-	\(mi	math minus
=	\(eq	math equals
*	\(**	math star
§	\(sc	section
'	\(aa	acute accent
`	\(ga	grave accent
—	\(ul	underrule
/	\(sl	slash (matching backslash)
α	\(*a	alpha
β	\(*b	beta
γ	\(*g	gamma
δ	\(*d	delta
ε	\(*e	epsilon
ζ	\(*z	zeta
η	\(*y	eta
θ	\(*h	theta
ι	\(*i	iota
κ	\(*k	kappa
λ	\(*l	lambda
μ	\(*m	mu
ν	\(*n	nu
ξ	\(*c	xi
ο	\(*o	omicron
π	\(*p	pi
ρ	\(*r	rho
σ	\(*s	sigma
ς	\(ts	terminal sigma
τ	\(*t	tau
υ	\(*u	upsilon
φ	\(*f	phi
χ	\(*x	chi
ψ	\(*q	psi
ω	\(*w	omega
Α	\(*A	Alpha†
Β	\(*B	Beta†
Γ	\(*G	Gamma
Δ	\(*D	Delta
Ε	\(*E	Epsilon†
Ζ	\(*Z	Zeta†
Η	\(*Y	Eta†
Θ	\(*H	Theta
Ι	\(*I	Iota†
Κ	\(*K	Kappa†
Λ	\(*L	Lambda
Μ	\(*M	Mu†
Ν	\(*N	Nu†

Char	Input Name	Character Name
------	------------	----------------

Ξ	\(*C	Xi
Ο	\(*O	Omicron†
Π	\(*P	Pi
Ρ	\(*R	Rho†
Σ	\(*S	Sigma
Τ	\(*T	Tau†
Υ	\(*U	Upsilon
Φ	\(*F	Phi
Χ	\(*X	Chi†
Ψ	\(*Q	Psi
Ω	\(*W	Omega
√	\(sr	square root
—	\(rn	root en extender
≥	\(>=	>=
≤	\(<=	<=
≡	\(==	identically equal
≈	\(≈	approx =
~	\(ap	approximates
≠	\(!=	not equal
→	\(->	right arrow
←	\(<-	left arrow
↑	\(ua	up arrow
↓	\(da	down arrow
×	\(mu	multiply
÷	\(di	divide
±	\(+ -	plus-minus
∪	\(cu	cup (union)
∩	\(ca	cap (intersection)
⊂	\(sb	subset of
⊃	\(sp	superset of
⊆	\(ib	improper subset
⊇	\(ip	improper superset
∞	\(if	infinity
∂	\(pd	partial derivative
∇	\(gr	gradient
¬	\(no	not
∫	\(is	integral sign
∝	\(pt	proportional to
∅	\(es	empty set
∈	\(mo	member of
	\(br	box vertical rule
‡	\(dd	double dagger
☞	\(rh	right hand
☜	\(lh	left hand
BS	\(bs	Bell System logo
	\(or	or
○	\(ci	circle
{	\(lt	left top of big curly bracket
[\(lb	left bottom
}	\(rt	right top
]	\(rb	right bot

<i>Char</i>	<i>Input Name</i>	<i>Character Name</i>	<i>Char</i>	<i>Input Name</i>	<i>Character Name</i>
}	\(lk	left center of big curly bracket			
}	\(rk	right center of big curly bracket			
	\(bv	bold vertical			
	\(lf	left floor (left bottom of big square bracket)			
	\(rf	right floor (right bottom)			
	\(lc	left ceiling (left top)			
	\(rc	right ceiling (right top)			
-	\(ru	baseline rule			
®	\(rg	registered			
©	\(co	copyright			
□	\(sq	square			

The next characters are on font Sm

■	\(bl	blot	†	\(-	
.	\(a.	accent	‡	\(=	
∀	\(oA	for all	ℏ	\(-h	h-bar
∃	\(oE	exists	∠	\(an	angle
ℏ	\(-h		ℵ	\(Ah	aleph
⊕	\(c+	circle plus	⋮	\(Vh	veth
⊗	\(c*	circle times	ג	\(Gl	gimel
⊙	\(c.	circle dot	£	\(BP	(british) sterling pound
⊥	\(pp	perpendicular to			
∨	\(OR	or			
∧	\(AN	and			
∉	\(nm	no member of			
⊃	\(cn	contains			
⊄	\(nc	not containing			
¶	\(as	paragraph mark			
∮	\(ic	contour integral			
∥	\(ll	parallel			
∏	\(cp	coproduct			
	\(l	absolute			
⇒	\(Dr	double arrow right			
⇔	\(<>	double two way arrow			
⇐	\(Dl	double arrow left			
⇌	\(D>				
↶	\(Cl	curly left arrow			
↷	\(Cr	curly right arrow			
↻	\(C>	wiggly right arrow			
↲	\(>				
↻	\(c>	>			
↻	\(=>	>=			
↻	\(=<	<=			
↻	\(c<	<			
↻	\(>>	>>			
↻	(<<	<<			
◦	\(cs	composite			
≈	\(=˜	approx =			
≈	\(C˜	approx			

ONTVANGEN 2 5 JAN. 1984