% $Id: modes.mf,v 1.31 2023/12/12 22:49:08 karl Exp $
%
% Compiled 1991, 1992, 1993, 1994, 1995, 1996, 1997, 1998, 2002, 2004,
% 2005, 2008, 2020 by Karl Berry. This file is not copyrighted and may
% be used freely. You can retrieve the latest version from
% https://ctan.org/pkg/modes, among other places.
%
% For bug reports, discussion, or to submit a new **mode_def**, please
% send it to tex-k@tug.org (explanations below).
%
% For general discussion about METAFONT, the mailing list
% tex-fonts@math.utah.edu is a good place.
%
% Feel free to change the definitions of *localfont*, *screen_cols*,
% and *screen_rows* at the end of file (see explanations below).
%
% The mode definitions start at 'Here are the modes', several hundred
% lines down. The companion files modelist.txt and
% modenames.txt list the modes one per line, with and without comments.
%
% A common use for modes nowadays is to make high-resolution bitmaps from
% METAFONT-only fonts to include in PDF output or for autotracing. The
% *dpdfezzz* mode is an 8000 dpi mode with canonical parameter values,
% intended for this purpose. (Running dvips -Ppdf will use this.)
% If you want a lower resolution, similar canonical modes are *supre*
% at 2400 dpi mode and *ultre* at 1200 dpi.
%
% This file can be run through mft and TeX to produce a nice
% pretty-printed listing; the resulting modes.pdf file is included
% in the distribution.
%
% @mffile{
%   author = "The Metafont community",
%   version = "4.3",
%   date = "Tue Dec 12 14:49:12 PST 2023"
%   filename = "modes.mf",
%   email = "tex-k@tug.org"
%   checksum = "2674 13389 97995",
%   codetable = "ISO/ASCII",
%   supported = "yes",
%   docstring = "
%
% This file is a collection of putatively all extant METAFONT modes.
%
% If you have a device which is not mentioned in this file, the best
% thing to do is try to find a device with similar resolution (search
% for the appropriate lines), and see if that suits (a list of fonts to
% try is given above). Otherwise, methods for fiddling with the
% parameters are described in detail below.
%
% Unfortunately, the number of modes eats up a lot of memory; if your
% METAFONT has not increased the table sizes, you may need to remove
% some of the modes from this file (please rename it to something else then,

% e.g., `local.mf`). If you can suggest a way to redefine **mode␣def**
% and/or **mode␣setup**, even better; right now, the amount of memory
% used is approximately four times the length of the **mode␣def** names.
%
% The primary names are strictly lowercase. This makes it feasible to use
% them for portable directory names, and the TeX Directory Structure
% standard recommends doing so. The synonyms are historical equivalents.
%
% It also makes definitions to put specials identifying the mode in
% the METAFONT GF output, and to put the coding scheme and
% other so-called Xerox-world information in the TFM output. This can
% be made to happen by calling **mode␣include␣extra␣info**.
%
% It also defines a macro **landscape** that inverts *aspect␣ratio* and
% changes *pixels␣per␣inch*, so you can say `mode := whatever;`
% `landscape; ...` to get landscape fonts. But I can't think of any
% reasonable way to reflect the **landscape** in the directory name, so
% there are also **mode␣def**'s for the devices with non-square aspect
% ratios in landscape mode.
%
% Finally, it has some code to handle write-white devices better; this
% code comes into play if a **mode␣def** includes the statement
% **mode␣write␣white␣setup␣**;. Such **mode␣def**s should also define
% *blacker␣min*. For further discussion of write/white and white/black
% devices, see Pierre MacKay's article in the proceedings of the
% 1991 Raster Imaging and Digital Typography conference:
%
% @String{proc-RIDT91 = "Raster Imaging and Digital Typography II"}
% @String{pub-CUP = "Cambridge University Press"}
%
% @Inproceedings{Mackay:RIDT91-205,
%   author =   "Pierre A. MacKay",
%   title =   "Looking at the Pixels: Quality Control for 300 dpi
%   Laser Printer Fonts, especially {METAFONT}s ",
%   pages =   "205--215",
%   crossref =   "Morris:RIDT91",
% }
%
% @Proceedings{Morris:RIDT91,
%   title =   proc-RIDT91,
%   booktitle =   proc-RIDT91,
%   year =   "1991",
%   editor =   "Robert A. Morris and Jacques Andr{é}",
%   publisher =   pub-CUP,
%   address =   pub-CUP:adr,
%   acknowledgement = ack-kb,
% }
%
% This file is typically loaded when making a METAFONT base; TeX Live does
% this by default, but to do it manually, for example, the command line
%   inimf plain input modes dump
% makes a file `plain.base` (or `plain.bas`, or something like that)
% with all the modes herein defined (plain itself defines modes called

% *proof*, *smoke*, and *lowres*.)
%
% You can make the Computer Modern base with the command line:
%   inimf plain input modes input cmbase dump
% It's generally best to avoid doing this, since it's
% easy to forget to update them. Just using `plain.base` is simplest.
%
% On Unix systems, you then install the base file in the system directory
% as `mf.base`. METAFONT uses the name it was invoked as to determine
% the format or base file to read; thus running `mf` reads
% `mf.base`, running `cmmf` reads `cmmf.base`, and so on.
% `plain.base` and `mf.base` should be the same file (either a hard
% or soft link is ok), so the examples in *The METAFONTbook* work.
%
% A user selects a particular mode when running METAFONT
% by assigning to the variable *mode*. For example:
%   mf \mode:=cx; input cmr10
% sets up values appropriate for the CanonCX engine.
%
% If no mode is assigned, the default is *proof* mode, as stated in
% *The METAFONTbook*. This is the cause of the ".2602gf" files which
% are the subject of periodic questions. The remedy is simply to assign
% a different mode—*localfont*, for example.
%
% Every site should define the mode *localfont* to be a synonym for the
% mode most commonly used. This file defines *localfont* to be *ljfour*.
% The values for *screen_rows* and *screen_cols*, which determine how big
% METAFONT's window for online output is, should perhaps also be
% changed; certainly individual users should change them to their
% own tastes.
%
% This file defines `?` to type out a list of all the known
% **mode_def**s (once only).
%
% Technically, a **mode_def** is a METAFONT definition that typically
% consists of a series of assignments to various device-specific variables,
% either primitive or defined in plain. These variables include the
% following (page numbers refer to *The METAFONTbook*:
%
% *aspect_ratio*: the ratio of the vertical resolution to the horizontal
% resolution (page 94).
%
% *blacker*: a correction added to the width of stems and similar
% features, to account for devices which would otherwise make them
% too light (page 93). (Write-white devices are best handled by a more
% sophisticated method than merely adding to *blacker*, as explained
% above.) Compare your results with a good high-resolution example,
% such as one of the volumes of *Computers & Typesetting*.
% If you compare against the output of a typical write-black 300 dpi
% engine, you will almost certainly wind up with something too dark.
% *blacker* should never be negative, the EC fonts do not compile with
% such a value.
%

% *fillin*: a correction factor for diagonals and other features which
% would otherwise be "filled in" (page 94). An ideal device would
% have *fillin* = 0 (page 94). Negative values for *fillin* typically
% have either gross effects or none at all, and should be avoided.
% Positive values lighten a diagonal line, negative values darken it.
% Changes in the *fillin* value tend to have abruptly non-linear effects
% on the various design-sizes and magnifications of a typeface.
%
% *fontmaking*: if nonzero at the end of the job, METAFONT writes
% a TFM file (page 315).
%
% *o_correction*: a correction factor for the "overshoot" of curves
% beyond the baseline or x-height. High resolution curves look better
% with overshoot, so such devices should have *o_correction* = 1; but
% at low resolutions, the overshoot appears to simply be a distortion
% (page 93). Here some additional comments about *o_correction*,
% courtesy of Pierre MacKay (edited by Karl):
%
% At present, I find that *o_correction* works nicely at 80 pixels per
% em, and gets increasingly disturbing as you move down towards 50
% pixels per em. Below that I do not think it ought to happen at all.
%
% The problem, of course, is that full *o_correction* is supposed to
% occur only at the zenith and nadir of the curve of 'o', which is
% a small region at high resolution, but may be a long line of
% horizontal pixels at medium resolution. The full *o_correction*
% does not change a 300 dpi `cmr10`, but it changes a 21-pixel
% high `cmr12` to be 23 pixels high. The extra height and depth
% is seen along a line of seven pixels at the bottom and five at
% the top. This is a pronounced overshoot indeed.
%
% For high-resolution devices, such as phototypesetters, the values
% for *blacker*, *fillin*, and *o_correction* don't matter all that much,
% so long as the values are within their normal ranges: between
% 0 and 1, with the values approaching 0, 0, and 1 respectively.
%
% *pixels_per_inch*: the horizontal resolution; the METAFONT primitive
% *hppp* (which is what determines the extension on the GF filename,
% as among other things) is computed from this (page 94). (An "inch"
% is 72.27 pt in the TeX world.)
%
% To be more precise, you can determine the resolution of a font given
% a **mode_def** and a magnification m by simply multiplying
% *pixels_per_inch* for that **mode_def** by m. (Your results may differ
% from METAFONT's if you don't use equivalent fixed-point arithmetic.)
% Then you can determine the number used in the name of the GF font
% output by rounding. For example, a font generated at magstep(.5)
% (which is $\sqrt{1.2}$, which METAFONT computes as 1.09544) for a printer
% with *pixels_per_inch* = 300 will have a resolution of 328.63312 dots
% per inch, and the GF filename will include the number `329`.
%
% *proofing*: says whether to put additional specials in the GF file for
% use in making proofsheets via, e.g., the utility program `GFtoDVI`

% (page 323–4).
%
% *tracingtitles*: if nonzero, strings that appear as METAFONT statements
% are typed on the terminal (page 187).
%
% Pierre MacKay `mackay@cs.washington.edu` has a collection of Unix
% tools to make up a minifont of indicator characters to help in testing.
%
% Neenie Billawala's article in the April 1987 issue of *TUGboat*
% describes how to test your printer for the best set of values for the
% magic numbers above. Here are some brief comments on the subject,
% courtesy of Rocky Bernstein and Paul Abrahams:
%
% For medium-to-low resolution devices, you can first set the *blacker*
% and *o_correction* to 0 and decide on a *fillin* value by looking at
% the diagonal of a lowercase 'z' in `cmtt10`, or various lines in
% LaTeX's `line10` font. The diagonal should be the same thickness
% as the horizontal bars of the 'z'. Then I decide on the *blacker*
% value, generally by checking the smaller fonts for too much filling
% in. Finally, you can set the *o_correction* using the guidelines
% suggested above.
%
% The easiest way to make a new **mode_def** is not by modifying this file
% and rebuilding your base file every time. Instead, use a separate file
% that contains the appropriate values for the mode parameters and read
% it in when running METAFONT. If you're using Dvips or another utility
% that calls `MakeTeXPK` to make PK files, remember you'll have to call
% METAFONT explicitly to make fonts until you've rebuilt the base files.
%
% To use a separate mode file with METAFONT, use the following
% command line:
%   mf \smode:="newmode.mf"; mag:=magstep (2.0); input cmr10
% substituting whatever font and magnification you wish, or omitting
% the magnification altogether.
%
% The file `newmode.mf` should contain lines like this (with no
% **mode_def** or **enddef**):
%   mode_param (pixels_per_inch, 100);
%   mode_param (blacker, 0);
%   mode_param (fillin, 0);
%   mode_param (o_correction, 1);
%   mode_common_setup_;
% changing the values as appropriate, of course. Once you're satisfied
% with the parameters, use inimf as described below to rebuild and
% install the plain (and any other) base files.
%
% For more information on the use of *smode*, see page 269 of
% *The METAFONTbook*.
%
% Matt Swift has contributed a short TeX file to help in testing new
% modes. Remember to remove a leading `"%% "` from each line after
% extracting it. (Only a single % is shown in the printed version.)
% If you don't use this file for testing, please mention what fonts

