名前

tgif − 階層的な作図作業が可能で, 複数図面を同時に扱うことも容易な作図環境 (X11用にXlibを使って実現). 他のアプリケーションの起動にも利用できます.

形式

tgif [**-display** ディスプレー名] [**-fg** <前景の色>] [**-bg** <背景の色>] [**-bd** <枠の色>] [**-rv**] [**-nv**] [**-bw**] [**-cwo**] [**-kinput**] [**-xim**] [**-geometry** <位置>] [=<**位置**>] [<*file*>[.*obj*]]

または

tgif -print [**-eps**] [**-p**] [**-ps**] [**-f**] [**-gray**] [**-adobe** | -adobe=<数>/<数>] [**-o** *dir*] [*file1 file2* ...]

機能説明

*Tgif*を使うことで, Xウィンドウ上で作図できます.作図する場合は1番めの形式によって, また作図せずにただ図面を印刷したいだけの場合は2番めの形式によって起動します (図面を印刷する場合に, 以前は*prtgif*という専用プログラムを使っていました).

1番めの形式によって起動した場合は, *-fg* (*f*o*r*eground colour), *-bg* (*b*ack*g*round colour), および *-bd* (*bo*r*d*er colour) によって, それぞれ前景の色, 背景の色, および枠の色を指定できます. Tgifで作図しかけの図面の作図をつづけようという場合は, その図面を収めてあるファイル名*file*を起動時に指定して下さい.図面の色を白黒反転するには*-rv* (*r*e*v*erse-*v*ideo)を指定します (逆に *-nv* (*n*o*r*mal-*v*ideo) を指定すると白黒を反転しません).白黒以外の色を使わない場合は*-bw* (*b*lack and *w*hite) を指定します.図面だけを表示し他の何も表示しないようにするには, *-cwo* (canvas *w*indow *o*nly) を指定します. これはXのリソース定義でTgif*CanvasWindowOnlyにtrueを設定するのと同じことです(なにも指定しない場合はfalseとみなします).なおtgifは*Xkb*だけを使って実現し, その動作はX11-R4で試しました.使用に際してはボタンの3つついているマウスが必要です.

2番めの形式によって起動した場合は, (*tgif*を使って作成した) ファイル*file1.obj, file2.obj*等々の印刷イメージを順次 PostScript(TM)形式のファイルに出力します. さらに**-eps**, **-p**, **-ps**のいずれも指定していない場合は, このファイルを*lpr*に渡して印刷します. このとき, 上記以外の指定のある場合には, これを*lpr*に対する指定として扱います. なお (詳細については後述の) シンボル・ファイルも, 拡張子*.sym*を明示的に指定すれば印刷できます.

-epsあるいは**-p**を指定して2番めの形式によって起動した場合は, EPS形式のファイル*file.eps*を作成します. このファイルは\psfig, \epsf, \psfileといった命令を使ってLaTeX形式の原稿に組み込めます (詳細については【LATEXへの図形の組み込み方法】の項で解説します). **-ps**あるいは**-f**を指定した場合は, PostScript形式のファイル*file.ps*を作成します. このファイルはlprコマンドなどを使えばPostScript印刷装置で印刷できます (**-eps**, **-p**, **-ps**, あるいは**-f**のいずれかを指定した場合, それ以外の指定は仮にあったとしても無視します). *-gray*を指定することと, Xのリソースとして Tgif*UseGrayScaleを指定することは同じことです (詳細は【Xのリソース】の項を参照して下さい). *-adobe*あるいは*-adobe*=<数>/<数>を指定することは, Xのリソースとして Tgif*UsePsAdobeStringを指定することと同じことです (詳細は【Xのリソース】の項を参照して下さい). *-o*を指定していない場合は, 出力する (*file.eps*または *file.ps*といった) ファイルを入力ファイルと同じディレクトリに作成しますが, *-o dir*を指定した場合は, 指定のディレクトリ*dir*に作成します.

なお, X11ビットマップあるいはX11ビックスマップ形式のファイルを作成するには, 最初の形式で起動し手動で印刷する方法しかないので注意して下さい.

基本機能

扱うことのできる基本的な図形は, 四角形, 円弧, 角の丸い四角形, 扇型, 折れ線, 多角形, 開いた (輪になっていない) スプライン曲線, 閉じた (輪になっている) スプライン曲線, 文字列, X11ビットマップ, X11ビックスマップの一部, EPSファイルです.複数の図形をまとめて**グループ化**できます. 基本図形やグループ化した図形を元に, その図形を表現部とし定義部が空であるような新しい部品を作成し, 作成した**記号**で元になった図形を置き換えたり, 新しい部品を作るために, 図形を**シンボル**として指定できます.

図面を書き込むファイルは, 以下の2種類のいずれかの形式です. 一つは図形を収めたファイルで, (物 (*object*)ファイル, という意味で) *.obj*という拡張子を付けます. もう一つは「部品」を収めたファイルで (部品を象徴している**シンボル** (*symbol*) を含んでいるファイル, という意味で) *.sym*という拡張子を付けます.複数の図面ファイルを**切替える**ために, **瞬間切替え**と名付けた方法を用意してあります.部品は, **表現部**および**定義部**からなります (定義部には何も書かれていないということもあり得ます).部品を「置く」機能があるので, 階層的な図形を, 細かな部分から段々と全体を作図してゆく流儀によって作図できます.一方任意の図形をもとに新しい部品を作成しておき, 後からその定義部を作図することで, 段階的に詳細化してゆく流儀によって作図するといった使い方もできるようになっています.図面を収めるファイルの, 2種類ある形式のどちらもPrologの式の形になっていますから, 図面を解釈するPrologのプログラムを書くことも可能です (このためのプログラムを自動的に生成するわけではありません. この点についての詳細は【PROLOGあるいはCによるドライバの例】の項を参照して下さい).すなわち, 後述の**ドライバ**と呼んでいるPrologプログラムをProlog処理系に渡して, 図形の処理をできるわけです (無論, これ以外の, 例えばCで書いたドライバによって図形を処理できます).

文字列以外の図形に対して, 文字列を**属性**として与えられます. 但し部品に対して定義した属性を, 一旦図面上に置いた後で取り去ることはできません. 詳しくは【属性】の項を参照して下さい.

出力できるファイルの形式は, 全部で4種類あります. 何も指定しない場合は PostScript(TM)の形式です (カラーPostScriptにも対応しています). これは一旦/tmp/Tgifa*という名前の (mktemp()ライブラリ手続きによって作った)ファイルに収め, それからlprに渡します (*のところには数字が入ります). PostScript形式の場合は, 選択欄 (tgifの上に表示されている各欄の名前は後に書いてあります) にレーザー印刷装置の絵を表示してあるはずです.また選択欄に*PS*の絵が表示してある場合は, lprに渡さずに *.ps*という拡張子をもつファイルに出力します.選択欄に*LaTeX*の絵が表示してある場合は, *.eps*という拡張子を持つファイルに出力します. このファイルはEPS形式で, *\psfig*あるいは*\epsf*命令を使ってLaTeXの原稿に組み込めます.組み込む方法については後述します.選択欄に*X11BM* (X11ビットマップ) の絵が表示してあり, かつ**カラー出力を選択していない**場合は, *.xbm*という拡張子を持つファイルに出力します. このファイルはX11ビットマップ形式です.選択欄に*X11BM* (X11ビットマップ) の絵が表示してあり, かつ**カラー出力を選択**した場合 (カラー出力を選択するには, キーボードで ^#K と押せばすればよく, またここに ^ はコントロール, # はメタキーのことであるとします) は, *.xpm*という拡張子を持つファイルに出力します. このファイルはX11ビックスマップ形式です (XPMファイルの形式のバージョンは, Xのリソース XPmOutputVersionの設定によります). X11ビットマップ, ある種のX11ビックスマップ (例えばtgifによって出力したもの, など, 詳しくは【X11ビックスマップ】の項を参照), EPSファイルは tgifの基本図形として図面の中に**取り込め**ます.

tgifによる図面はA4 (横210mm, 縦296mm) です. 紙を縦長, あるいは横長方向のいずれもに使うことも可能です. キーボードで #% と押すことによって縮小あるいは拡大を指定すれば, 図面を縮小 (あるいは拡大) できます. tgif をコンパイルした時に (Imakefile あるいは Makefile.noimake中の) -DA4PAPER の定義を外してあれば, 出力図面の大きさはレターサイズ (横約 216mm, 縦約280mm) になります.