% at what sizes you tested your new mode on. This will help other
% people wondering where particular values came from. Ideally,
% you would try normal, bold, and italic variants, at sizes around
% 5 pt, 10 pt and 15 pt.
%
% % modetest.tex   -- a file to test a METAFONT mode
% %
% % by Matt Swift <swift@alum.mit.edu>
% %
% % This file is in the public domain.
% %
% % \def\fileversion{v1.2}
% % \def\filedate{1995/12/31}
% %
% % This LaTeX 2e file generates a test page useful for finding a good
% % METAFONT mode for your printer. It includes the most sensitive
% % letters in three sizes and all standard CMR font shapes.
% %
% % I've made the macros abstract, and I think this file could easily
% % be adapted to test modes for other METAFONT fonts, or simply font
% % appearance in general.
% %
% % If you want to adapt this to a non-LaTeX format, the LaTeX-specific
% % commands below that must be altered are \documentclass,
% % \begin{document}, \end{document}, \makeatletter, \makeatother,
% % \@for, \@setfontsize, \encodingdefault, \pagestyle, \normalfont,
% % \rmfamily, \sffamily, \ttfamily, \mdseries, \bfseries, \upshape,
% % \itshape, \scshape, and \slshape.
%
% \def\encodingdefault{T1} % New "Cork" font encoding (dc fonts).
% \def\encodingdefault{OT1} % Old font encoding (cm fonts).
%
% \documentclass{article}
% \begin{document}
%
% % This line can be replaced (by, e.g., sed) to contain a mode name.
%
% ::Mode::
%
% \def\makesize#1#2#3{
%   \expandafter\def\csname ptsize#1\endcsname{#2}
%   \expandafter\def\csname blsize#1\endcsname{#3}
% }
%
% %%
% % DEFINE HERE THE POINT SIZES with baselineskips you would like to test. %
% % With the defaults of 5, 10, and 14 point sizes, everything will fit on %
% % one page very easily. Twocolumn would allow several more sizes.   %
% %%
%
% \makesize {A}{5}{6}
% \makesize {B}{10}{12}
% \makesize {C}{14}{18}

```
%
% \def\sizelist{A,B,C}
%
% \def\letters{%
% MoOzZffii-a\"egsS [/$\backslash$\par
% }
%
% \makeatletter
% \let\setfontsize\@setfontsize
% \let\for\@for
% \parindent\z@
% \makeatother
%
% \pagestyle{empty}
%
% \def\showfonts{%
% %
% % The groups prevent warnings when intermediate fonts are not available.
% %
% {\rmfamily \mdseries \upshape \letters}  % allow no space before this
%   {\rmfamily \mdseries \slshape \letters}
%   {\rmfamily \mdseries \itshape \letters}
%   {\rmfamily \mdseries \scshape \letters}
%
%   {\rmfamily \bfseries \upshape \letters}
%   {\rmfamily \bfseries \slshape \letters}
%   {\rmfamily \bfseries \itshape \letters}
%
%   {\sffamily \mdseries \upshape \letters}
%   {\sffamily \mdseries \slshape \letters}
%
%   {\sffamily \bfseries \upshape \letters}
%
%   {\ttfamily \mdseries \upshape \letters}
%   {\ttfamily \mdseries \slshape \letters}
%   {\ttfamily \mdseries \itshape \letters}
%   {\ttfamily \mdseries \scshape \letters}
% }
%
% % The \expandafters expand \sizelist.
% %
% \expandafter   \for
% \expandafter   \sizename
% \expandafter   :%
% \expandafter   =%
%                \sizelist
%   \do {\setfontsize {\sizename}
%                     {\csname ptsize\sizename\endcsname}
%                     {\csname blsize\sizename\endcsname}%
%        \vskip 1ex\noindent
%        \llap{\normalfont\csname ptsize\sizename \endcsname\,pt\quad}%
%        \showfonts}
%
```

% \end{document}
% % end of modetest.tex
%
%"
% }

% Don't let ourselves be processed twice.
if known *modes_mf*: **endinput**; **fi**;
modes_mf := 4.3;

% Identify ourselves in the format file.
base_version := *base_version* & "/modes 4.3";

% Here are useful macros (also called definitions) we use throughout.

% First, some comments about how the *mode_defs* are constructed (from
% rocky@panix.com). In the past, *mode_defs* unconditionally
% assigned a value to the various mode-dependent parameters.
% For example, they contained an assignment *fontmaking* := 1, which
% tells METAFONT to write a TFM file.
%
% But suppose you want to generate a font using all of the setup for
% some mode m, but do not want to generate a TFM? One could create
% another mode that doesn't have the assignment, but this seems a bit
% wasteful since the rest of the code in the mode would be duplicated.
% Furthermore, given the way the mode definitions were written, it was
% not possible to change the mode parameters. To see why, I review how
% a METAFONT run typically works.
%
% First, METAFONT is invoked with some base file to load. Then you might
% want give additional instructions, such as **scrollmode**, or *mode* := *cx*.
% Next, you input a parameter file, say cmr10. The parameter file
% calls a driver file such as roman.mf with the command
% **generate roman**. Finally, the driver finishes with **bye** or **end**.
% Thus, any additional commands you give after the input of the
% parameter file are ignored.
%
% Usually, one of the first things a driver file does is to call
% **mode_setup**. It is here that the mode parameters are set. (In our
% hypothetical mode, it would be here that *fontmaking* is assigned.)
%
% To allow a flexible setting of *fontmaking*, we can make a simple
% change: in the **mode_def**, first test to see if a value has been
% defined prior and only make the assignment if not. That is:
% **if** unknown *fontmaking*: *fontmaking* := 1; **fi**.
%
% Alas, this doesn't work. Primitives, like *fontmaking*, are always
% known. So instead we create "guard" variables, specifically,
% *mode_guard_.fontmaking*; we set the guard when we assign the
% parameter. Then we test whether the guard is known before we
% actually do an assignment. This allows more flexible definitions: you
% can specify some of the parameters, and keep the defaults for others.
%
% It is also possible to write a program that creates a **mode_def**
% on the fly. Although useful, this has a slightly different focus

8

% than starting with an existing **mode_def** and changing a couple
% of parameters. In particular, one still has to peek inside the
% file to see what the old values were and set them again (in the
% new context). Also, such on-the-fly **mode_def** generation programs
% are inherently less machine-independent than a scheme that does
% everything in METAFONT itself.
%
% The upshot of all this is the following: we say, e.g.,
% **mode_param**(*fontmaking*, 1) below, instead of using the assignment
% primitive directly. The name (“**mode_param**”) is kept somewhat
% short because you can also use this to override a mode assignment
% on the command line or in response to the ** prompt.

def mode_param(suffix *v*)(**expr** *e*) =
    if unknown *mode_guard_.v*:
        v := *e*;
        mode_guard_.v := 1;
    fi
enddef;

% This macro is invoked by all the modes, after *pixels_per_inch*
% has been defined, thus saving some space and time.
def mode_common_setup_ =
    mode_param(*proofing*, 0);
    mode_param(*fontmaking*, 1);
    mode_param(*tracingtitles*, **if** *pixels_per_inch* > 1200: 1 **else**: 0 **fi**);
enddef;

% In a similar spirit, here are definitions to change to “landscape”
% mode. You just say `mode := whatever; landscape; ...`,
% and when **mode_setup** is executed, the *aspect_ratio* will be
% inverted, and *pixels_per_inch* changed.
def landscape =
    extra_setup := *extra_setup* & `"landscape_;"`
enddef;

def landscape_ =
    begingroup
        interim *warningcheck* := 0;
        pixels_per_inch := *aspect_ratio* ∗ *pixels_per_inch*;
        aspect_ratio := 1/*aspect_ratio*;
        fix_units; % Too bad we can't do this after any *extra_setup*.
    endgroup
enddef;

% Here are macros to add specials with mode information to the GF file.
%
% Specifically, we add the *pixels_per_inch*, *o_correction*,
% *aspect_ratio* (if not 1), *mag*, *fillin*, and **mode_def** name. This
% information can be used to automatically verify that a font file name
% matches the specification within the file. For example, you could
% check that the number in the filename matches *mag* ∗ *pixels_per_inch*.
% Or, if the **mode_def** name is part of the font directory path
% (e.g., you put fonts in `.../texmf/fonts/pk/cx`), that all of the
% bitmap files in the directory have the expected **mode_def** name.

```
def mode_special_(suffix $) =
    string s, d;
    s := str $;
    d := decimal scantokens s;
    special s & "=" & d;
enddef;

def mode_output_specials_ =
    begingroup
        save d, s, p, p_p_i;
        string p;

        interim warningcheck := 0; % In case pixels_per_inch > 4096.

        % We need the old pixels_per_inch to compute
        % the true device resolution.
        p_p_i = pixels_per_inch/mag;

        % But now we want to change pixels_per_inch,
        % so save the old value.
        save pixels_per_inch;
        pixels_per_inch := p_p_i;

        special "jobname=" & jobname;
        mode_special_(mag);

        p := if string mode:
                        mode
                    else:
                        substring(0, length(mode_name[mode]) − 1) of mode_name[mode]
                    fi;
        special "mode=" & p;

        mode_special_(pixels_per_inch);
        if aspect_ratio ≠ 1:
            mode_special_(aspect_ratio);
        fi;
        mode_special_(blacker);
        mode_special_(fillin);
        mode_special_(o_correction);
    endgroup
enddef;
```

% Here are macros for Xerox-world font info, which can be useful even
% if you never use a Xerox printer. For instance, **crudetype** uses
% the **coding_scheme** and it is nice to have the font family on record.
% This goes into both the TFM file (as **headerbyte** information), and
% into the GF file (as a **special**).

% Make the string s be n bytes long.
```
def BCPL_string(expr s, n) =
    for l := if length(s) ≥ n: n − 1 else: length(s) fi: l
        for k := 1 upto l: , substring(k − 1, k) of s endfor
        for k := l + 2 upto n: , 0 endfor
    endfor
enddef;
```

% The string s names the encoding scheme, e.g., `TeX text`.

def coding_scheme expr s =
    headerbyte 9: **BCPL_string**$(s, 40)$;
    special "`codingscheme=`" $\&$ s
enddef;

% The string s names the font family, e.g., `CMR`.
def font_family expr s =
    headerbyte 49: **BCPL_string**$(s, 20)$;
    special "`fontid=`" $\&$ s
enddef;

% The integer x gives the family member number, which should be
% between 0 and 255.
def font_face_byte expr x =
    headerbyte 72: x;
    special "`fontfacebyte`";
    numspecial x
enddef;

% So users can say **if** known *Xerox_world*: ... **fi**, per *The METAFONTbook*.
Xerox_world := 1;

% For users who want extra information in the output file.
% This used to be done automatically by redefining end, but DEK reported
% that as a serious bug on 19 February 2008 to tex-implementors.
%
def *mode_extra_info* =
    if *fontmaking* > 0:
        font_family *font_identifier_*;
        coding_scheme *font_coding_scheme_*;
        font_face_byte $\max(0, 254 - \text{round } 2designsize)$;
        mode_output_specials_;
    fi;
enddef;

% This macro **mode_include_extra_info** will insert the above extra
% information, most importantly the **coding_scheme**, upon **end**.
% This is called from the `mktextfm` and `mktexpk` scripts if
% the environment variable `MF_MODE_EXTRA_INFO` is set; the
% `mftrace` program (`https://ctan.org/pkg/mftrace`) can use this.
%
% We need to redefine **end** in the macro to output the information, so
% save the primitive meaning. And we must make both **end** and **bye**
% **inner** tokens, so we can define them in the macro.
%
let *original_end_* = **end**;
inner end, bye;
%
def mode_include_extra_info =
    def end =
        mode_extra_info;
        *original_end_*;
    enddef;

    % *The METAFONTbook* gives **bye** two different definitions (on pages
    % 278 and 321). The first is used in `plain.mf` and is merely

% a synoynym for the primitive **end**. The second, which is not part
% of `plain.mf`, is similar to the code given above. We want the
% extra information to get into the output files regardless of whether
% the METAFONT source used **end** or **bye**. We just changed **end**;
% now we have to redefine **bye** again (as on page 278).
let bye = **end**;
enddef;

% This is tested in `mktextfm` with known to see if it is ok to
% call **mode_include_extra_info**, since there is apparently no way to test
% whether a macro is defined, unlike T_EX.
boolean *mode_include_extra_info_available*;
mode_include_extra_info_available := **true**;

% Now make **end** and **bye outer** again; it seems **let** does not
% restore this attribute.
%
outer end, **bye**;

% Here are macros to handle write-white devices.
%
% The basic correction for write-white fonts occurs in the definition
% of *font_setup*. This can be used to extend or change the write-black
% definition in Computer Modern's `cmbase.mf` or other base files
% based on CM, such as `dxbase.mf`. This has no effect at 1200 dpi.
def mode_write_white_setup_ =
  newinternal *blacker_min*;
  def define_whole_blacker_pixels(**text** t) =
          forsuffixes $ = t: $:= hround($\#$ * *hppp* + *blacker*);
                  if $ \leq$ *blacker_min* $- 1$: $:= *blacker_min*; **fi endfor enddef**;

  def define_whole_vertical_blacker_pixels(**text** t) =
          forsuffixes $ = t: $:= vround($\#$ * *hppp* + *blacker*);
                  if $ \leq$ *blacker_min* $- 1$: $:= $blacker\_min_o\_$; **fi endfor enddef**;

  % Only do the above once, in case a font file (unnecessarily)
  % calls **mode_setup** more than once.
  let mode_write_white_setup_ = **relax**
enddef;

% Here are the modes, given mostly in alphabetical order.

% From `J.Hicks@warwick.ac.uk`.
mode_def *agfafzz* =   % AGFA 400PS (406dpi)
    mode_param(*pixels_per_inch*, 406);
    mode_param(*blacker*, .2);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, .6);
    mode_common_setup_;
enddef;
AgfaFourZeroZero := *agfafzz*;

% From `picheral@univ-rennes1.fr`.
mode_def *agfatfzz* =   % AGFA P3400PS (400dpi)
    mode_param(*pixels_per_inch*, 400);
    *cx_*;
enddef;
AgfaThreeFourZeroZero := *agfatfzz*;

% From `rokicki@neon.stanford.edu`.
mode_def *amiga* =   % Commodore Amiga (100dpi)
    mode_param(*pixels_per_inch*, 100);
    mode_param(*blacker*, 0);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, .2);
    mode_common_setup_;
enddef;
onezz := *amiga*;
OneZeroZero := *amiga*;

mode_def *aps* =   % Autologic APS-Micro5 (723dpi)
    mode_param(*pixels_per_inch*, 722.909);
    mode_param(*blacker*, .2);
    mode_param(*fillin*, .2);
    mode_param(*o_correction*, 1);
    mode_common_setup_;
enddef;

% From `rocky@panix.com`. Tested on the single APS-6 at IBM Research.
mode_def *apssixhi* =   % Autologic APS-Micro6 (1016dpi)
    mode_param(*pixels_per_inch*, 1016);
    mode_param(*blacker*, 0);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, 1);
    mode_common_setup_;
enddef;

% From `ee@dacth51.bitnet`.
mode_def *atariezf* =   % Atari ST SLM 804 printer (300dpi)
    mode_param(*pixels_per_inch*, 300);
    mode_param(*blacker*, 0);
    mode_param(*fillin*, .5);
    mode_param(*o_correction*, 0);
    mode_param(*blacker_min*, 2);
    mode_common_setup_;
    mode_write_white_setup_;

enddef;
AtariSLMEightZeroFour := *atariezf*;

% From `W.Spit@fys.ruu.nl`. `N.Poppelier@elsevier.nl` says that
% different previewers use different resolutions (95 dpi, 96 dpi,
% or 101 dpi), but no one seems to know what the real resolution is.
mode_def *atarinf* =   % Atari previewer (95dpi)
    mode_param(*pixels_per_inch*, 95);
    mode_param(*blacker*, 0);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, 0.1);
    mode_common_setup_;
enddef;
AtariNineFive := *atarinf*;

mode_def *atarins* =   % Atari previewer (96dpi)
    mode_param(*pixels_per_inch*, 96);
    mode_param(*blacker*, 0);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, 0.1);
    mode_common_setup_;
enddef;
AtariNineSix := *atarins*;

% From `ee@dacth51.bitnet`.
mode_def *atariotf* =   % Atari ST SM 124 screen (101dpi)
    mode_param(*pixels_per_inch*, 101);
    mode_param(*blacker*, 0);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, .4);
    mode_common_setup_;
enddef;
AtariSMOneTwoFour := *atariotf*;

mode_def *bitgraph* =   % BBN Bitgraph (118dpi)
    mode_param(*pixels_per_inch*, 118);
    mode_param(*blacker*, .55);
    mode_param(*fillin*, .1);
    mode_param(*o_correction*, .3);
    mode_common_setup_;
enddef;

% From `sjwright@cix.compulink.co.uk`, 9 February 1994.
mode_def *bjtenex* =   % Canon BubbleJet 10ex (360dpi)
    mode_param(*pixels_per_inch*, 360);
    mode_param(*blacker*, .6);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, .6);
    mode_common_setup_;
enddef;

% `cgweav@eskimo.com` (Clayton Weaver), 4 February 1997.
% Might want to recheck *o_correction*, which could vary per unit.
mode_def *bjtzzex* =   % Canon BubbleJet 200ex (360 dpi)
    mode_param(*pixels_per_inch*, 360);
    mode_param(*blacker*, 1.2);

```
        mode_param(fillin, .2);
        mode_param(o_correction, 0);
        mode_common_setup_;
enddef;

% Alastair.Jenkins@nrsc.no, 30 January 1997.
mode_def bjtzzs =                                           % Canon BubbleJet 200 (720x360dpi)
        mode_param(pixels_per_inch, 720);
        mode_param(aspect_ratio, 0.5);
        mode_param(blacker, 0.0);
        mode_param(fillin, 0);
        mode_param(o_correction, 1.0);
        mode_common_setup_;
enddef;

% Alastair.Jenkins@nrsc.no, 30 January 1997.
mode_def bjtzzl =                                           % BubbleJet 200 landscape (360x720 dpi)
        bjtzzs_;
        landscape;
enddef;

mode_def boise =                                            % HP 2680A (180dpi)
        mode_param(pixels_per_inch, 180);
        mode_param(blacker, .55);
        mode_param(fillin, .1);
        mode_param(o_correction, .3);
        mode_common_setup_;
enddef;

% From Yves.Arrouye@imag.fr.
mode_def canonbjc =                                         % Canon BJC-600 (360dpi)
        mode_param(pixels_per_inch, 360);
        mode_param(blacker, 0);
        mode_param(fillin, 0);
        mode_param(o_correction, .8);
        mode_common_setup_;
enddef;
CanonBJCSixZeroZero := canonbjc;

% From swartz@cs.wisc.edu, 8 April 1993. The straightforward
% mode with blacker = 0, fillin = 0, o_correction = 1 seems to
% work fine for the Canon EX engine inside Apple's LaserWriter Pro 630.
% It produces light, clear lines and type. But ajcarr@ccvax.ucd.ie
% sent in the revised values below on 12 December 1993, tested on
% the major CM fonts at 5, 7, and 10 pt and producing slightly
% better results.
mode_def canonex =                                         % LaserWriter Pro 630 (600dpi)
        mode_param(pixels_per_inch, 600);
        mode_param(blacker, .2);
        mode_param(fillin, .1);
        mode_param(o_correction, .85);
        mode_common_setup_;
enddef;
CanonEX := canonex;

mode_def canonlbp =                                         % Symbolics LGP-10 (240dpi)
```

```
    mode_param(pixels_per_inch, 240);
    mode_param(blacker, .2);
    mode_param(fillin, .2);
    mode_param(o_correction, .4);
    mode_common_setup_;
enddef;
CanonLBPTen := canonlbp;

% This is really 1301.5; MF produces 1301, so use that.
mode_def cg =                                        % Compugraphic 8600 (1301x1569dpi)
    mode_param(pixels_per_inch, 1301);
    mode_param(aspect_ratio, 1569/pixels_per_inch);
    mode_param(blacker, .2);
    mode_param(fillin, .2);
    mode_param(o_correction, 1);
    mode_common_setup_;
enddef;
CompugraphicEightSixZeroZero := cg;

mode_def cgl =                                        % Compugraphic 8600 landscape (1569x1302dpi)
    cg_;
    landscape;
enddef;

% These values from Linotype Linotronic [13]00 modified to 1200 dpi.
% From wagman%muse.hepnet@Csa2.LBL.Gov.
mode_def cgnszz =                                     % Compugraphic 9600 (1200dpi)
    mode_param(pixels_per_inch, 1200);
    mode_param(blacker, .65);
    mode_param(fillin, −.1);
    mode_param(o_correction, .5);
    mode_common_setup_;
enddef;
CompugraphicNineSixZeroZero := cgnszz;
```

% This has a resolution of $5333 + \frac{1}{3}$ pixels per inch.

```
mode_def crs =                                        % Alphatype CRS (5333dpi)
    mode_param(pixels_per_inch, 4000 + 4000/3);
    mode_param(blacker, 4);
    mode_param(fillin, 0);
    mode_param(o_correction, 1);
    mode_common_setup_;
enddef;
```

% This applies to the LaserWriter Plus, HP LaserJet, HP LaserJet Plus,
% and also the Canon LBP-LX engine, in the LaserJet IIP, QMS 410,
% and Apple Personal LaserWriter, and also to the CanonSX engine,
% in the LaserWriter II family. And hammond@jila02.Colorado.EDU
% says it works well for the "enhanced-resolution" LaserJet III.
% swartz@cs.wisc.edu is developing a mode for the Canon EX engine
% inside an Apple Pro 630 printer.

```
mode_def cx =                                         % Canon CX, SX, LBP-LX (300dpi)
    mode_param(pixels_per_inch, 300);
    mode_param(blacker, 0);
    mode_param(fillin, .2);
    mode_param(o_correction, .6);
```

mode_common_setup_;
enddef;
CanonCX := *cx*;
corona := *cx*;
dp := *cx*; % some kind of DataProducts
hplaser := *cx*;
imagen := *cx*;
kyocera := *cx*;
laserwriter := *cx*;
% I have seen a claim the LaserJet II was the Canon SX
% write-white engine, but I don't think that's right.
laserjethi := *cx*;
laserjet := *cx*;
% `ogawa@orion.arc.nasa.gov` says that this is definitely not a
% write-white engine, despite earlier versions of this file claiming
% the contrary. Thus, probably the same parameters as *cx* will do.
CanonSX := *cx*;
CanonLBPLX := *cx*;

% At least magstep 2 is recommended at this low resolution.
mode_def *datadisc* =   % DataDisc (70dpi)
    mode_param(*pixels_per_inch*, 70);
    mode_param(*blacker*, 0);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, .2);
    mode_common_setup_;
enddef;
DD := *datadisc*;

mode_def *newdd* =   % DataDisc (70x93dpi)
    mode_param(*aspect_ratio*, $^4/_3$);
    *datadisc_*;
enddef;
DataDiscNew := *newdd*;

mode_def *newddl* =   % DataDisc landscape (93x70dpi)
    *newdd_*;
    landscape;
enddef;

% From `mcgrant@rascals.stanford.edu`. True resolution is 98.2236
% by 102.4. See comments for *DECsmall* just above.
mode_def *declarge* =   % DEC 19-inch, 1280 x 1024 (100dpi)
    mode_param(*pixels_per_inch*, 100);
    mode_param(*blacker*, 0);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, 0);
    mode_common_setup_;
enddef;
DEClarge := *declarge*;
elvira := *declarge*;

% From `mcgrant@rascals.stanford.edu`. True resolution is 78.1069
% by 86.0612, but a square aspect ratio works better; furthermore,
% Computer Modern isn't prepared to deal with fractional pixel values.
mode_def *decsmall* =   % DEC 17-inch, 1024 x 768 (82dpi)

```
        mode_param(pixels_per_inch, 82);
        mode_param(blacker, 0);
        mode_param(fillin, 0);
        mode_param(o_correction, 0);
        mode_common_setup_;
enddef;
DECsmall := decsmall;

% From fieberjr@whitman.bitnet.
mode_def deskjet =                                              % HP DeskJet 500 (300dpi)
        mode_param(pixels_per_inch, 300);
        mode_param(blacker, 0);
        mode_param(fillin, 0);
        mode_param(o_correction, .6);
        mode_common_setup_;
enddef;
HPDeskJet := deskjet;

% From stsmith@ll.mit.edu, 10 May 93.
% With fillin = 0, the diagonal of cmtt10's 'z' is too thin.
% blacker = .8 too thin, 2 too thick.
mode_def docutech =                                            % Xerox 8790 or 4045 (600dpi)
        mode_param(pixels_per_inch, 600);
        mode_param(blacker, 1);
        mode_param(fillin, .1);
        mode_param(o_correction, 0.9);
        mode_common_setup_;
enddef;
XeroxDocutech := docutech;

% From waits.mf.
mode_def dover =                                               % Xerox Dover (384dpi)
        mode_param(pixels_per_inch, 384);
        mode_param(blacker, 1.2);
        mode_param(fillin, 0);
        mode_param(o_correction, .6);
        mode_common_setup_;
enddef;

% Reported by Silas Brown, 4 April 2003, via Debian bug 184875.
% dvips -Ppdf wants 8000 dpi fonts, so define a mode for that.
mode_def dpdfezzz =                                            % dvips -Ppdf (8000dpi)
        mode_param(pixels_per_inch, 4000 * 2);
        mode_param(blacker, 0);
        mode_param(fillin, 0);
        mode_param(o_correction, 1);
        mode_common_setup_;
enddef;

% ghibo@galileo.polito.it, for the Amiga ShowDVI previewer.
mode_def eighthre =                                            % EightThree (83dpi)
      mode_param(pixels_per_inch, 83);
      mode_param(blacker, 0);
      mode_param(fillin, 0);
      mode_param(o_correction, .2);
      mode_common_setup_;
```

enddef;
EightThree := *eighthre*;

% `arno.wagner@acm.org`, 25 November 1997. To print in 360dpi (much
% faster, but lower quality) use the *epstylus* mode instead. This
% printer seems to make smaller dots when printing at 720 dpi and
% larger ones when printing at 360 dpi. I tried 720x1440 resolution
% as well, but found it not worth the additional time. If you use
% Ghostscript, you need at least version 5.03 to support 720 dpi on
% this printer. This mode may work with the Epson Stylus color 800 as
% well. I tested this mode with Matt Swift's test, found above.
% With *fillin* set to zero, the test had no 'át 5 pt.
%
mode_def *epscszz* =   % Epson Stylus Color 600 (720 dpi)
    mode_param(*pixels_per_inch*, 720);
    mode_param(*blacker*, .25);
    mode_param(*fillin*, 0.5);
    mode_param(*o_correction*, .8);
    mode_common_setup_;
enddef;

% From `metcalf@catfish.LCS.MIT.EDU`, 5 Dec 1992.
mode_def *epsdrft* =   % Epson (120x72dpi)
    mode_param(*pixels_per_inch*, 120);
    mode_param(*aspect_ratio*, 72/*pixels_per_inch*);
    *epson_*;
enddef;
epsdraft := *epsdrft*;

mode_def *epsdrftl* =   % Epson (72x120dpi)
    *epsdrft_*;
    landscape;
enddef;