図形

図面ファイル (*.obj*) に書き込むめるのは, 基本図形, グループ化した図形, それに**記号**です.**シンボル**ファイル (*.sym*) に収められるのは, 図面ファイルに収められる図形任意個と, 一つの**シンボル**です (一つのシンボルファイルが一つの部品を表しているということに注意して下さい).シンボルファイル中のシンボルは, 部品の表現部です. またシンボル・ファイルのシンボル以外の図形は, 部品の定義部になります.その他の図形と区別するために, シンボルを点線で囲んで強調するようになっています. 部品を図面上に置くと, 部品中のシンボル (表現部) を図面上に転写します. 図面上に転写したシンボルのことを, (図面上で部品を表している) 「記号」と呼びます.

Tgif中の図形は, 場所を動かしたり, 複製を作ったり, 削除したり, 回したり,向きを反転したりできます (注意: 文字列を水平方向に反転すると文字列の行揃えの状態が変わるだけですし, 文字列を垂直方向に反転すると,大抵の場合は位置が変わるだけです). 文字列と固まった記号以外の図形は,伸び縮みさせられます (スケーリング) (固まった記号については【TGIFの操作】の項を参照して下さい).

Tgifでは32種類の塗りつぶし模様, 32種類のペン先, 7種類の線幅 (増やすことも可), 4種類の線の形 (直線, 頭に矢印, 尾に矢印, 頭と尾の両方に矢印), 3種類の文字列の行ぞろえ, 4種類の字体 (ローマン体, イタリック体, ボールド体, ボールドイタリック体), 11種の文字の大きさ (75dpiの大きさのフォントについては8, 10, 12, 14, 18, 14ポイント, 100dpiの大きさのフォントについては11, 14, 17, 20, 25, 34ポイント), 5種類の英語フォント (タイムズローマン, クーリエ, ヘルベチカ, New Century Schoolbook, 記号), 2種類2方向の日本語フォント (横書き, 縦書き用の明朝体, ゴシック体), 11種類の色 (マゼンタ, 赤, 緑,青, 黄, ピンク, シアン, くすんだ青緑, 白,黒, 暗い灰, 増やすことも可能)を使えます. 図形の回転については, 直角 (90度) 単位だけです.

ポップアップメニューから選ぶか, あるいはキーボードから入力することによって, ほとんど全ての操作を行なえます.図形を何がしか変更する操作は, どのような操作でも, その操作をなかったことにしたり, やり直したりできます. 図面を拡大表示したり, 図面の表示位置をずらしたり, 文字列を選択していない状態で

フォントの選択を替えたりする, といった操作をなかったことにはできません.いくつ前の操作までなかったことにできるのかは, Xのリソース Tgif*HistoryDepthに指定できます.

TGIF画面上の欄の種類と操作

Tgifの画面上に表示してある各々の欄の種類と, それぞれの欄における操作について以下に解説します.

一番上の欄

現在のドメインとファイル名を表示してあるはずです. ここをマウスでクリックしたり, キーボードを押したりしても何も起きません.

表示欄 一番上の欄のすぐ下の左側の欄です. tgifの状況を表示するための欄です.この欄でマウスの左ボタンをクリックすると表示をずらして先に進みます.逆に右ボタンをクリックすると, 表示をずらして前に戻ります. 中央のボタンをクリックするか, 中央ボタンを使ってマウスをドラッグすると, それに合わせて表示を前後にずらせます.

選択欄 一番上の欄のすぐ下の右側の欄です. ここには, tgifの状態を表している様々の印が表示されているはずです. 左上が図形の種類, その下が紙の向き (縦長か横長か), 以下順に, EDIT (後で解説します), 印刷の形式, 図面の拡大率,制約付の移動 (と拡大縮小), 角の丸い四角形の角の丸さ加減, 文字列の方向,ページ番号あるいは行とカラム位置, ページ表示方法 (積み重ねあるいはタイル状), 水平方向の位置合わせ (L C R S -) , 垂直方向の位置合わせ (T M B S -), フォント, 文字の大きさ, 文字列の行間幅, 行ぞろえ, 点線の形, 線分の形, 折れ線かスプライン曲線か, 線分の幅, 塗りつぶし模様, ペンの形, 色,その他(後で解説します) を意味しています. この欄内でキーボードを押しても何も起きません.

選択欄に表示されている印をクリックすることで, tgifの状態を変えられます.印は, それぞれ何らかのtgifの状態を表しています. 印を左ボタンでクリックすると, 状態を変化させ, 印の表示を新しい状態を表す印と代えます. 右ボタンをクリックすると, 状態を逆方向に変化させます.印の上で中央ボタンを使ってマウスをドラッグすると, 大抵の場合は選択できる状態の一覧を表示します. 選択欄中の印は, それぞれ図面欄のメニュー中の項目と対応しており, ある印の上で中央ボタンをクリックした時に表示する一覧は, 図面欄のメニュー中の対応する項目を選択した時に表示する一覧と同じものです (先に述べた「EDIT (編集)」および「その他」の印の上で右あるいは左ボタンをクリックしても何も起こりません. これは, この2つの印が図面欄のメニュー中の同名の項目を表すために便宜的に設けたものだからです).中央ボタンを押した時, 図面の拡大率, 角の丸さ加減, 行間の印に関しては,異なる動きをします. これらの印の上で中央ボタンを押しながらマウスを左あるいは上にずらすと数字が増え, 逆に右あるいは下にずらすと数字が減ります.

図面欄中のある図形を選択した状態で選択欄上の操作を行なった場合, その選択した図形の状態を変えることになります. この際, 選択欄上の操作をマウスの中央ボタンを使って行なえば, tgifの状態は変化しません.

水平方向および垂直方向の位置合わせによって影響を受けるのは, 次の通りです. 第一はキーボードから ^l を押した時に複数の図形あるいは頂点をどのように位置合わせするか, ということ. 第二は ^t を押した時に図形あるいは頂点を格子線に対してどのように配置するか, ということ. 第三は #l を押した時に複数の図形を相互にどのように配置するのか, ということ. 第四は ^#u を押した時にそれぞれ記号を最新のものとどのように入れ換えるのか, ということです.水平方向の位置合わせは, 左そろえ (L), 中ぞろえ (C), 右そろえ (R), 空白 (S) および位置合わせしない (-) のいずれかです. 垂直方向の位置合わせは上そろえ (T), 中揃え (C), 下そろえ (B), 空白 (S) および位置合わせしない (-) のいずれかです. 一般的に位置合わせでは空白 (S) と位置合わせしない (-) は同じことです. 空白の指定は, 複数図形を配置する場合に, 図形を等間隔に並べる場合に用います. 頂点モードでは, 位置合わせしない, 以外を指定している場合, 選択している頂点を等間隔で並べることになります.これらの動作について完全に理解するには, なにはともあれ試してみることを勧めます.

文字列の行間を設定すると, 文字列の入力中に改行を押したときの改行幅が自動的に決まります. 2つの行が完全に重なってしまうほどこの値を負の大きな値に設定することはできません. また, 実際の空白がどの程度のものになるのかは, フォントの種類と大きさによって変わります.

図面欄 ここで作図します. マウスでクリックしたり, マウスを動かしたりした時に何が起こるのかは, 現在図形の種類として何を指定しているのかによって決まります. 一般的にいって, 右ボタンを使ってマウスをドラッグすると, 図形の種類一覧を表示します. 図形の種類には, 一覧表の上から

図形の選択, 文字列, 四角形, 円弧, 折れ線 (あるいは開いたスプライン曲線), 多角形 (あるいは閉じたスプライン曲線), 扇型, 角の丸い四角形, 手書きの折れ線 (あるいは手書きのスプライン曲線), 頂点の選択があります.

図形の選択を指定している場合, 左ボタンによって図面欄上の図形を選択したり, 位置を動かしたり, 伸縮したり, 変形したりできます (頂点の選択を指定している場合にダブルクリックすると, 全ての選択している図形の選択を解除できます).シフトキーを押したまま選択していない図形をクリックすると, (今まで選択した図形は選択した状態のまま) 新たにその図形も選択した状態にできます.選択した状態の図形に対して同じことをすると, その図形だけ選択を解除できます.取手をつまむことで, 多角形を変形できます. それ以外の図形の場合は, 取手をつまむことで (縦方向, 横方向, あるいは縦横両方向に) 伸縮できます. 図形を伸縮する際, 取手を**つまんだ後**にキーボード上でシフトを押すと,縦横比例させた伸縮になります (図形の大きさだけを変える操作です). 文字列や固まった記号は伸縮したり拡大縮小したりできません (なお固まった記号とは, その値が"not_rigid"であるemptyという名前の属性を持っていないものです. 但し, ある固まってない記号の中に別の記号が含まれている場合には,含まれている記号が固まったものであっても, 全体として固まっていないものとして扱います).キーボード上でシフトを押しながら中ボタンをクリックすると, 瞬間切替え機能を起動します. 詳しくは【瞬間切替え】の項を見てください.矢印キーを使って図形を動かせます. また, 図形を一つも選択していない時には, 矢印キーを使って図面の表示部分を少しずつ変えられます. <コントロール>キーを押しながら矢印キーを使うことで, 図面の表示部分を一画面分ずつ変えられます.

頂点選択モードでは, 左マウスボタンを使って, 頂点を選び, 選んだ頂点を動かせます. 頂点選択モードで操作できるのは, 頂点選択モードにしたときに選択してあった, グループ化していない折れ線または開いたスプライン曲線,あるいは多角形または閉じたスプライン曲線です. 頂点選択モードでは, 操作できる図形の頂点に正方形の印がついているはずです. 既に述べた図形選択と同様の操作によって頂点を選ぶと, その頂点を選んだことを示す印として,この頂点には「+」の印を付けます. これら「+」のついた頂点は, 動かしたり,消したり, (お互いの間で) 位置合わせをしたり, (等間隔に) 並べたり, 位置を格子線に合わせたりできます. また矢印キーを使ってこれらの頂点を動かせます.

図形は (#<を押すことで) 操作を禁止 (Lock) できます. 図形の操作を禁止すると,その図形の取手 (とって) は灰色になり, 動かしたり, 拡大したり, 縮小したり, あるいは反転したり, 回転したりできなくなります. 操作を禁止した図形を含む複数個の図形をグループ化すると, グループ化した図形全体も操作できません. 操作を禁止してあっても, 色, フォント, ペン先, などの属性は変えられますし, 消去することさえできます.

制約付の移動, 拡大縮小を指定してある場合に任意の図形を移動したり拡大縮小したりすると, グループ化していない折れ線も, それにつられて以下のように移動したり変形したりします (なお, 制約付の移動, 拡大縮小は, #@を押すことで指定, あるいは指定を解除できます). 図形を移動する場合において,もしその図形の内部に両端が含まれている折れ線があると, その折れ線も図形と一緒に移動します. あるいは, その図形の中に片端のみが含まれている折れ線があると, その含まれている方の端点も図形と一緒に移動します. この際に, 移動した端点の次の点とを結ぶ線分が垂直または水平の場合は, この線分が垂直または水平を保つように, 端点につられて端点の次の点も水平または垂直に移動します. 図形を拡大縮小する場合において, もし端点がその図形の中に含まれている折れ線があると, その端点も図形につられて移動します.移動した端点と, その次の点とを結ぶ線分が垂直または水平の場合は, 前述した移動の場合と同じようになります.

図形の種類として文字列を指定してある場合に (この場合, カーソルは垂直な線分の形になっています), 文字列の上で左ボタンを押すと, その文字列を編集できます. <シフト>を押しながら文字列の上で左ボタンをクリックすると,文字列の一部を反転表示します. 文字列を編集している間に押した文字は,そのまま文字列の一部として挿入することになります. 矢印キーを押すと,それに応じてカーソルを動かせます. <ESC>を押してから文字を押すと, その文字の第7ビットを立てたものとして扱うので, 非 ASCII文字を入力できます.しかし, こういった文字の中には, X11 の上では表示できるがPostScriptでは印字できないものがあり, tgifではそのような文字の入力はできないようになっています. 編集している文字列が図形の属性として定義したものである場合に<メタ><タブ>を押すと, その図形の見えている属性のうちの次のものにカーソルが移動します. <シフト><タブ>の場合は逆に前のものに移動します.

図形の種類として多角形 (閉じたスプラインは含みません) を指定してある場合に, シフトを押しておくと, 多角形をゴムバンドを伸ばしたり縮めたりするようにして操作できます.

図形の種類として手書きの折れ線を指定することで, 折れ線あるいは開いたスプラインを作図できます. 図形の種類として通常の折れ線を指定した場合は, マウスボタンをクリックすることによって折れ線の各頂点を指定しましたが, 手書きの折れ線の場合は, ただマウスを動かすだけで, 自動的に頂点を指定して行けます. 折れ線の終点はマウスボタンを放すことで指定します.

いつでも, 中央のマウスボタンを押すことでメニューを使ってtgifを操作できます. シフトを押しながら右マウスボタンを押すことで, いつでも図形の種類として**選択** を指定した状態にできます. <コントロール>あるいは<メタ>を押しながら文字列を押す (一般的には, こうすることで制御文字を入力できるので, こうして押した文字のことを**非図形文字**と呼びます) ことで, 諸処理を起動できます. どのような処理を起動できるのかについては, 次項に詳しく解説します.通常非図形文字を押すことで起動している諸処理を, **短縮形** として, 1文字を押すことだけで起動できるように指定できます. 詳しくは【短縮形】の項で解説します.

スクロールバー

垂直, あるいは水平スクロールバーの上で左ボタンをクリックすると, それに応じて図面の表示位置が下, あるいは右にずれます. 右ボタンをクリックした場合はその逆です (ファイルやドメインの選択時に現れるスクロールバーについても同様です). シフトキーを押しながらクリックした場合は, 1画面分ずれます. 右ボタンをクリックするか, あるいは右ボタンを使ってマウスをドラッグした場合は, スクロールバー上の灰色の部分として示してある位置を表示します (画面に表示してある図面の左上角は, 実際の印刷図面の左上角から見て右下にずれた点にあります).

定規

定規上の印を見ることで, マウスの現在位置を知ることができます. 定規上でマウスをクリックしたりキーボードを押しても何もおきません. 図面の拡大縮小率が100%の場合は, 定規の目盛はセンチ・メートルです (インチ目盛にすることもできます). 図面の拡大縮小率が100%以外の場合は, 定規の目盛は特に意味を持ちません.

割り込み

XのリソースとしてTgif*IntrCheckIntervalに正の値を設定しておくと, 図形の描画中に表示欄のすぐ下, 水平スクロールバーの左に割り込み可能であることを示す記号を表示します. 図面の描画中にこの記号をクリックすると, 描画を中止させられます. 図面を (Open()やPush()などによって) ファイルから読み込んでいるときも, 図形の描画を中止させられますが, この場合ファイルからの読み込自体を中止するわけではありません (ファイルからは読み込みますが, それを表示しないだけです).

メニュー

メニューを表示したままマウスを (訳注: 大きく) ドラッグすると, そのメニューを**ピン止めしたままにしておけます** (どのくらい動かすとピン止めできるのかは, XのリソースTgif*MainMenuPinDistanceの設定によります).表示したままになっているメニューの上で右ボタンをクリックすると, そのメニューを消せます. 左ボタンを使ってマウスをドラッグすると, メニューの位置を動かせます. また中央のボタンをクリックすることで, メニュー中の該当項目に対応する処理を起動できます.