% From `metcalf@catfish.LCS.MIT.EDU`, 5 Dec 1992.
mode_def *epsfast* =   % Epson fast (60x72dpi)
    mode_param(*pixels_per_inch*, 60);
    mode_param(*aspect_ratio*, 72/*pixels_per_inch*);
    *epson_*;
enddef;

mode_def *epsfastl* =   % Epson fast landscape (72x60dpi)
    *epsfast_*;
    landscape;
enddef;

% From Hippocrates Sendoukas `<isendo@mail.ariadne-t.gr>`, 17 April 1999.
mode_def *epsmed* =   % Epson med MX/FX 9-pin (240x144dpi)
    mode_param(*pixels_per_inch*, 240);
    mode_param(*aspect_ratio*, 144/*pixels_per_inch*);
    *epson_*;
enddef;

mode_def *epsmedl* =   % Epson med MX/FX 9-pin landscape (144x240dpi)
    *epsmed_*;
    landscape;
enddef;

% These values from Charles Karney, *TUGboat* 8(2), page 133. This
% is for the Epson MX/FX family (-85, -286), which are 9-pin printers.
% The 24-pin LQ family have higher resolutions; no one has sent me
% definitions for them yet. Ditto for Epson's laser printer.
% (Thanks to `cargo@escargot.cray.com` for all this information.)
mode_def *epson* =   % Epson MX/FX 9-pin (240x216dpi)
    mode_param(*pixels_per_inch*, 240);
    mode_param(*aspect_ratio*, 216/*pixels_per_inch*);
    mode_param(*blacker*, 0);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, .2);
    mode_common_setup_;
enddef;
EpsonMXFX := *epson*;
epshi := *epson*;
epsonfx := *epson*;

mode_def *epsonl* =   % Epson MX/FX 9-pin landscape (216x240dpi)
    *epson_*;
    landscape;
enddef;

% From `sdh@po.cwru.edu`, 6 September 93.
% The modes *cx* and *HPLaserJetIIISi* are too spindly.
% This works (not awesome, o's and e's are slightly taller than
% they should be in large pt. fonts) on my Epson Action Laser 1500
% with LaserJetIIIsi emulation and RITech (Epson's Resolution
% Enhancement). It might work for the model 1000 or some HP's.
mode_def *epsonact* =   % Epson Action Laser 1500 (300dpi)
    mode_param(*pixels_per_inch*, 300);
    mode_param(*blacker*, .8);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, 0.95);
    mode_common_setup_;
enddef;
EpsonAction := *epsonact*;

% Corrected to 216 dpi vertically, 5 Dec 1992.
% From `metcalf@catfish.LCS.MIT.EDU`.
mode_def *epsonlo* =   % Epson (120x216dpi)
    mode_param(*pixels_per_inch*, 120);
    mode_param(*aspect_ratio*, 216/*pixels_per_inch*);
    *epson_*;
enddef;
epslo := *epsonlo*;

mode_def *epsonlol* =   % Epson landscape (216x120dpi)
    *epsonlo_*;
    landscape;
enddef;

% From `Sebastian_Kirsch@kl.maus.de`, 19 April 1996. In comparison
% to some postscript fonts, the characters seemed to light with blacker
% 0, but much too heavy with a blacker greater than 1. I tried blacker
% .6 and finally settled for .7. All the other values are rather
% fictional, I didn't really test them out.

mode_def *epsonsq* =   % Epson SQ 870 (360dpi)
    mode_param(*proofing*, 0)
    mode_param(*pixels_per_inch*, 360);
    mode_param(*blacker*, .7);
    mode_param(*fillin*, .2);
    mode_param(*o_correction*, .9);
    mode_common_setup_;
enddef;
EpsonSQEightSevenZero := *epsonsq*;

% Following three modes from `marc@mpi.nl` (Marc Fleischeuers).
% I could not quite get the 'z' diagonal to get as thin as the
% horizontal lines, even pushing *fillin* up to 0.8. This printer tends
% to make things lighter on lower resolutions so I compensate a little
% with increasing *blacker*. But not all the way, as this would fill in
% the little holes in the 'e' and 's' at 5 pt. Otherwise it's pretty
% cool, not as crisp as an *ljfour* but better than most inkjets I've seen.
mode_def *epstypro* =   % Epson Stylus Pro (360dpi)
    mode_param(*pixels_per_inch*, 360);
    mode_param(*blacker*, 0.2);
    mode_param(*fillin*, 0.8);
    mode_param(*o_correction*, 0);
    mode_common_setup_;
enddef;
EpsonStylusPro := *epstypro*;

mode_def *epstyplo* =   % Epson Stylus Pro (180dpi)
    mode_param(*pixels_per_inch*, 180);
    mode_param(*blacker*, .35);
    mode_param(*fillin*, 0.8);
    mode_param(*o_correction*, 0);
    mode_common_setup_;
enddef;
EpsonStylusProLow := *epstyplo*;

% Good time saver, almost as good as 720x720 but a lot faster.
mode_def *epstypmd* =   % Epson Stylus Pro (720x360dpi)
    mode_param(*pixels_per_inch*, 720);
    mode_param(*aspect_ratio*, 360/*pixels_per_inch*);
    mode_param(*blacker*, 0);
    mode_param(*fillin*, 0.8);
    mode_param(*o_correction*, 0);
    mode_common_setup_;
enddef;
EpsonStylusProMed := *epstypmd*;

mode_def *epstypml* =   % Epson Stylus Pro landscape (360x720dpi)
    *epstypmd_*;
    landscape;
enddef;
epstypmdl := *epstypml*;

mode_def *epswlo* =   % Epson low MX/FX 9-pin (120x144dpi)
    mode_param(*pixels_per_inch*, 120);
    mode_param(*aspect_ratio*, 144/*pixels_per_inch*);
    *epson_*;

enddef;

mode_def *epswlol* =   % Epson low MX/FX 9-pin landscape (144x120dpi)
    *epswlo_*;
    landscape;
enddef;

mode_def *esphi* =   % Epson Stylus Pro (720dpi)
    mode_param(*pixels_per_inch*, 720);
    mode_param(*blacker*, 0);
    mode_param(*fillin*, 0.8);
    mode_param(*o_correction*, 1);
    mode_common_setup_;
enddef;
EpsonStylusProHigh := *esphi*;

% From `Tobias.Guenzler@uni-konstanz.de`, 8 December 1994.
%
% The *blacker* parameter is the most critical; changing *o_correction*
% has lesser effect, and may also be increased or decreased somewhat.
% I tested this by comparing output with printouts of a HP LaserJet
% printer using the LJ fonts. This printer had the fancy resolution
% enhancement feature (RET) which makes the pixel steps almost
% invisible. I did most of the comparison with `cmr12`, `cmbx12`,
% `cmr12` magstep2 and `cmss9`.
%
% The Stylus printer is a ink printer, but it works with a piezo drive
% instead of a bubble jet. This may be the reason why it draws its lines
% very tiny and thin. At least the pixel diameters are very sharp and
% they are far away from that bulky dots produced by the needles of
% a NEC P6.
mode_def *epstylus* =   % Epson Stylus (360dpi)
    mode_param(*pixels_per_inch*, 360);
    mode_param(*blacker*, .35);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, .8);
    mode_common_setup_;
enddef;

% `ghibo@galileo.polito.it`, for the Amiga ShowDVI previewer.
mode_def *fourfour* =   % FourFour (44dpi)
  mode_param(*pixels_per_inch*, 44);
  mode_param(*blacker*, 0.05);
  mode_param(*fillin*, .1);
  mode_param(*o_correction*, .2);
  mode_common_setup_;
enddef;
FourFour := *fourfour*;

% From `drstrip@intvax.uucp`.
% Revised by `dak@pool.informatik.rwth-aachen.de`, 24 May 1994.
mode_def *gtfax* =   % G3fax (204x196dpi)
    mode_param(*pixels_per_inch*, 204);
    mode_param(*aspect_ratio*, 196/*pixels_per_inch*);
    mode_param(*blacker*, 0);
    mode_param(*fillin*, .2);

      mode_param(*o_correction*, .2);
      mode_common_setup_;
enddef;
GThreefax := *gtfax*;
gtfaxhi := *GThreefax*;

mode_def *gtfaxl* =   % G3fax landscape (196x204dpi)
      *gtfax_*;
      landscape;
enddef;

% From `dak@pool.informatik.rwth-aachen.de`, 24 May 1994.
mode_def *gtfaxlo* =   % G3fax (204x98dpi)
      mode_param(*pixels_per_inch*, 204);
      mode_param(*aspect_ratio*, 98/*pixels_per_inch*);
      *gtfax_*;
enddef;

mode_def *gtfaxlol* =   % G3fax landscape (98x204dpi)
      *gtfaxlo_*;
      landscape;
enddef;

% `ron@mlfarm.com`, 30 October 1995.
mode_def *highfax* =   % G3fax (200dpi)
      mode_param(*pixels_per_inch*, 200);
      mode_param(*blacker*, 0);
      mode_param(*fillin*, .2);
      mode_param(*o_correction*, .2);
      mode_common_setup_;
enddef;
hifax := *highfax*;

% Martin Ruckert, 7 September 2020.
% 600dpi is much higher resolution than currently available on laptops
% or mobile devices, but they do do antialiasing. Here are some words
% from Martin about it:
%
% PK fonts are strictly black and white. On real paper, the ink dots will
% be fuzzy at the edges smoothing the outline. On electronic devices, a
% black and white font does look jagged (unless the device resolution is
% very high). So it is better to produce a black and white font at a
% higher resolution and let the graphics card scale it down to the device
% resolution. At the edges then one device pixel will correspond to
% several font pixels and the graphics card will average over these pixels
% and produce a gray value. The fonts then look much smoother. 600dpi is a
% good compromise. The font is not too big, and it will still look nice.
% If the device resolution is e.g. only 300dpi, 4 pixel in the font will
% map to one pixel on the screen. So around the edges you get 5 different
% gray-levels from all black (0) to all white (4). If the device
% resolution is even lower, the shading at the edges uses even more gray
% values. *blacker* is the only parameter with much effect here.
mode_def *hitexlaptop* =   % HiTeX (HINT) laptop (600dpi)
      mode_param(*pixels_per_inch*, 600);
      mode_param(*blacker*, 0.6);
      mode_param(*fillin*, 0.2);

```
        mode_param(o_correction, .4);
        mode_common_setup_;
enddef;
```

% Martin Ruckert, 7 September 2020. See above.
```
mode_def hitexmobile =                                    % HiTeX (HINT) mobile (600dpi)
        mode_param(pixels_per_inch, 600);
        mode_param(blacker, 1.6);
        mode_param(fillin, 0.2);
        mode_param(o_correction, .4);
        mode_common_setup_;
enddef;
```

% `brumski+@osu.edu`, 27 August 1993.
```
mode_def hprugged =                                       % HP RuggedWriter 480 (180dpi)
        mode_param(pixels_per_inch, 180);
        mode_param(blacker, .55);
        mode_param(fillin, .1);
        mode_param(o_correction, .3);
        mode_common_setup_;
enddef;
```

% Some general comments on the IBM printers, courtesy of `rocky@panix.com`.
%
% IBM 3820's, 3825's, 3827's and 3835's have some sort of corner imaging
% or shading that the IBM 3812's and 3816's don't. The latter two models
% may get this feature in the future.
%
% The IBM 3827 is made by Kodak, the rest are IBM engines.
%
% Some of the other printers have a knob that allows a service engineer
% to set one of up to ten levels of darkness. At IBM Research, we run
% very black. The service engineer sets the level by running a completely
% black page and then two completely blank ones. The black page
% must be black and the following two must be completely white.
%
% Thanks to Jim Hafner (`hafner@ibm.com`) for experimenting with
% *blacker*, and Paul Dantzig for information about these printers.
%
% From `ARNALDO@RIOSC.bitnet`. This is for the 3820, but can be used
% for 3812, 3816, 3825, 3837 3800 and 3827 printers (these are all
% 240 pels IBM printers that use the same font format when driven
% by PSF/VM).
```
mode_def ibm_a =                                          % IBM 38xx (240dpi)
        mode_param(pixels_per_inch, 240);
        mode_param(blacker, .35);
        mode_param(fillin, −.2);
        mode_param(o_correction, .2);
        mode_common_setup_;
enddef;
```

% From `rocky@panix.com`. For the typewriter, slanted, and italic
% fonts, *blacker* = 0 makes the 'M's and 'W's more legible. But then
% the weight of the font does not match the others.
```
mode_def ibmd =                                           % IBM 38xx (240dpi)
```

```
        mode_param(pixels_per_inch, 240);
        mode_param(blacker, .3);
        mode_param(fillin, .4);
        mode_param(o_correction, .75);
        mode_common_setup_;
enddef;
```

% These values from `melvin@math.psu.edu`.
```
mode_def ibmega =                                    % IBM EGA monitor (96x81dpi)
        mode_param(pixels_per_inch, 96);
        mode_param(aspect_ratio, .841);
        mode_param(blacker, .3);
        mode_param(fillin, 0);
        mode_param(o_correction, 0);
        mode_common_setup_;
enddef;

mode_def ibmegal =                              % IBM EGA monitor landscape (81x96dpi)
        ibmega_;
        landscape;
enddef;
```

% From `sperber@provence.informatik.uni-tuebingen.de`, 30 October 1993.
% The difference of 0.1 in *blacker* really does make a difference.
```
mode_def ibmfzon =                                          % IBM 4019 (300dpi)
        mode_param(pixels_per_inch, 300);
        mode_param(blacker, .1);
        mode_param(fillin, 0);
        mode_param(o_correction, .75);
        mode_param(blacker_min, 2);
        mode_common_setup_;
        mode_write_white_setup_;
enddef;
IBMFourZeroOneNine := ibmfzon;
```

% From `rocky@panix.com`. The print engine is made by Lexmark. The
% printing person I asked, Paul Dantzig, says that the print quality of
% the 4019 is fairly regular. Unlike the IBM 4216's, to his knowledge
% only there is only one print engine by Lexmark has been ever used in
% the 4019. And unlike the IBM 4029, there are not knobs on the inside
% that would permit one to adjust the blacker to ones taste.
%
% While both RicohA and cx modes settings are acceptable, it looks
% to me that the RicohA fonts are superior. I base this judgement on
% tops and bottoms of curves on `cmr10` such as 'S', 'U' 'e' 'o' and
% the apostrophes. This effect is especially noticeable in a small font
% like `cmr6`.
%
% If you want to experiment with another setting, I'd start with *RicohA*
% and set *blacker* to .1 or 0 instead of .2 but definitely keep
% **mode_write_white_setup_**; I'd leave *fillin* and *o_correction* unchanged.
%
% From `vumalki%weizmann.weizmann.ac.il@taunivm.tau.ac.il`
% and `plotkin@theory.stanford.edu`.
%

% `hafner@almaden.ibm.com` (Jim Hafner) reports that this works fine
% for the Lexmark 4039, a.k.a. IBM 4039, as along as the "Printer
% Darkness" control is set to "darker".
%
mode_def *ibmfztn* =   % IBM 4029-30-39, 4250 (600dpi)
    mode_param(*pixels_per_inch*, 600);
    mode_param(*blacker*, .05);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, .75);
    mode_common_setup_;
enddef;
IBMFourZeroTwoNine := *ibmfztn*;
IBMFourTwoThreeZero := *ibmfztn*;
IBMFourTwoFiveZero := *ibmfztn*;
IBMFourZeroThreeNine := *ibmfztn*;
LexmarkFourZeroThreeNine := *ibmfztn*;

% From Rick Simpson via `erikjan@icce.rug.nl`.
mode_def *ibmpp* =   % IBM ProPrinter (240x216dpi)
    mode_param(*pixels_per_inch*, 240);
    mode_param(*aspect_ratio*, 216/*pixels_per_inch*);
    mode_param(*blacker*, 0);
    mode_param(*fillin*, .2);
    mode_param(*o_correction*, 1);
    mode_common_setup_;
enddef;
IBMProPrinter := *ibmpp*;
proprinter := *IBMProPrinter*;

mode_def *ibmppl* =   % IBM ProPrinter (216x240dpi)
    *ibmpp_*;
    landscape;
enddef;

% From Rick Simpson via `erikjan@icce.rug.nl`. Also gave values
% of zero for *blacker*, *fillin*, and *o_correction*.
mode_def *ibmsoff* =   % IBM 6154 display (118dpi)
    mode_param(*pixels_per_inch*, 118);
    mode_param(*blacker*, .8);
    mode_param(*fillin*, .2);
    mode_param(*o_correction*, 1);
    mode_common_setup_;
enddef;
IBMSixOneFiveFour := *ibmsoff*;

% From `rocky@panix.com`. This is an old, untested definition.
mode_def *sherpa* =   % IBM 6670 (Sherpa) (240dpi)
    mode_param(*pixels_per_inch*, 240);
    mode_param(*blacker*, 1);
    mode_param(*blacker_min*, 2);
    mode_param(*fillin*, 1);
    mode_param(*o_correction*, .5);
    mode_common_setup_;
    mode_write_white_setup_;
enddef;

IBMSixSixSevenZero := *sherpa*;

% From `vumalki%weizmann.weizmann.ac.il@taunivm.tau.ac.il`.
mode_def *ibmteot* =   % IBM 3812 (240dpi)
    mode_param(*pixels_per_inch*, 240);
    mode_param(*blacker*, .6);
    mode_param(*blacker_min*, 2);
    mode_param(*fillin*, .4);
    mode_param(*o_correction*, 0);
    mode_common_setup_;
    mode_write_white_setup_;
enddef;
IBMThreeEightOneTwo := *ibmteot*;
IBMUlfHolleberg := *IBMThreeEightOneTwo*;

% These values from `d_webb@chcc.harwell.aea-technology.uk`.
mode_def *ibmtetz* =   % IBM 3820 (240dpi)
    mode_param(*pixels_per_inch*, 240);
    mode_param(*blacker*, .78);
    mode_param(*fillin*, .25);
    mode_param(*o_correction*, .5);
    mode_common_setup_;
enddef;
IBMThreeEightTwoZero := *ibmtetz*;

% From `x92@vm.urz-uni-heidelberg.de` via `schoepf@sc.zib-berlin.de`.
mode_def *ibmtont* =   % IBM 3193 screen (100dpi)
    mode_param(*pixels_per_inch*, 100);
    mode_param(*blacker*, 0);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, 0);
    mode_common_setup_;
enddef;
IBMThreeOneNineThree := *ibmtont*;

% From `x92@vm.urz-uni-heidelberg.de` via `schoepf@sc.zib-berlin.de`.
mode_def *ibmtosn* =   % IBM 3179 screen (87x65dpi)
    mode_param(*pixels_per_inch*, 87);
    mode_param(*aspect_ratio*, 0.75);
    mode_param(*blacker*, 0);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, 0);
    mode_common_setup_;
enddef;
IBMThreeOneSevenNine := *ibmtosn*;

mode_def *ibmtosnl* =   % IBM 3179 screen landscape (65x87dpi)
    *ibmtosn_*;
    landscape;
enddef;

% These values from `d_webb@chcc.harwell.aea-technology.uk`.
% `melvin@math.psu.edu` thinks $pixels\_per\_inch = 96$ might be better.
mode_def *ibmvga* =   % IBM VGA monitor (110dpi)
    mode_param(*pixels_per_inch*, 110);
    mode_param(*blacker*, .3);

```
        mode_param(fillin, 0);
        mode_param(o_correction, 0);
        mode_common_setup_;
    enddef;
```

% The Chelgraph IBX is the machine introduced to North American TEX
% users by Type 2000 in Mill Valley, California; telephone (415) 388-8873.
% Since the machine's stated output resolution is only 2000 dpi
% this truly spectacular 9600 dpi must be used for translation to
% an outline font description. This has been tested and used in a
% publication of the University of Washington Press. These values
% from Pierre MacKay, based on Lance Carnes' crs values, at magstep 1.8.

```
mode_def ibx =                                              % Chelgraph IBX (9600dpi)
        mode_param(pixels_per_inch, 4000 + 4000 + 1600);
        mode_param(blacker, 4);
        mode_param(fillin, 0);
        mode_param(o_correction, 1);
        mode_common_setup_;
    enddef;
ChelgraphIBX := ibx;
```

% From `local.mf` via `cudat@cu.warwick.ac.uk`.

```
mode_def itoh =                                             % CItoh 8510A (160x144dpi)
        mode_param(pixels_per_inch, 160);
        mode_param(aspect_ratio, 144/pixels_per_inch);
        mode_param(blacker, 0);
        mode_param(fillin, 0);
        mode_param(o_correction, .1);
        mode_common_setup_;
    enddef;
CItohEightFiveOneZero := itoh;
```

```
mode_def itohl =                                            % CItoh 8510A landscape (144x160dpi)
        itoh_;
        landscape;
    enddef;
```

% From `rokicki@cs.umb.edu`.

```
mode_def itohtoz =                                          % CItoh 310 (240x144dpi)
        mode_param(pixels_per_inch, 240);
        mode_param(aspect_ratio, 144/pixels_per_inch);
        mode_param(blacker, 0);
        mode_param(fillin, 0);
        mode_param(o_correction, .2);
        mode_common_setup_;
    enddef;
citohtoz := itohtoz;
CItohThreeOneZero := itohtoz;
cthreeten := itohtoz;
```

```
mode_def itohtozl =                                         % CItoh 310 landscape (144x240dpi)
        itohtoz_;
        landscape;
    enddef;
```

% Perhaps the value for *fillin* should be 0.

mode_def *iw* =   % Apple ImageWriter (144dpi)
    mode_param(*pixels_per_inch*, 144);
    mode_param(*blacker*, 0);
    mode_param(*fillin*, 0.3);
    mode_param(*o_correction*, .2);
    mode_common_setup_;
enddef;
imagewriter := *iw*;

% From `stsmith@ll.mit.edu`, 20 August 93.
% The mode *cx* is too spindly.
mode_def *jetiiisi* =   % HP Laser Jet IIISi (300dpi)
    mode_param(*pixels_per_inch*, 300);
    mode_param(*blacker*, .2);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, .7);
    mode_common_setup_;
enddef;
HPLaserJetIIISi := *jetiiisi*;

% From John Sauter.
mode_def *lasf* =   % DEC LA75 (144dpi)
    mode_param(*pixels_per_inch*, 144);
    mode_param(*blacker*, .3);
    mode_param(*fillin*, −.1);
    mode_param(*o_correction*, 0);
    mode_common_setup_;
enddef;
LASevenFive := *lasf*;

% Michael Covington's (`mcovingt@ai.uga.edu`) definition for the
% Lexmark Optra R (4049), reflecting a taste for a heavier than
% normal rendition of the Computer Modern fonts.
%
% You may prefer a lesser value of blacker (down to maybe 1.0).
% Initially tested on 10, 12, 17-point CMR and 10-point math italic.
%
% While we're talking about the Optra R, here's another useful fact:
% it takes 32-bit-wide 72-pin SIMMs, 70 or 80 ns. Contrary to the
% documentation, you do not have to use IBM's special SIMMs.
%
% The resolution of 1200 and the *blacker* value of 3 causes `cmbsy7`
% to be generated with incorrect arrows and radical sign. The
% *vtftzzlo* mode also fails. Decreasing *blacker* to 2 works around.
% From `infovore@xs4all.nl` (Olaf Weber) and Henrik Schmiediche.
%
mode_def *lexmarkr* =   % Lexmark Optra R 4049 (1200dpi)
    mode_param(*pixels_per_inch*, 1200);
    mode_param(*blacker*, 2); % used to be 3; works around `cmbsy7` bug
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, 1);
    mode_common_setup_;
enddef;
LexmarkOptraR := *lexmarkr*;

% Klaus Guntermann `<guntermann@iti.informatik.tu-darmstadt.de>`.
% 19 January 1998. Mode for a Lexmark Optra S laser printer in true
% 1200dpi mode. This printer model seems to be the successor of the
% Optra R series above.
%
mode_def *lexmarks* =   % Lexmark Optra S 1250/1650/2450 (1200dpi)
    mode_param(*pixels_per_inch*, 1200);
    mode_param(*blacker*, 1);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, 1);
    mode_common_setup_;
enddef;
LexmarkOptraS := *lexmarks*;

% `uri@watson.ibm.com` (Uri Blumenthal), 9 March 1997.
% This is really a 1200 dpi printer, but it can be operated in 600dpi mode.
mode_def *lexmarku* =   % Lexmark Optra R+ 4049 (600dpi)
    mode_param(*pixels_per_inch*, 600);
    mode_param(*blacker*, .5);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, .75);
    mode_param(*tracingtitles*, 0);
    mode_common_setup_;
enddef;

mode_def *linolo* =   % Linotype Linotronic [13]00 (635dpi)
    mode_param(*pixels_per_inch*, 635);
    *linoone_*;
enddef;
LinotypeOneZeroZeroLo := *linolo*;
linohalf := *LinotypeOneZeroZeroLo*;

% Mode for Linotype Linotronic L-330 with a RIP-50 raster.
% From: Steven T. Smith `stsmith@ll.mit.edu`, 26 October 95.
mode_def *linolttz* =   % Linotronic L-300 with RIP-50 (3386dpi)
    mode_param(*pixels_per_inch*, 3386);
    mode_param(*blacker*, 0);
    mode_param(*o_correction*, 1);
    mode_param(*fillin*, 0);
    mode_common_setup_;
enddef;
LinotypeLThreeThreeZero := *linolttz*;

% These values from `d_webb@chcc.harwell.aea-technology.uk`.
% The 'a' in `cmr5` looks better with *blacker* = .3. Values of .2
% for both *blacker* and *fillin* have also been used.
mode_def *linoone* =   % Linotronic [13]00 (1270dpi)
    mode_param(*pixels_per_inch*, 1270);
    mode_param(*blacker*, .65);
    mode_param(*o_correction*, 1);
    mode_param(*fillin*, −.1);
    mode_common_setup_;
enddef;
LinotypeOneZeroZero := *linoone*;
linohi := *LinotypeOneZeroZero*;

linothreelo := *LinotypeOneZeroZero*;