非図形文字からの諸処理の起動

非図形文字を押すことで, ほとんどの処理を起動できます (一部メニュー上で選ぶか, あるいは短縮形を定義する方法でしか起動できないものがあります).本項では, <コントロール>または<メタ>, あるいはその両方を押しながらキーを押すことによって起動できる処理についてまとめてあります.以下「^」は<コントロール>キーを, 「#」は<メタ>キーを表しているものとします(tgifと一緒に配布しているファイル「*keys.obj*」には, 本項と同じことについてまとめてあります. このファイルはtgifを使って見ることができます. もしtgifを正しくインストールしてあるなら, 【ファイル】の項で説明しているファイル「tgificon.obj」と同じディレクトリにあるはずです).

^a 全選択 [*a*ll]
^b 選択した図形を後ろへ [*b*ack]
^c ドメインを変える [*c*hange]
^d 選択した図形を複製 [*d*uplicate]

^e 描画モードをセーブ/復帰
^f 選択した図形を前面に [*f*ront]
^g 選択した図形をグループ化 (グループ化した後図形を前面に) [*g*roup]

^i 部品を図面上に置く [*i*nstantiate]
^k 図面の上位の階層に戻りシンボルファイルを閉じる (^v の逆) [pop back]
^l 選択した図形の位置を合わせる [*a*lign]
^n 新規に作図を始める [*n*ew]
^o 図面ファイルを開く [*o*pen]
^p 印刷する (あるいはxbm, xpm, eps, ps形式のファイルを作る) [*p*rint]
^q 作業を終える [*q*uit]
^r 図面を描き直す [*r*edraw]
^s 現在編集中の図面を, 図面, あるいはシンボルファイルに書き込む [*s*ave]
^t 選択した図形を格子線に合わせる
^u グループ化を解除 [*u*ngroup]
^v 選択した記号の定義部分を表示
^w 図形の種類として文字列を指定
^x 選択した図形の全部を削除
^y 選択した図形をカットバッファに転写
^z ドライバを起動
^, 左に図面をずらす
^. 右に図面をずらす
^- 印刷する (印刷のための命令を指定する)

#a 選択した図形に, 選択した文字列を属性として付与する [*a*ttach attribute]
#b ドライバを起動
#c 選択した図形を反時計方向に回転 [*c*ounter clockwise]
#d 格子線の間隔を狭く [*d*ecrement]
#e 選択した折れ線の軌跡を示す
#f 選択した折れ線を瞬 (またた) かせる [*f*lush]
#g 格子線を表示する/しない [toggle *g*rid]
#h 選択した図形を水平方向に反転 [flip *h*orizontally]
#i 格子線の間隔を大きく [*i*ncrement]
#j 選択した図形の属性名を隠す
#k 図形の種類として選択を指定
#l 選択した図形を位置合わせの指定に従って配置
#m 選択した図形の属性を動かす, あるいは位置をそろえる
#n 選択した図形の全ての属性名を表示する [attribute *n*ame]
#o 図面の拡大率を1段階小さく [*z*oom *o*ut]
#p 図面ファイルの内容を現在の図面に取り込む [im*p*ort]
#q 図形の種類として折れ線/開いたスプライン曲線を指定
#r 図形の種類として四角形を指定 [*r*ectangle]
#s ドライバを起動
#t 選択した図形の全ての属性を取り去る [de*t*atch]
#u 操作をなかったことに [*u*ndo]
#v 選択した図形を垂直方向に反転 [flip *v*ertically]
#w 選択した図形を時計方向に回転
#x ドライバを起動
#y ドライバを起動
#z 図面の拡大率を1段階大きく [*z*oom in]
#9 円弧を描く (時計の12時の向きを0度として, 半径などを数字で指定できる)
#0 選択した図形を現在のtgifの状態に合わせて描き直す
#, 図面を上にずらす
#. 図面を下にずらす

#- 選択した図形の全ての属性を表示
#[図形の左端をそろえる
#= 図形を水平方向に中ぞろえする
#] 図形の右端をそろえる
#{ 図形の上端をそろえる
#+ 図形を垂直方向に中ぞろえする
#} 図形の下端をそろえる
#" 選択した多角形を正多角形に (もとの図形と同じような大きさに収まるようにする)
#% 印刷時の拡大縮小率を設定 (100以下だと縮小, 100以上だと拡大)
#: 図面の拡大率をデフォルト拡大率に戻す
#‘ ページ全体が一度に見られる大きさまで縮小表示
#~ 新しいファイルに図面を書き込む
#; ビットマップ/ピックスマップ図形のおおきさおよび拡大縮小率を変える
#_ 選択した図形を水平方向に並べる
#| 選択した図形を垂直方向に並べる
選択した文字列を1文字ごとに分割
#^ SaveOrigin()で保存したた原点までスクロール
#@ 移動や拡大に制約あり/なしのモードを切替える
#$ 図形の種類として頂点選択を指定
#& 選択された図形を現在の位置合わせの指定に従って図面に対して位置合わせする
#* 操作をやり直す
#(EPSファイルを取り込む
#) 指定した図形を指定に従って横X, 縦Y倍に拡大
#< 選択した図形を操作禁止に (動かしたり伸ばしたり反転したり回転したりできなくなる)
#> 選択した図形の操作禁止を解除 (操作できるようになる)
^#a 選択した多角形, 折れ線, スプライン曲線に頂点を追加 [*a*dd points]
^#b 文字の形をボールド体に [*b*old style]
^#c 以降はテキストをセンタリング [center]
^#d 選択した多角形, 折れ線, スプライン曲線から頂点を削除 [*d*elete points]
^#e 図形の種類として角の丸い四角形を指定
^#f 指定したビットマップの白黒を逆転
^#g 格子点単位で作図するかどうか細かな作図許すかどうかを切替える [*g*rid points]
^#h 選択した図形の全ての属性を隠す [*h*ide all attributes]
^#i 選択した図形をもとに新しい部品を作り, 作成した記号で置き換える [make *i*conic]
^#j 選択した記号を (普通の) グループ化した図形と置き換える
^#k 出力のカラーあるいは白黒指定を切替える
^#l 文字列を左寄せにするよう指定 [*l*eft justify]
^#m 選択した図形をシンボル化
^#n 選択したシンボルをグループ化した図形に
^#o 字体としてローマン体を指定
^#p 字体として太いイタリック体を指定
^#q 図形の種類として多角形/閉じたスプライン曲線を指定
^#r 文字列を右寄せにするよう指定 [*r*ight justify]
^#s 新しい名前のファイルに図面を書き込む [*s*ave new]
^#t 字体としてイタリック体を指定
^#u 選択した記号を更新 [*u*pdate]
^#v 図形の種類として円弧を指定 [*o*val]
^#w 折れ線とスプラインの指定を切替える
^#x ファイル出力の形式を切替える
^#y カットバッファの中身を貼り込む
^#Y 選択部分 (X selection) から文字列を取り込む
^#z 図形の種類として扇型を指定
^#. X11ビットマップファイルを取り込む
^#, X11ビックスマップファイルを取り込む

^#- 目盛の単位をセンチ・メートルとインチ間で切替える

短縮形

通常非図形文字を押すことで起動している諸処理を, **短縮形**として, 1文字を押すことだけで起動できるように指定できます. 短縮形は, Xのリソース Tgif*ShortCuts に定義します (なお, 短縮形は図形の種類として文字列を指定して**いない**場合に限って使えるものです). 省略形の指定は, 文字 (大文字小文字を区別することもできます) とその文字を押した時に起動したい処理の名前の対を並べたものです.名前の対は空白で区切って並べ, それぞれの名前の対は, 文字とその文字を押した時に起動したい処理の名前を「:」で区切ります.文字は, *:<Key>x*, または*!<Key>x*, あるいは*<Key>x*のいずれかの形式で記述します (ここに「x」は任意の文字を表し, *<Key>*の部分はそのまま記述するものとします).最初の2つの形式は, いずれも**小文字**のxを意味します. 3つめの形式では, 大文字「X」と小文字「x」のいずれでもよいことを意味します.起動したい処理の名前は, tgifのメニューに現れる文字列と同じものです (例外については後述します). 具体例で示しましょう.編集メニューに以下のような2つの項目があるとします.