% These values from `d_webb@chcc.harwell.aea-technology.uk`.
mode_def *linotzzh* =   % Linotype Linotronic 300 (2540dpi)
    mode_param(*pixels_per_inch*, 2540);
    mode_param(*blacker*, .2); % Copied from *aps*—conjectural.
    mode_param(*fillin*, .2); % (ditto)
    mode_param(*o_correction*, 1); % (ditto)
    mode_common_setup_;
enddef;
linothree := *linotzzh*;
LinotypeThreeZeroZeroHi := *linotzzh*;
linosuper := *linotzzh*;

% (From Matt Swift `swift@alum.mit.edu`, 1 Jan 1996.) This is a mode for
% the HP LaserJet 5P, using dvipsk-5.58f and gs-2.6.2. I tuned it using
% the file modetest.tex. The first sweep was (b,f,o) = (0, 0, .3, .6,
% 0). The diagonal of 10 pt lowercase z was too thin at .6, too thick
% at 0. The second sweep was (.4, .5, .6, .7, .3, 0). At .6, the 5 pt
% small-cap lower-case A was on the verge of being filled in, but .6 is
% a reasonable value. Blacker .4 looked a little spindly for many of
% the 5 pt fonts, especially italic and small-cap. The next sweep was
% (.5, .3, 0, .4, .7, 1). The sides of the 14 pt upper-case O
% (especially roman and small-cap) are relatively fatter with
% o-correction 0 as opposed to 1. Almost every other mode for 600 dpi
% printers has 1, so I'm going with that.
%
mode_def *ljfive* =   % HP LaserJet 5 (600dpi)
    mode_param(*pixels_per_inch*, 600);
    mode_param(*blacker*, .75);
    mode_param(*fillin*, .3);
    mode_param(*o_correction*, 1);
    mode_common_setup_;
enddef;
laserjetfive := *ljfive*;

% From Michael Neuhauser `neuhauser@eiunix.tuwien.ac.at`. This is a
% mode for HP LaserJet 5MP. I started with ljfive and found the Computer
% Modern fonts much too black. Therefore I experimented with different
% values of *blacker* to find .4 to be best.
mode_def *ljfivemp* =   % HP LaserJet 5MP (600 dpi)
    mode_param(*pixels_per_inch*, 600);
    mode_param(*blacker*, .4);
    mode_param(*fillin*, .3);
    mode_param(*o_correction*, 1);
    mode_common_setup_;
enddef;
laserjetfivemp := *ljfivemp*;
% There have been many modes for the LaserJet 4. The current values were
% found by `kb@tug.org` to be reasonable on a LaserJet 4MP
% (at the default density setting, with resolution enhancement enabled).
% I don't intend to change them again (unless someone convinces me
% that they are truly mistaken in some way), although I would consider adding
% different modes for other LaserJet 4 printers, if people contribute them.
%

% (`cthiele@ccs.carleton.ca` gets better results with the density
% setting on the printer at 4, instead of 3.)
%
% The first LaserJet 4 mode came from `tonnie@ctrl.phys.tue.nl`,
% 13 January 1993, with *blacker* = 0, *fillin* = 0, and *o_correction* = .6.
% (This definition was forwarded to me by Barbara Beeton, and was
% intended to be preliminary.)
%
% `fj@iesd.auc.dk` says that *IBMFourZeroTwoNine* works fine.
%
% `mbr@research.nj.nec.com` supplied another set of values:
% *blacker* = .6, *fillin* = 0, and *o_correction* = 1. He writes:
% I've tested it extensively at 10 pt and 12 pt in both roman, italic,
% and bold, and I've checked all the standard smaller sizes (5, 6, 7, 8,
% and 9 pt). Works reasonably well on both the LaserJet 4 and the 4si,
% although characters come out somewhat lighter on the 4si. Assumes
% that the density controls are set to their default values and that the
% resolution enhancement feature is enabled. The *blacker* value was
% chosen to make 12 pt text look good; for 10 pt text, set *blacker* = .66.
%
% I felt the output with *blacker* = .6 was too dark; Computer Modern
% was never intended to be as dark as it appears on 300 dpi printers.
% So I've decreased *blacker* to the value below. The other parameters
% don't seem to matter much. (Even *blacker* doesn't matter all
% *that* much.)
%
% Works for a 600 dpi Accel-a-Writer `mackay@cs.washington.edu`,
% 16 August 95.
%
% Possibly also usable for the LaserJet 6 family.
% From `chj@lin.foa.se` (Christian Jvnsson), 29 January 1997.
%
mode_def *ljfour* =   % HP LaserJet 4 (600dpi)
    mode_param(*pixels_per_inch*, 600);
    mode_param(*blacker*, .25);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, 1);
    mode_common_setup_;
enddef;
laserjetfour := *ljfour*;
% `fn@junior.mathtok.polymtl.ca` uses this for the QMS-860.
qmsesz := *ljfour*;
% `pete@lovelace.thi.informatik.uni-frankfurt.de` uses this for the
% Apple LaserWriter Select 360, with a Fuji Xerox Xerographic engine.
aselect := *ljfour*;

% `jrenkema@worldonline.nl`, 2 January 1998. The LaserJet 5M
% **mode_def**, *blacker* = .35, had very light output on the 4000. The
% *blacker* = 2 setting results in output comparable to the LaserJet 5M
% with *blacker* = .35. It is also noteworthy that in the ProRes 1200 mode
% HP's resolution enhancement technology (RET) is not used, thus output
% is exactly according to METAFONT. But perhaps *blacker* = 2 is too much, as
% Computer Modern is supposed to be pretty light.
mode_def *ljfzzz* =   % LaserJet 4000N, ProRes mode (1200dpi)

32

```
    mode_param(pixels_per_inch, 1200);
    mode_param(blacker, 1);
    mode_param(fillin, .1);
    mode_param(o_correction, 1);
    mode_common_setup_;
enddef;
laserjetfourzerozerozero := ljfzzz;
laserjetfourthousand := ljfzzz;

% Nicolai Langfeldt <janl@math.uio.no>, 16 May 1998.
% This is for the default 16ppm 600dpi mode.
%
mode_def ljfzzzfr =                                    % HP LaserJet 4000 FastRes (600dpi)
    mode_param(pixels_per_inch, 600);
    mode_param(blacker, 0);
    mode_param(fillin, .2);
    mode_param(o_correction, 1);
    mode_common_setup_;
enddef;

% From ST-TeX.MF via braams@pttrnl.nl. (The 300dpi LaserJet
% is another cx.)
mode_def ljlo =                                        % HP LaserJet (150dpi)
    mode_param(pixels_per_inch, 150);
    mode_param(blacker, 0);
    mode_param(fillin, .1);
    mode_param(o_correction, 1);
    mode_common_setup_;
enddef;
laserjetlo := ljlo;

% Niko Sauer <nikos@friedrichs.up.ac.za>, 11 October 2000.
% Here are modes developed for and tested on the HP LaserJet 2100T/TN.
% Mode ljtozz is for a resolution of 1200dpi, and ljtozzfr for 600dpi
% Tradeoffs between fillin and blacker resulted in very clear,
% sharp renderings of Computer Modern fonts which appears to be
% preferable to what the modes ljfzzz ljfzzzfr for HP LaserJet 4000
% yield on this printer. Preferences were tested by scrutiny of the
% results by sample of people in the vicinity.
%
mode_def ljtozz =                                      % HP LaserJet 2100T/TN (1200dpi)
    mode_param(pixels_per_inch, 1200);
    mode_param(blacker, .7);
    mode_param(fillin, .15);
    mode_param(o_correction, 1);
    mode_common_setup_;
enddef;
laserjettwoonezerozero := ljtozz;
%
mode_def ljtozzfr =                                    % HP LaserJet 2100T/TN (600dpi)
    mode_param(pixels_per_inch, 600);
    mode_param(blacker, .25);
    mode_param(fillin, .3);
    mode_param(o_correction, 1);
    mode_common_setup_;
```

enddef;
laserjettwoonezerozerofastres := *ljtozzfr*;

% From `mackay@cs.washington.edu`, 13 January 1993. The actual
% machine resolution of this machine is 1000×400, but it is
% adjusted with the aid of software so that a 1000×1000 PK file
% is used. The *o_correction*, however, seems grossly overdone if
% the expected value of at or near unity is applied (on the grounds
% that a 1000 dpi font should be able to do full o-correction).
% Apparently the 400 dpi physical resolution has some effect here.
% In any case, *o_correction* = 0.4 looks better, and lines up with
% about the right optical adjustment on curves. Tested at American
% School of Classical Studies Publications on 18 July, 1992.
mode_def *lmaster* =   % LaserMaster (1000dpi)
    mode_param(*pixels_per_inch*, 1000);
    mode_param(*blacker*, 0.2);
    mode_param(*fillin*, 0.0);
    mode_param(*o_correction*, 0.4);
    mode_common_setup_;
enddef;
lasermaster := *lmaster*;

% From `fran@hexamon.demon.co.uk`, 10 March 1996. I tried the entry
% in modes.mf for a DEC LN03. This turned out much too dark—everything
% looks bold. I did try sending write-black fonts to this printer, the
% hairlines disappear. I don't know if these printers have a
% blackness knob . . .
mode_def *lnotr* =   % DEC LN03R Scriptprinter (300dpi)
    mode_param(*pixels_per_inch*, 300);
    mode_param(*blacker*, 0);
    mode_param(*blacker_min*, 2);
    mode_param(*fillin*, −.6);
    mode_param(*o_correction*, .5);
    mode_common_setup_;
    mode_write_white_setup_;
enddef;
LNOthreR := *lnotr*;

% From Richard Watson at the Queensland Institute of Technology. This
% printer is said to have some kind of Xerox engine, but I don't know which.
mode_def *lnzo* =   % DEC LN01 (300dpi)
    mode_param(*pixels_per_inch*, 300);
    mode_param(*blacker*, .9);
    mode_param(*blacker_min*, 2);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, .5);
    mode_common_setup_;
    mode_write_white_setup_;
enddef;
LNZeroOne := *lnzo*;
lps := *lnzo*;
LPSFourZero := *lnzo*;

% From `hammond@jila.Colorado.EDU`, 21 January 1993. Modified from
% *qms*. Prints exactly like the QMS fonts from Northlake Software.

mode_def *lpstz* =   % DEC lps20 (300dpi)
  mode_param(*pixels_per_inch*, 300);
  mode_param(*blacker*, .6);
  mode_param(*fillin*, −.3);
  mode_param(*o_correction*, .6);
  mode_common_setup_;
enddef;
LPSTwoZero := *lpstz*;

mode_def *lqlores* =   % Epson LQ-500 (180dpi)
  mode_param(*pixels_per_inch*, 180);
  mode_param(*blacker*, 0);
  mode_param(*fillin*, 0);
  mode_param(*o_correction*, .1);
  mode_common_setup_;
enddef;
EpsonLQFiveZeroZeroLo := *lqlores*;

% This and *EpsonLQFiveZeroZeroLo* also work for a Mannesmann 300
% (from `cudat@csv.warwick.ac.uk`, 4 September 1991). The 360×360
% modes for these printers fails for `cudat`, however.
mode_def *lqmed* =   % Epson LQ-500 (360x180dpi)
  mode_param(*pixels_per_inch*, 360);
  mode_param(*aspect_ratio*, 180/*pixels_per_inch*);
  mode_param(*blacker*, 0); % 0.3 avoids 'holes'.
  mode_param(*fillin*, 0);
  mode_param(*o_correction*, .1);
  mode_common_setup_;
enddef;
lqmedres := *lqmed*;
EpsonLQFiveZeroZeroMed := *lqmed*;

mode_def *lqmedl* =   % Epson LQ-500 landscape (180x360dpi)
  *lqmed_*;
  landscape;
enddef;

% These values from `karl@cs.umb.edu`. *blacker* = .8 or more
% thickens dots, to their detriment. *blacker* = .6 produces two-pixel
% stems, which looks pretty good for `cmr`, but it's a little dark
% for `cmti`, and `cmbx` and `cmr` then turn out the same.
% *o_correction* = 1 made no difference. *fillin* = 1 made no
% difference.
mode_def *lview* =   % Sigma L-View monitor (118x109dpi)
  mode_param(*pixels_per_inch*, 118.06);
  mode_param(*aspect_ratio*, 109.25/*pixels_per_inch*);
  mode_param(*blacker*, 0);
  mode_param(*fillin*, 0);
  mode_param(*o_correction*, 0);
  mode_common_setup_;
enddef;

mode_def *lviewl* =   % Sigma L-View monitor landscape (109x118dpi)
  *lview_*;
  landscape;
enddef;

% From `Pierre.Soille@ipk.fhg.de`, 13 February 1995.
% This printer also runs at 300 dpi (try *cx*), 400 dpi (*next*),
% and 600 dpi (*ljfour*).
mode_def *lwpro* =   % Apple LaserWriterPro 810 (800dpi)
    mode_param(*pixels_per_inch*, 800);
    mode_param(*blacker*, 0);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, 1);
    mode_common_setup_;
enddef;

% This is untested.
mode_def *macmag* =   % Mac screens at magstep 1 (86dpi)
    mode_param(*pixels_per_inch*, 86.4);
    mode_param(*blacker*, .35);
    mode_param(*fillin*, .1);
    mode_param(*o_correction*, .3);
    mode_common_setup_;
enddef;

% From the VMS distribution tape (except `karl@cs.umb.edu` changed
% the *o_correction* to zero).
mode_def *mactrue* =   % Mac screen (72dpi)
    mode_param(*pixels_per_inch*, 72);
    mode_param(*blacker*, 0);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, 0);
    mode_common_setup_;
enddef;
MacTrueSize := *mactrue*;

% From `mcgrant@rascals.stanford.edu`, 17 December 1992.
% Various other values made little difference.
mode_def *ncd* =   % NCD 19-inch (95dpi)
    mode_param(*pixels_per_inch*, 95);
    mode_param(*blacker*, 0);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, 0);
    mode_common_setup_;
enddef;

% From `rokicki@neon.stanford.edu`.
mode_def *nec* =   % NEC (180dpi)
    mode_param(*pixels_per_inch*, 180);
    mode_param(*blacker*, 0);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, .2);
    mode_common_setup_;
enddef;

% This is the same as *cx*, except for the resolution.
mode_def *nechi* =   % NEC-P6 (360dpi)
    mode_param(*pixels_per_inch*, 360);
    *cx_*;
enddef;
lqhires := *nechi*;

% `fkr@tooyoo1.l.u-tokyo.ac.jp`, 7 June 1995.
mode_def *neclm* =   % NEC PC-PR406LM (320dpi)
    mode_param(*pixels_per_inch*, 320);
    mode_param(*blacker*, .1);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, .6);
    mode_common_setup_;
enddef;

% `fkr@tooyoo1.l.u-tokyo.ac.jp`, 7 June 1995.
mode_def *nectzo* =   % NEC PC-PR201 series (160dpi)
    mode_param(*pixels_per_inch*, 160);
    mode_param(*blacker*, 0);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, .2);
    mode_common_setup_;
enddef;
NecTwoZeroOne := *nectzo*;

% From `rokicki@neon.stanford.edu`.
mode_def *nexthi* =   % NeXT Newgen (400dpi)
    mode_param(*pixels_per_inch*, 400);
    *cx_*;
enddef;
NeXTprinter := *nexthi*;
Newgen := *nexthi*; % From `lambert@silver.cs.umanitoba.ca`.

% From `rokicki@neon.stanford.edu`.
mode_def *nextscrn* =   % NeXT monitor (100dpi)
    mode_param(*pixels_per_inch*, 100);
    mode_param(*blacker*, 0);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, 0);
    mode_common_setup_;
enddef;
NeXTscreen := *nextscrn*;
nextscreen := *nextscrn*;

% `ghibo@galileo.polito.it`, for the Amiga ShowDVI previewer.
mode_def *nineone* =   % NineOne (91x91) (91dpi)
  mode_param(*pixels_per_inch*, 91);
  mode_param(*blacker*, 0);
  mode_param(*fillin*, 0);
  mode_param(*o_correction*, .2);
  mode_common_setup_;
enddef;
NineOne := *nineone*;

% From `jbotz@mtholyoke.edu`, 21 April 1993.
% Make TFM files only.
mode_def *nullmode* =   % TFM files only (101dpi)
    % The resolution is irrelevant, but METAFONT always ships out
    % characters, so don't use the default huge *proof* resolution.
    mode_param(*pixels_per_inch*, 101);
    mode_param(*proofing*, −1);
    mode_param(*fontmaking*, 1);

enddef;

% `ghibo@galileo.polito.it`, for the Amiga ShowDVI previewer.
mode_def *onetz* =   % OneTwoZero (120/120) (120dpi)
  mode_param(*pixels_per_inch*, 120);
  mode_param(*blacker*, 0);
  mode_param(*fillin*, 0);
  mode_param(*o_correction*, .2);
  mode_common_setup_;
enddef;
OneTwoZero := *onetz*;

% From `deby@cs.utwente.nl` and `issue@vax.oxford.ac.uk`.
mode_def *ocessfz* =   % OCE 6750-PS (508dpi)
    mode_param(*pixels_per_inch*, 508);
    mode_param(*blacker*, 0);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, .7);
    mode_common_setup_;
enddef;
OCESixSevenFiveZeroPS := *ocessfz*;

% From `rokicki@neon.stanford.edu`.
mode_def *okidata* =   % Okidata (240x288dpi)
    mode_param(*pixels_per_inch*, 240);
    mode_param(*aspect_ratio*, 288/*pixels_per_inch*);
    mode_param(*blacker*, 0);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, .2);
    mode_common_setup_;
enddef;
okihi := *okidata*;

mode_def *okidatal* =   % Okidata landscape (288x240dpi)
    *okidata_*;
    landscape;
enddef;

% `roussel@henri.chem.uleth.ca`. For the dark smoothing mode.
mode_def *okifte* =   % Okidata 410e in 600DPI mode (600dpi)
    mode_param(*pixels_per_inch*, 600);
    mode_param(*blacker*, .6);
    mode_param(*fillin*, .1);
    mode_param(*o_correction*, .85);
    mode_common_setup_;
enddef;
okifourten := *okifte*;

% From `AMSmodes.def`.
mode_def *pcscreen* =   % also, e.g., high-resolution Suns (118dpi)
    mode_param(*pixels_per_inch*, 118);
    mode_param(*blacker*, .5);
    mode_param(*fillin*, .1);
    mode_param(*o_correction*, .3);
    mode_common_setup_;
enddef;

% `fkr@tooyoo1.l.u-tokyo.ac.jp`, 7 June 1995. With the existing
% *bitgraph* and *pcscreen* modes, 'm' looks bad: a long vertical
% line extends higher than the letter itself.
mode_def *pcprevw* =   % PC screen preview (118dpi)
    mode_param(*pixels_per_inch*, 118);
    mode_param(*blacker*, .2);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, .2);
    mode_common_setup_;
enddef;

% Tektronix Color PostScript printer, from `craig@sunspot@noao.edu`
% on 14 January 1993. He writes: This is a thermal wax paper printer.
% The values were determined using the `cmr10` and `cmti10` fonts.
% The generated fonts look reasonable, although vertical lines and
% things like the `[`, `]`, and `/` characters are pretty thin.
mode_def *phaser* =   % Tektronix Phaser PXi (300dpi)
    mode_param(*pixels_per_inch*, 300);
    mode_param(*blacker*, 1.1);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, 1);
    mode_common_setup_;
enddef;

% From `metod.kozelj@rzs-hm.si` (Metod Kozelj), 30 July 1998.
% Parameters other than *blacker* have little effect.
mode_def *phaserfs* =   % Tektronix Phaser 560 (1200dpi)
    mode_param(*pixels_per_inch*, 1200);
    mode_param(*blacker*, 0);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, 1);
    mode_common_setup_;
enddef;
phaserfivesixzero := *phaserfs*;

% Tektronix Phaser 350 is a 600-by-300 colour wax printer.
% From `dag@ifi.uio.no` (Dag Langmyhr), 10 January 1997.
% Perhaps too fat at small sizes (5 pt) but looks OK for 8 pt and more.
mode_def *phasertf* =   % Tektronix Phaser 350 (600x300dpi)
    mode_param(*pixels_per_inch*, 600);
    mode_param(*aspect_ratio*, 300/*pixels_per_inch*);
    mode_param(*blacker*, 0);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, .6);
    mode_common_setup_;
enddef;

mode_def *phasertl* =   % Tektronix Phaser 350 landscape (300x600)
    *phasertf_*;
    landscape;
enddef;
phasertfl := *phasertl*;

% From Aries Arditi `<aries@play.lighthouse.org>`, 3 February 1998.
% This definition makes one pixel one point, which is convenient when
% you want to image-process the letter images after rendering, and don't

% want to add any device corrections. If you want to grab the images off
% the screen, it's useful to add lines to the definition, as well:
%
%   **mode_param**(*proofing*, 1);
%   extra_endchar:=extra_endchar&"showit";
%   extra_setup := extra_setup&"def openit = openwindow currentwindow from
%   origin to (screen_rows,screen_cols) at (0,50) enddef";
mode_def *pixpt* =   % one pixel per point (72.27dpi)
      mode_param(*pixels_per_inch*, 72.27);
      mode_param(*blacker*, 0);
      mode_param(*fillin*, 0);
      mode_param(*o_correction*, 1);
      mode_common_setup_;
enddef;

% For Hans Hagen and Mikael Sundqvist's use of bitmaps, December 2023.
mode_def *potrace* =   % potrace use in LMTX
            mode_param(*pixels_per_inch*, 2 * 3600);
            mode_param(*blacker*, 0);
            mode_param(*fillin*, 0);
            mode_param(*o_correction*, 1);
            mode_common_setup_;
enddef;

% This is a write-white PostScript laser-setter, made by a Xerox
% subsidiary. Its true aspect ratio is 1200 dpi horizontally and
% 600 dpi vertically, but mis@apsedoff.bitnet says that the
% printer hides this, and PostScript programs should treat it as having
% a square aspect ratio. But george@trevnx.bio.dfo.ca says that
% using the nonsquare aspect ratio produces identical output and uses
% only half the disk space. He also says the fonts are much too dark
% in general, and produce invisible diagonals in the CM typewriter
% fonts—but other changes either produce errors or dark output.
%
% Printware's headquarters is in Minnesota; telephone (612) 456-1400.
mode_def *prntware* =   % Printware 720IQ (1200dpi)
      mode_param(*pixels_per_inch*, 1200);
      mode_param(*blacker*, 0);
      mode_param(*fillin*, 0);
      mode_param(*o_correction*, 1);
      mode_common_setup_;
enddef;
PrintwareSevenTwoZeroIQ := *prntware*;
printware := *prntware*;

% From John Gourlay. See *TUGboat* 8(2), page 133.
mode_def *qms* =   % QMS (Xerox engine) (300dpi)
      mode_param(*pixels_per_inch*, 300);
      mode_param(*blacker*, .6);
      mode_param(*blacker_min*, 2);
      mode_param(*fillin*, −.3);
      mode_param(*o_correction*, .6);
      mode_common_setup_;
      mode_write_white_setup_;
enddef;

% From `Boris.Hemkemeier@HRZ.Uni-Bielefeld.De`, 24 June 1993.
% With the QMSOneSevenZeroZero mode, the left stem of 'M'
% in `cmr10` vanishes completely.
mode_def *qmsostf* =   % QMS 1725 (600dpi)
    mode_param(*pixels_per_inch*, 600);
    mode_param(*blacker*, 1);
    mode_param(*blacker_min*, 2);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, 1);
    mode_common_setup_;
    mode_write_white_setup_;
enddef;
QMSOneSevenTwoFive := *qmsostf*;

% From `queinnec@geant.cenatls.cena.dgac.fr`, 24 March 1993.
% `k316670@aearn.bitnet` says this print has a CanonNX engine
% switchable between 300 and 600 dpi.
%
% From `mimi@scri.fsu.edu` (Mimi Burbank), 12 September 1996:
% . . . When I found the note about the left stem of the 'M'
% disappearing I was concerned.
%
% The error, I believe, is due to the fact that the font is generated at
% 600 dpi, and was most likely printed on a QMS printer with 300 dpi
% resolution. I just had the same thing happen to me, but with our QMS
% 860 set at 600dpi (the default for only one of our printers) the
% output was beautiful! (I printed the same ps file on a QMS 2000 with
% 300 dpi resolution, and on a QMS 860 with 600 dpi resolution.)
mode_def *qmsoszz* =   % QMS 1700 (600dpi)
    mode_param(*pixels_per_inch*, 600);
    mode_param(*blacker*, .2);
    mode_param(*blacker_min*, 2);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, 1);
    mode_common_setup_;
    mode_write_white_setup_;
enddef;
QMSOneSevenZeroZero := *qmsoszz*;

% From `teddy@fukt.hk-r.se`, 28 September 1996.
mode_def *qmstftf* =   % QMS 2425 (1200dpi)
    mode_param(*pixels_per_inch*, 1200);
    mode_param(*blacker*, .3);
    mode_param(*fillin*, .5);
    mode_param(*o_correction*, 1);
    mode_common_setup_;
enddef;
QMSTwoFourTwoFive := *qmstftf*;

% These values from Stan Osborne, *TUGboat* 8(2), page 134.
mode_def *ricoh* =   % e.g., TI Omnilaser (300dpi)
    mode_param(*pixels_per_inch*, 300);
    mode_param(*blacker*, .2);
    mode_param(*blacker_min*, 2);
    mode_param(*fillin*, −.2);

mode_param(*o_correction*, .5);
        mode_common_setup_;
        mode_write_white_setup_;
enddef;
RicohFourZeroEightZero := *ricoh*;
RicohFortyEighty := *ricoh*;