"Delete ^x"
"SelectAll ^a"

これは, <コントロール>xを押すことによってDelete()の処理を, また<コントロール>aを押すことによってSelectAll()の処理を起動できることを意味しています.ここに出てきたようなDelete()やSelectAll()といった名前を, 短縮形を指定する場合の処理の名前として使えるわけです.従って, 以下のように短縮形を指定すると, 小文字の「x」を押すことで Delete()の処理を, また「a」あるいは大文字の「A」を押すことで SelectAll()の処理を起動できるようになります.

Tgif*ShortCuts: !<Key>x:Delete() \n\
<Key>a:SelectAll()

Xのリソース設定の例として, tgifと一緒に配布しているtgif.Xdefaultsというファイルに, 短縮形定義の例をいろいろと載せてありますから参考にして下さい.

以下に, メニューに現れる文字列と, その項目を選択することで起動できる処理の名前とが異なっているものを一覧としてまとめます. 表の左の列は処理の名前です. また右の列はメニューに現れる文字列です.

CyclePrintFormat() 印刷, LaTeX図形出力, および (生の) PSファイル出力の切替え
ToggleBW/ColorPS() 白黒, カラーの切替え
ToggleGridSystem() センチ・メートル目盛, インチ目盛の切替え
ToggleMapShown() ビット/ビックスマップ表示, 非表示の切替え
ToggleUseGrayScale() 中間階調 (グレースケール) の使用, 不使用の切替え
ToggleMoveMode() 制約付, 制約なしの移動指定の切替え

ToggleLineType() (折れ線とスプラインの指定を切替えます)
ScrollPageUp() (表示部分を上方向に1画面分ずらします)
ScrollPageDown() (表示部分を下方向に1画面分ずらします)
ScrollPageLeft() (表示部分を左方向に1画面分ずらします)
ScrollPageRight() (表示部分を右方向に1画面分ずらします)
FreeHandMode() (手書きと多角形の指定を切替えます)
CenterAnEndPoint() (折れ線の端点を別の図形の中心にもってゆきます)
ToggleNamedAttrShown(<x>=) (属性<x>の表示/非表示を切替えます)

以下も処理名として指定できます (訳注: [かぎかっこ]内はその意味です). ScrollLeft() [表示部分を左へずらします], ScrollRight() [表示部分を右へずらします], ScrollUp() [表示部分を上へずらします], ScrollDown() [表示部分を下へずらします], SelectMode() [図形の種類を選びます], DrawText() [文字列を描きます], DrawBox() [四角形を描きます], DrawOval() [円弧を描きます], DrawPoly() [折れ線を描きます], DrawPolygon() [多角形を描きます], DrawRCBox() [角の丸い四角形を描きます], DrawArc() [扇型を描きます], SelectVertexMode() [頂点モードにします].

属性

属性とは, **名前=値**あるいは**値**という内容の文字列で, 文字列以外の任意の図形に対して与えられます.属性を与えたり取り去ったりすることは自由にできますが, 以下の場合は例外です.

すなわち, 一旦 (部品を積み上げ) 記号として図面上に置いた後では, その部品に対して与えてある属性を取り去ることはできません. なぜなら, これら記号に付随する属性は, 元となる部品から

引き継いだものであると考えられるからです (もしどうしても属性を取り去りたいのであれば, 一旦メニュー Special中のUnMakeIconic()を使って記号をグループ化した図形に置き換えて下さい. こうすれば属性を取り去ることはできます).

属性の全体を表示したり, あるいはその名前部分だけを表示したりするかどうかを指定できます (メニューspecialにある関連命令, 例えば#m, #n, #j, #-, ^#hなどを使います).

瞬間切替え

Tgifでは複数の図面ファイルを切替えながら編集作業を進められます.ある図形の上でシフトキーを押しながら中央のマウスボタンをクリックすると, (標準の設定では) その図形の*warp_to* という名前の属性を探します.その名の属性があれば, その属性の値をもつ図面ファイル名を読み込みます.もし図形に色違いの*warp_to*という属性が複数個ある場合は, 選択窓に見えている色と同じ色の属性の指定に従います.製図中の図面を変更した場合, 新しい図面ファイルを読み込む前に現在の図面をファイルに書き込むか確認します.もし図形に色違いの*warp_to*という属性が複数個ある場合は, 選択窓に見えている色と同じ色の属性の指定に従います. *warp_to*属性の値が「/」から始まっている場合は, 値は絶対パス名として扱います. そうでない場合は, 相対パス名として扱います.

他のアプリケーションの起動

他のアプリケーションをtgifの中から起動できます.ある図形の上で<シフト>と一緒に中央のマウスボタンをクリックすると, その図形に与えてある属性のうち (標準設定では) *launch*という名前のものを探し, もしそういう属性が見つかると, その値部分の文字列をsh(1)に渡して処理します. 色の指定については, 瞬間切替えと同様です.もし文字列の最後の文字が「&」であれば, 子プロセスを (forkで) 作り, その子プロセスからアプリケーションを起動します. もし文字列の最後の文字が「&」でなければ, (popen()を使って) アプリケーションを起動します (この場合, もしアプリケーションが終了せず, しかもそのアプリケーションを強制的にも終らせられないとなると, それ以上tgif上での作業を進められなくなってしまいます).文字列中では, その図形の他の属性の値を参照できます. 具体的には, *attr*がその図形に与えてある別の属性の一つであるとすると, *$(attr)*とすることでその値を参照できます.

例えばman(1)を起動したいものとしましょう. そのためにまず四角形を描いたとします. 次に「title=tgif」という文字列および「launch=xterm -rw -e man $(title)」という文字列を入力し, ここでキーボード上でˆaと押して今入力した2つの文字列と最初に描いた四角形全部を選択ます. 次に#aと押すと, 2つの文字列を属性としてこの四角形に与えられます. こうしておいてから, <シフト>キーを押しながらこの四角形 (または文字列)のうえで中央のマウスボタンをクリックするとman(1)を起動できます.四角形の代わりにX11ビックスマップを使い, 「launch」属性を表示しないように設定し, しかもキーボードで#mと押すことで「title」の属性名部分を非表示にして中央揃えにしたりすれば, もっと見栄えをよくできます.

内部命令

内部命令を直接起動できます. ある図形の上で<シフト>キーを押しながら中央のマウスボタンをクリックした時に, もしその図形に (標準の設定では) *exec*という名前の属性があると, その値を内部命令とみなして処理します (複数命令はセミコロンで区切ります). 色の指定については, 瞬間移動と同様です.命令は通例次のような形式で指定します:

<命令> (<引数1>, <引数2>, ..., <引数N>)

命令に与える実引数は, 文字列でも数値でもかまいません. 文字列の場合は二重引用符でくくっておきます. 数値の場合は, 数値そのものでもかまいませんし, 例えば*x*という属性がある場合に「$(*x*)」というふうに属性名を指定してもかまいません (属性の値に置き換えます). 文字列中にも属性名が含まれていても構いませんが, 置き換えは1回しか行ないません (置き換えた文字列中にまた属性名があっても再度置き換えることはありません).

現在のところ, 数はあまりありませんが, 以下の内部命令をこのようにして指定することのできます:

launch(<属性名>*)*
: <属性名>の値をsh(1)に渡して実行します. 詳しくは【他のアプリケーションの起動】の項を参照して下さい.

exec(<属性名>*)*
: <属性名>の値を内部命令として処理します. これはサブルーチン呼び出しのようなものです.

mktemp(<文字列>,<属性名>*)*
: 他のファイルと重複しないようなファイル名を作り, <属性名>の属性の値として設定します. <文字列>はファイル名を作る時の雛型としてつかいます. 例えば「/tmp/TgifXXXXXX」のようなものです. 詳しくはCのライブラリmktempの解説を参照して下さい.

create_file_using_simple_template(<雛型>,<出力>,<文字列>,<属性>*)*
: <雛型>という名前のファイルの中に<文字列>と同じ行があるかどうかを調べ,もしあればその行を<属性>の値で置き換えます. 結果は<出力>という名前のファイルに出力します.

update_eps_child(<*eps*ファイル名>*)*
: この命令は基本図形に対しては働きません. 図形中に取り込んだEPS図形がある場合, これを<epsファイル名>という名前のEPSファイルで置き換えます. 図形中にEPS図形がない場合は, EPS図形を追加します.

flip_deck(<回数>,<毎秒の切替え回数>,<方式>*)*
: この命令は基本図形に対しては働きません. また, 同じ場所にある同じ大きさのX11ビットマップまたはX11ビックスマップだけを含む図形に対してのみ働きます (訳注: この命令によって, 複数のビットマップまたはビックスマップ図形を, 順次切替えながら表示させられます). <回数>には, 切替える総回数として, 数または「infinite」 (無限) を指定します. <毎秒の切替え回数>には1から60の間の値を指定します. <方式>には「linear」 (循環) または「ping_pong」 (往復) を指定します.

図形の操作禁止

キーボードから #< と押すことで図形の操作を禁止できます. また#> で解除できます. 操作を禁止すると, その図形の取手を灰色にします. 操作を禁止した図形を動かしたり, 伸ばしたり, 反転したり, 回転したりすることはできません. しかし, 塗りつぶし模様や線の太さなどは変更できます. また, 操作を禁止した図形でも消去することはできます. 操作を禁止した図形を1つ以上含む複数の図形をグループ化した場合, グループ化した図形全体も操作を禁止しているものとして扱います. 操作を禁止した図形は, 他の図形の位置合わせをする際の基準として使えますが, 操作を禁止した図形を含む複数の図形に対して DistributeObjs() を指示することはできんません. 操作を禁止した図形に対しては, 部分的に頂点を選択することもできません.

操作の取消/再実行

ほとんどの操作をなかったことにしたり, 再実行したりできます. Xのリソース Tgif*HistoryDepthに, いくつ前の操作までを覚えておくかを指定します.もしここに-1を設定すると, 無限に覚えておきますが, New()あるいはOpen()をメニューFileから選んだ場合, メニューEditからFlushUndoBuffer()を選んだ場合, シンボルファイル中でPop()を選んだ場合などは, その次の操作から覚え直すことになります. プライベートカラーマップを使用している場合(新しい色を用いようとした場合に, その色が標準のカラーマップ中に見つからない場合は自動的にプライベートカラーマップを使用します), FlushUndoBuffer()を実行すると, プライベートカラーマップを初期化しようとします (ただしコンパイル時に-DDONT_FREE_COLORMAPを指定している場合はこの限りでありません).

ドメイン

ドメインとは, 部品類をまとめたもののことです. 部品類をまとめたものは, 実態としてはシンボルファイルがおいてあるディレクトリのことで,それゆえドメインとは, 実態としては, シンボルファイルの検索パス名のことです. 検索パス名列中に含まれる複数のディレクトリに同名のシンボルファイルがある場合, 検索パス名列の先頭に近い方のディレクトリにあるシンボルファイルの方を図面上に置くことになります.

ドメインの個数は, XのリソースMaxDomainsによって設定できます. ドメイン名は, XのリソースDomain#に設定します. 検索パス名列は環境変数として指定します. 詳しくは【Xのリソース】および【環境変数】の項を参照して下さい.

一覧表からの選択

ファイル名, 部品名, ドメイン名などを選択する場合, 候補の一覧を表示しますので, この中から適当なものをマウスでクリックして選んで下さい.マウスでなく名前を直接キーボードから入力することもできます. この場合, 1文字入力する度に合致する候補があるのかどうかを調べます. ある文字を入力した時に, その文字と合致する候補が1つもなければ, その文字を無視します. ˆn, ˆj, あるいは下向き矢印キーを押

すと1つ下の候補を指定できます. ^p, ^k, あるいは上向き矢印キーを押すと1つ上の候補を指定できます. ^f, ^d,あるいは<コントロール>キーと同時に下向き矢印キーを押すと10行下の候補を指定できます. ^b, ^uまたは<コントロール>キーと当時に上向き矢印を押すと10行上の候補を指定できます. '$'キーは最後, '^'キーは最初の候補の指定になります. ^wまたは^yを押せば指定を初めからやり直せます. もし指定した候補がディレクトリの場合は, <改行>を押すとそのディレクトリに移って候補指定をつづけることになりますし, ディレクトリではない場合は, その候補を最終的な選択結果とします.また, いずれの場合も<ESC>を押せば選択を中止できます.

読み込む, あるいは取り込もうとするファイルを選択しようとしている場合に'/'を押すと,ルートディレクトリからの絶対指定でパス名を指定できます.通常1文字入力するたびに該当候補があるかどうかを調べますが, '/'を押した場合は, <TAB>または<CR>を押すまでの間, 一時的に該当候補があるか否かを調べなくなります.

候補の指定結果はポップアップウィンドウの最上部に表示してあります. ここでバックスペースを押すときには注意して下さい. '/'の前まで消してしまうと, その瞬間に一つ上のディレクトリに移ってしまいます.

EPSファイルの取り込み

キーボードで #(と押すことで, EPSファイルを取り込めます.もし対応する表示用ビットマップ (これは*pstoepsi*ユーティリティを使って作れます) があれば, それを表示します (Layoutメニューの中のHideBit/Pixmapを選択すれば表示を消せます).取り込んだEPS図形を図面ファイルあるいはシンボルファイルに書き込んだとしても, 表示用ビットマップあるいは EPS ファイル中の PostScript のコードが書き込まれるわけではありません.従って, EPS図形を含む図面ファイルあるいはシンボルファイルを印刷するためには,そのファイルと同じところに元の EPS ファイルが存在していなければならないことに注意して下さい.

部品の作り方

この項では部品の作り方について解説します. 部品を用いることで, 図面を階層的に構成できるようになります.

1) 部品の表現部を製図します. これらをグループ化し, グループ化した図形を選択します.

2) メニューから「その他 (Special)」を選び, 表示された一覧から「MakeSymbolic」を選ぶことで, 選択した図形をシンボルにします. シンボルの周りには破線が現れるはずです.

3) 属性になるべき文字列を書き, この文字列と先に作っておいたシンボルとを選択します. ここで #a (*Attach*) と押して, 文字列を属性としてシンボルに付与します.

4) 部品の定義部を製図します (しなくてもかまいません). 定義部の製図の方法については「flip-flop.sym」ファイルを参考にして下さい. このファイルを見るためには, 初めに ^i (*Instantiate*) と押し, 「flip-flop」を選び, 位置を決めて, 部品「flip-flop」を図面上に置きます. こうしてから ^v (*Push*) と押すとflip-flopの定義を見ることができます.

5) ファイルに書きだします. 現在のライブラリパス名にカレント・ディレクトリ (あるいは「.」) が入っている場合, ^i と押すことで, ここで述べたようにして作った部品を図面上に置けます.