% From `Martin.Ward@durham.ac.uk`. Apparently the engine is
% different from the Ricoh 4080. With a larger value of *blacker*,
% characters like the 'e' in `cmtt8` look bad.
mode_def *ricoha* =   % e.g., IBM 4216 (300dpi)
        mode_param(*pixels_per_inch*, 300);
        mode_param(*blacker*, .2);
        mode_param(*blacker_min*, 2);
        mode_param(*fillin*, 0);
        mode_param(*o_correction*, .75);
        mode_common_setup_;
        mode_write_white_setup_;
enddef;
RicohA := *ricoha*;
IBMFourTwoOneSix := *ricoha*;

% From John Sauter.
mode_def *ricohlp* =   % e.g., DEC LN03 (300dpi)
        mode_param(*pixels_per_inch*, 300);
        mode_param(*blacker*, .65);
        mode_param(*blacker_min*, 2);
        mode_param(*fillin*, −.2);
        mode_param(*o_correction*, .5);
        mode_common_setup_;
        mode_write_white_setup_;
enddef;
RicohLP := *ricohlp*;
LNOthree := *ricohlp*;
LNZeroThree := *ricohlp*;

% From `nishida@src.ricoh.co.jp` (Akihiro Nishida), 30 August 1996.
% These printers are available only in Japan.
mode_def *ricohsp* =   % Ricoh sp10ps/lp7200-ux (600dpi)
        mode_param(*pixels_per_inch*, 600);
        mode_param(*blacker*, 0);
        mode_param(*fillin*, 0.2);
        mode_param(*o_correction*, .6);
        mode_common_setup_;
enddef;

% From `dickson@eeserv.ee.umanitoba.ca`. `gil.cc.gatech.edu`
% has different values; `img@ai.edinburgh.ac.uk` sets *blacker* = .1.
% Corrected by `andy@vlsi.cs.caltech.edu`, 28 August 1991.
% The darkness knob on the printer has a much larger effect than
% any of these parameters. `carlos@snfep1.if.usp.br` points out
% that the printer can operate at either 300 dpi or 400 dpi, and
% if your fonts don't match the setting, naturally they won't look
% very good. He says the following works in Dvips' `config.ps` file
% to set 400 dpi:

```
% /SetResolution {
%     /setres where {
%         /setres get exec
%     }{
%         pop
%     } ifelse
% } def
% %%BeginFeature *SetResolution 400
% 400 SetResolution
% %%EndFeature
% %%EndSetup
%
```
% (This is the file `resolution400.ps` supplied with NeWSprint.)
% `simpson@math.psu.edu` only got this work by downloading the code
% via an extra header file, i.e., having this in the Dvips config file:
```
% M sparcptr
% D 400
% h resolution400.ps
%
```
mode_def *sparcptr* =   % Sun SPARCprinter (400dpi)
    mode_param(*pixels_per_inch*, 400);
    mode_param(*blacker*, .25);
    mode_param(*fillin*, .2);
    mode_param(*o_correction*, .6);
    mode_common_setup_;
enddef;
SparcPrinter := *sparcptr*;

% From `ee@dacth51.bitnet`.
mode_def *starnlt* =   % Star NL-10 (240x216dpi)
    mode_param(*pixels_per_inch*, 240);
    mode_param(*aspect_ratio*, 216/*pixels_per_inch*);
    mode_param(*blacker*, 0);
    mode_param(*fillin*, .2);
    mode_param(*o_correction*, .4);
    mode_common_setup_;
enddef;
StarNLOneZero := *starnlt*;

mode_def *starnltl* =   % Star NL-10 landscape (216x240dpi)
    *starnlt_*;
    landscape;
enddef;

% From `alejolo@sue.ideam.gov.co`, 26 November 1998. I have tested
% the default stylewriter mode in modes.mf v3.4 with OzTEX and my
% StyleWriter II, and still output is too light, particularly the serifs
% and thin cusps such as in CMR's small e, c, t, b and d. Thus I cooked
% up this mode that prints output similar to a standard system font (I
% compared text output with Minion Web as it comes with Internet
% Explorer 4). In general I'd suggest that this mode definition is
% appropriate for all inkjet printers using a BJC-02 ink cartridge.
mode_def *styletwo* =   % Apple StyleWriter II (360dpi)
    mode_param(*pixels_per_inch*, 360);
    mode_param(*blacker*, 0.25);

    mode_param(*fillin*, 0);
    mode_param(*o_correction*, 0.6);
    mode_common_setup_;
enddef;
swtwo := *styletwo*;

% *stylewriter* mode added by Andrew Trevorrow
% `<akt@netspace.net.au>` for OzTEX users. All
% parameters (except *pixels_per_inch*) are the same as the *cx* mode so
% that PK files can be shared by both types of printers.
%
% With *blacker* = 0, hbar is indistinguishable for h, i.e., the bar
% disappears. Thus 0.1. From Wulf Hofbauer `<wh@echo.chem.TU-Berlin.DE>`,
% 5 June 1998.
mode_def *stylewri* =   % Apple StyleWriter (360dpi)
    mode_param(*pixels_per_inch*, 360);
    mode_param(*blacker*, 0.1);
    mode_param(*fillin*, .2);
    mode_param(*o_correction*, .6);
    mode_common_setup_;
enddef;
stylewriter := *stylewri*;
stylewr := *stylewri*;
% From `px@fct.unl.pt (Joaquim Baptista [pxQuim])`. I find
% *epstylus* far too dark. It seems to me that plain values of 0 to
% *blacker* and *fillin* work perfectly with values of *o_correction* in
% the range of .6 to .8. I ended up using [this mode:]
epstylwr := *stylewri*;
% Andrew defines *sw* as well, but I am reluctant to use such a
% potentially common identifier –`kb@cs.umb.edu`, 8 October 1996.

% From `grunwald@foobar.colorado.edu`. Sun monitors have several
% different resolutions, but this seems the most common of the lot.
% Use *pcscreen* for high-resolution monitors.
mode_def *sun* =   % Sun and BBN Bitgraph (85dpi)
    mode_param(*pixels_per_inch*, 85);
    mode_param(*blacker*, .35);
    mode_param(*fillin*, .1);
    mode_param(*o_correction*, .3);
    mode_common_setup_;
enddef;

mode_def *supre* =   % Ultre*setter (2400dpi)
    mode_param(*pixels_per_inch*, 2400);
    mode_param(*blacker*, 0);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, 1);
    mode_common_setup_;
enddef;

mode_def *toshiba* =   % Toshiba 13XX, EpsonLQ (180dpi)
    mode_param(*pixels_per_inch*, 180);
    mode_param(*blacker*, 0);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, .2);

mode_common_setup_;
enddef;
epsonlq := *toshiba*;

mode_def *ultre* =   % Ultre*setter (1200dpi)
    mode_param(*pixels_per_inch*, 1200);
    mode_param(*blacker*, 0);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, 1);
    mode_common_setup_;
enddef;
Prism := *ultre*;

% From `Martin.Ward@durham.ac.uk`.
mode_def *vs* =   % VAXstation monitor (78dpi)
    mode_param(*pixels_per_inch*, 78);
    mode_param(*blacker*, 0);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, 0);
    mode_common_setup_;
enddef;
VAXstation := *vs*;
gpx := *vs*;

% From `erikjan@icce.rug.nl`, 23 August 1991.
mode_def *vtftzz* =   % Varityper 4200 B-P (1800dpi)
    mode_param(*pixels_per_inch*, 1800);
    mode_param(*blacker*, 0);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, 1);
    mode_common_setup_;
enddef;
VarityperFourTwoZeroZero := *vtftzz*;

% From `mjm@as.arizona.edu`, 26 February 1992.
mode_def *vtftzzhi* =   % Varityper 4300P (2400dpi)
    mode_param(*pixels_per_inch*, 2400);
    mode_param(*blacker*, 0);
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, 1);
    mode_common_setup_;
enddef;
VarityperFourThreeZeroZeroHi := *vtftzzhi*;

% From `mjm@as.arizona.edu`, 26 February 1992.
mode_def *vtftzzlo* =   % Varityper 4300P (1200dpi)
    mode_param(*pixels_per_inch*, 1200);
    mode_param(*blacker*, 2); % used to be 3.5, see *lexmarkr* comments.
    mode_param(*fillin*, 0);
    mode_param(*o_correction*, 1);
    mode_common_setup_;
enddef;
VarityperFourThreeZeroZeroLo := *vtftzzlo*;

% From `rocky@panix.com`. This can also be used for the Autologic's
% APS6 cut sheet dry process printer. For that printer, perhaps

% *blacker* = 0.8 is better. For the Varityper, though, at *blacker* = 0.8
% the dots of the umlaut start to fill in. For *blacker* < 0.6, the tops
% and bottoms of lowercase g's and s's in `cmr5` drop out.
mode_def *vtfzszw* =   % Varitype 5060W, APS 6 (600dpi)
  mode_param(*pixels_per_inch*, 600);
  mode_param(*blacker*, .7);
  mode_param(*fillin*, 0);
  mode_param(*o_correction*, 1);
  mode_common_setup_;
enddef;
VarityperFiveZeroSixZeroW := *vtfzszw*;
APSSixMed := *vtfzszw*;

% The worst problem is toner irregularity. This may be the same printer
% as the IBM 4250.
mode_def *vtszz* =   % Varityper Laser 600 (600dpi)
  mode_param(*pixels_per_inch*, 600);
  mode_param(*blacker*, 0);
  mode_param(*fillin*, 0);
  mode_param(*o_correction*, 1);
  mode_common_setup_;
enddef;
VarityperSixZeroZero := *vtszz*;
VTSix := *vtszz*;
varityper := *vtszz*;

% Some information about Xerox printers, from `siemsen@barnard.usc.edu`:
% The Docutech system and the 4135 have the same engine.
% The 4050, 4075 and 4090 have the same engine.
% The 4650 has a unique engine.
% The 4850 has a unique engine.

% From `SamuelKey@comcast.net`, for enhanced resolution mode. In
% 600x600 mode, *ljfour* works ok.
mode_def *xpstzz* =   % Xerox Phaser 6200DP (2400x600dpi)
  mode_param(*pixels_per_inch*, 2400);
  mode_param(*aspect_ratio*, 600/*pixels_per_inch*);
  mode_param(*blacker*, 0);
  mode_param(*fillin*, 0);
  mode_param(*o_correction*, 1);
  mode_common_setup_;
enddef;
XeroxPhaserSixTwoZeroZeroDP := *xpstzz*;

mode_def *xpstzzl* =   % Xerox Phaser 6200DP landscape (600x2400dpi)
  *xpstzz_*;
  landscape;
enddef;

% From `u12570@uicvm.uic.edu`. These values are mostly guesses.
mode_def *xrxesnz* =   % Xerox 8790 or 4045 (300dpi)
  mode_param(*pixels_per_inch*, 300);
  mode_param(*blacker*, 0.4);
  mode_param(*blacker_min*, 2);
  mode_param(*fillin*, 0);

```
        mode_param(o_correction, 0.2);
        mode_common_setup_;
        mode_write_white_setup_;
enddef;
XeroxEightSevenNineZero := xrxesnz;

% From u12570@uicvm.uic.edu. Many variations for different fonts.
% bart@cs.tamu.edu says this works for the Xerox 4700, also.
mode_def xrxfzfz =                                      % Xerox 4050/4075/4090/4700 (300dpi)
        mode_param(pixels_per_inch, 300);
        mode_param(blacker, .7);
        mode_param(fillin, 0);
        mode_param(o_correction, .5);
        mode_common_setup_;
enddef;
XeroxFourZeroFiveZero := xrxfzfz;

% From u12570@uicvm.uic.edu. He sent many variations of this,
% for different fonts. I don't know a reasonable way to put them in
% yet, so this is just the basic entry.
mode_def xrxnszz =                                      % Xerox 9700 (300dpi)
        mode_param(pixels_per_inch, 300);
        mode_param(blacker, .7);
        mode_param(fillin, 0);
        mode_param(o_correction, .5);
        mode_common_setup_;
enddef;
XeroxNineSevenZeroZero := xrxnszz;

% From lee@sq.com. These values may be improvable.
mode_def xrxtszz =                                      % Xerox 3700 (300dpi)
        mode_param(pixels_per_inch, 300);
        mode_param(blacker, .85);
        mode_param(blacker_min, 2);
        mode_param(fillin, −.1);
        mode_param(o_correction, .5);
        mode_common_setup_;
        mode_write_white_setup_;
enddef;
XeroxThreeSevenZeroZero := xrxtszz;

mode_def help =                                         % What modes are available?
  for i = 1 upto number_of_modes:
        message mode_name[i];
  endfor;
  % Doesn't make sense to be able to do this twice, so forget this
  % definition after it's been used.
  save?;
enddef;

let mode_help = help_;

% These variables determine the size of METAFONT's (window system)
% window for online output. These numbers should match whatever
% the window system is told, or bizarre positioning of the output
% in the window results. Properly implemented online device drivers
```

% will use these values as the default size. The defaults here are
% from `plain.mf`.
$screen\_rows := 400;$
$screen\_cols := 500;$

% The mode most commonly used to make fonts here.
$localfont := ljfour;$