X11ビックスマップの形式

Tgifでは以下の条件を満たすX11ビックスマップのみを取り込めます. すなわち形式1 (XPM1) あるいは形式3 (XPM3) でなければなりません. またXPM3の全てを受け付けられるわけではなく, XPM3のうち, 色指定の*key*フィールドの値が「c」であるものに限ります. XPM1形式のファイルを作ることのできる道具には, ビットマップやピクスマップ間の変換を行うフリーウエアの*pbmplus* (*xv*と一緒に使えばビックスマップ図形の色を変えることもできます) や*xgrabsc* (これもフリーウェアです) などがあり, また*xloadimage*を使えばXPM1形式のファイルを表示できます (これらもいずれはXPM3形式も扱えるようになるでしょう). XPM3形式のファイルを作ることのできる道具としては, *xsnap*や *sxpm*などがあります. 色指定の文字列に指定してある各色に実際に色を割り当てますが, もし割当てられなかった場合は, 『現在の色』で代用します.最初の色指定の文字が逆引用符「`」あるいは空白文字で, かつXのリソースTgif*GuessXPmBgColorに「true」を指定してある場合は,該当する色をtgifウインドウの*background (*背景)の色で置き換えます(ビクスマップが正常,反転表示の両方で正しく見えるようにこのような仕様になっています).簡単な (XPM1形式の) ビクスマップの例を以下に示します.

```
#define arrow_format 1
#define arrow_width 5
#define arrow_height 3
#define arrow_ncolors 3
#define arrow_chars_per_pixel 1
static char *arrow_colors[] = {
  "`", "Black",
  "a", "red",
  "b", "yellow"
};
static char *arrow_pixels[] = {
"`a``",
"aabbb",
"`a``"
};
```


LaTeXへの図形の組み込み方法

ここでは LaTeX ファイルに使うための方法を示します. 最初に *\psfig* (または *\epsf*) 命令を使う方法,それから *psfile* 命令を使う方法です (著者は *psfile*命令を利用するのはお勧めしません). Tgifと一緒に配布しているファイル「example.tex」に,それらの使用例があります.

*\psfig*や*\epsf*命令を使ってLaTeX文書に(*EPS*形式の) 図面を組み込むには, 最初に (レーザ印刷装置の印を左マウスボタンでクリックして) LaTeXモードに切替えてから^pを押してEPSファイルを作ります.例えばそのファイル名が「an-sr-flip-flop.obj」である場合,出力するファイル名は「an-sr-flip-flop.eps」になります. このファイルは, 以下のようにすることでLaTeX文書に組み込めます.

```
\input{psfig}
\begin{figure*}[htb]
\centerline{\psfig{figure=an-sr-flip-flop.eps}}
\caption{An SR flip-flop. \label{fig:an-sr-flip-flop}}
\end{figure*}
```

*\epsf*命令を使う以下のような方法もあります.

```
\input{epsf}
\begin{figure*}[htb]
\centerline{\epsffile{an-sr-flip-flop.eps}}
\caption{An SR flip-flop. \label{fig:an-sr-flip-flop}}
\end{figure*}
```

上の例の中の\centerline命令は, 絵を中央に配置するためのものです. なお複数の (tgifで作図した) 図面をLaTeX文書に組み込む場合でも, psfigマクロは\begin{document}の後に一回だけ(\input{psfig}か\input{epsf}のように) 書けば十分です.

もしEPSファイルを使えない場合には, 以下に述べるようにして*psfile*命令を使う方法があります. 但し, この方法では図面の枠 (bounding box)がどこにあるのかをLaTeXに教えられないので, うまくやるにはちょっとしたコツが必要です.ここにうまく行きそうな方法について述べます.最初にその図形を頁の中心に置きます (即ち, 縦書きのの紙幅は210mmなので図の中央をcm定規で10.5のところに合わせるわけです). そして, 図面の一番上がが頁の上端から約7mm下になるようにした後, 選択欄でLaTeX形式を選んで出力します.これで*psfig*の時と同様に.*eps*の拡張子の付いたファイルが出来ます.このファイルはLaTeXの文章に以下のようにして組み込めます.

```
\begin{figure*}[htb]
\special{psfile="an-sr-flip-flop.eps" hoffset=-40}
\rule{0mm}{28mm}
\caption{An SR flip-flop. \label{fig:an-sr-flip-flop}}
\end{figure*}
```

\rule{0mm}{28mm}は, 28mmの高さの見えない箱を表しており, それが図面 an-sr-flip-flopの高さになります.

Xのリソース

*Tgif*Geometry:* 幅x高さ+X+Y
ウィンドウの位置を位置を指定します.

*Tgif*IconGeometry:* +X+Y
アイコンにした時の位置を指定します.

*Tgif*Foreground:* 色
前景の色を指定します.

*Tgif*Background:* 色
背景の色を指定します.

*Tgif*BorderColor:* 色
枠の色を指定します.何も指定しない場合は前景の色と同じになります.

*Tgif*ReverseVideo: [on,off]*
ビデオ反転を「on」にすると, モノ黒端末では背景色が黒になり, カラー端末では背景色がTgif*Foregroundが指定した色になります.

*Tgif*InitialFont: [Times,Courier,Helvetica,NewCentury,Symbol]*
最初に使用するフォントを指定します. 何も指定しないとCourieになります.

*Tgif*InitialFontStyle: [Roman,Bold,Italic,BoldItalic]*
最初に使用するフォントの体を指定します. 何も指定しないとRoman体になります.

*Tgif*InitialFontJust: [Left,Center,Right]*
最初に使用するフォントの揃え方を指定します. 何も指定しないと左揃えになります.

*Tgif*InitialFontDPI: [75,100]*
最初に使用するフォントの解像度DPI (インチ当たりのドット数) を指定します. 何も指定しないと75DPIになります.

*Tgif*InitialFontSizeIndex: [0, 1, 2, 3, 4, 5]*
最初に使用するフォントの大きさを0から5の間の数字で指定します. 75DPIの活字を使ったの場合, 0, 1, 2, 3, 4, 5は, それぞれ8, 10, 12, 14, 18, 24ポイントに対応します. 100DPIの活字のときは, 同様に11, 14, 17, 20, 25, 34ポイントに対応します. 何も指定しないと, 75DPIの解像度では18ポイントに対応した4を, 100DPIの解像度では17ポイントに対応した2を指定したものとみなします.

*Tgif*MsgFontSizeIndex: [0, 1, 2, 3, 4, 5]*
表示欄やメニューを表示する際に用いるフォントの大きさを, 0から5の間の数字で指定します. 各数字の意味はTgif*InitialFontSizeIndexで説明したとうりです. 何も指定しないと, 75DPIの解像度では18ポイントに対応した4を, 100DPIの解像度では17ポイントに対応した2を指定したものとみなします.

*Tgif*DefaultCursor: [X_cursor, arrow,...]*
選択モードの時のカーソルの形を指定します. <X11/cursorfont.h>中の名前から頭のXC_を取り去った部分を指定します. 何も指定していない場合は arrowです.

*Tgif*DrawCursor: [X_cursor, arrow,...]*
図形を描く時のカーソルの形を指定します. <X11/cursorfont.h>中の名前から頭のXC_を取り去った部分を指定します. 何も指定していない場合は Tgif*DefaultCursor と同じものを指定したものとして扱います.

*Tgif*DragCursor: [X_cursor, arrow,...]*
マウスをドラッグしている時のカーソルの形を指定します. <X11/cursorfont.h>中の名前から頭のXC_を取り去った部分を指定します.何も指定していない場合はhand2です.

*Tgif*VertexCursor: [X_cursor,arrow,...]*
頂点選択モード時のカーソルの形を指定します. <X11/cursorfont.h>中の名前から頭のXC_を取り去った部分を指定します. 何も指定していない場合はplusです.

*Tgif*RubberBandColor:* 色
曲線を変形させる際 (訳注: Xのリソース*Tgif*SplineRubberband*参照)の曲線の色を指定します (ここに指定した色と, 背景色との各ビットごとの排他的論理和をとった色が実際の表示色になります). 何も指定しない場合は前景色に同じです.

*Tgif*PrintCommand:* 命令
PSファイルを印刷する際にどのような命令を用いて印刷するのかを (例えば *lpr -h -Pprintername*のようにして) 指定します. 何も指定しない場合は*lpr*です.

*Tgif*WhereToPrint: [Printer,EPS,PS,Bitmap]*
印刷形式の初期値を設定します. 何も指定しない場合はEPSです.

*Tgif*PrintDirectory:* パス名
印刷形式が印刷装置でない場合, 出力ファイルを何処に作るのかを指定します.ここに何も指定しないと, 現在処理中のファイルがあるディレクトリに作ることになります.

*Tgif*NoTgifIcon: [true,false]*
「true」を指定すると, tgif独自のアイコンを用いません (訳注: どのようなアイコンにするのかをウィンドウマネージャーに任せます). 何も指定しない場合は「false」です.

*Tgif*UseWMIconPixmap: [true,false]*
「true」を指定すると, アイコン用のビックスマップを用意し, それをアイコン表示に用いるように設定して, 後はウィンドウマネージャーに任せます.これを指定すると, 自動的にTgif*NoTgifIconリソースもtrueに設定します.何も指定しない場合は「false」です.

*Tgif*DontShowVersion: [true,false]*
「true」を指定するとtgifウィンドウの一番上にバージョン番号を表示しなくなります (訳注: ウィンドウが1行分だけ広く使えるようになります). 何も指定しない場合は「false」です.

*Tgif*XBmReverseVideo: [true,false]*
「true」を指定すると, Xビットマップファイルを取り込む際に, 白黒を反転してから取り込みます. 何も指定しない場合は「false」です.

*Tgif*AskForXBmSpec: [true,false]*
「true」を指定すると, Xビットマップを取り込む際に拡大縮小率, および (部分的に取り込む) 取り込み位置について指定するよう求めます. この場合, $MAG=WxH+X+Y$のようにして, 拡大率MAG, 取り込み部分の幅 Wおよび高さH, 取り込み部分の位置$+X+Y$あるいは $+X-Y$, $-X+Y$, $-X-Y$を指定します. なお, 拡大縮小率を省略する場合でも, 部分的な取り込みを指定する際には「=」を省略することはできません. 何も指定しない場合は「false」です.

*Tgif*AskForXPmSpec: [true,false]*
「true」を指定すると, Xビックスマップを取り込む際に拡大縮小率, および (部分的に取り込む) 取り込み位置について指定するよう求めます. 指定の方法はAskForXBmSpecと同じです. 何も指定しない場合は「false」です.

*Tgif*StripEPSComments: [true,false]*
「true」を指定すると, EPSファイルを取り込む際に'%'で始まるコメント部分 (最初の行を除きます) を取り除きます. 何も指定しない場合は「true」とみなします.

*Tgif*GuessXPmBgColor: [true,false]*
「true」を指定してXビックスマップを取り込む場合に, 色指定の最初の文字が' ' (空白文字) あるいは'`' (バッククオート文字) の場合, 定義してある最初の色を**背景色**として扱います. すなわち, Xビックスマップを読み込んだ際に, その中の色指定にしたがって背景色が変わる, ということです.何も指定しない場合は「false」とみなします (何も指定しない場合, tgif-2.7 patch 2までは「*true*」とみなしていました. このXリソースは互換性のために残してありますが, 使わないで下さい).

*Tgif*XPmOutputVersion:* 数
Xビックスマップを出力する際の, ファイル形式のバージョンを指定します.数には1ないし3を指定します. 何も指定しない場合は1です.

*Tgif*XPmInXGrabSCFormat: [true, false]*
Tgif*XpmOutputVersionに1を指定している場合に, このリソースに「true」を指定すると, 生成するX11ビックスマップの形式をxgrabscの出力する形式と同じようなものにします. 何も指定しない場合はfalseです.

*Tgif*UseGrayScale: [true,false]*
「true」を指定すると, 印刷時間短縮のために, タイル模様を印刷する際に中間階調を用います. 何も指定しない場合は「false」です.

*Tgif*AutoPanInEditText: [true,false]*
「true」を指定すると, 文字列を編集している際に (図面の左または上にはみ出している部分を除いて) 文字カーソルが常に見えるように, 図面の表示位置を自動的に移動します. 遅いサーバーを使っている時には, この機能を用いないほうがよいでしょう. 何も指定しない場合は「true」です.

*Tgif*PercentPrintReduction:* 拡大縮小率
印刷時の拡大縮小率を指定します. 何も指定しない場合は100です.

*Tgif*ConstrainedMove: [true,false]*
制約付の移動にするかどうかを指定します. 「true」を指定すれば, 図形の移動したり拡大縮小したりする際に, もしその図形の内部に端点が含まれている折れ線あるいは開いたスプライン曲線があると, その端点およびその隣の点も,図形と一緒に移動します. 詳しくは【特記事項】の項を参照して下さい.何も指定しない場合は「false」です.

*Tgif*DoubleQuoteDoubleQuote: [true,false]*
「true」を指定すると, 文字列を出力する際, 含まれる二重引用符を二つの連続する二重引用符として表します. 「false」の場合は, 二重引用符の前に逆スラッシュを付けて表します. 何も指定しない場合は「false」です.

*Tgif*GridSystem: [Metric, English]*
目盛の単位系を設定します. 標準はセンチ・メートル (Metric) 目盛です. インチ目盛の場合はEnglishと指定します.

*Tgif*InitialGrid:* 間隔
格子間隔を指定します. 「間隔」に-1, 0, +1, +2を指定すると, 格子間隔をそれぞれ1mm, 2mm, 5mm, 1cmにします. 何も設定しない場合は0です (インチ目盛の場合は-2, -1, 0, +1, +2がぞれぞれ1/32, 1/16, 1/8, 1/4, 1/2インチに対応します).

*Tgif*DropObsIconAttrWhenUpdate: [true,false]*
「true」を指定すると, UpdateSymbols命令を実行した際に, 記号中の (もはや元の部品の定義からなくなってしまった) 属性を, 無条件に削除します.「false」を指定した場合は, どのようにしたらよいのかを尋ねます. 何も指定しない場合は「false」です.

*Tgif*UseRecentDupDistance: [true,false]*
「true」を指定すると, 図形を複製する際に, 前回の複製の後 (および複製に続いて図形を動かした場合も, その後) の元の図形と複製して作った図形との相対位置と同じような相対位置に, 複製して新たに作った図形を表示します.「false」を指定した場合はある決まった相対位置に表示します. 何も指定しない場合は「true」です.

*Tgif*SplineTolerance:* 数
スプライン曲線を描く場合の「手抜き具合」を指定します. 小さい数を指定すればするほど (手抜きをしない) 滑らかな曲線になります. 何も指定しない場合は3です (最小3から最大13の間で指定して下さい).

*Tgif*SplineRubberband: [true,false]*
「true」を指定すると, (開いたもの, 閉じたもの, いずれの) スプライン曲線についても, 描画, 移動, 拡大縮小する際に「ゴム輪」のように曲線そのものが伸び縮みするように見えます (スプライン曲線が余りに多くの点からなる場合は, (曲線の描き直しが) 遅くてやってられないかもしれません). 何も指定しない場合は「true」です.

*Tgif*Synchronize: [on,off]*
「on」指定すると, (訳注: ライブラリ関数) XSynchronizeを (訳注: 起動時に) 呼び出し (訳注: Xサーバーとのやりとりを同期モードで行なうようにし)ます (訳注: この結果Xサーバーとのやりとりが目に見えて遅くなります. 主としてデバッグ目的に用いるためのものです). 何も指定しない場合はoffです.

*Tgif*DoubleClickUnIconify: [true,false]*
「true」を指定すると, アイコンをダブルクリックすることで, tgifのウィンドウを開けるようになります (訳注: ウィンドウマネージャーの設定によらないので, ウィンドウマネージャーによる統一的な操作方法を破ることになってしまうことに注意して下さい). この場合, アイコンを (1回だけ) クリックしたり, ドラッグしたりしても何も起こりません. 何も設定しない場合はfalseとして扱います.

*Tgif*MainMenuPinDistance:* 数
ここに設定した距離 (画素数) よりも長くポップアップメニューをドラッグすると, メニューをピン止めします. 何も指定しない場合は80とします (もしメニューをピン止めしたりしたくない場合は, ここに画面の幅よりも大きな値を設定して下さい). 左マウスボタンを使ってドラッグすることで, ピン止めしたメニューを動せます. 右マウスボタンでクリックすると, ピン止めしたメニューを消せます.

*Tgif*DoubleClickInterval:* 間隔
ここに設定した間隔 (ミリ秒) よりも短い時間内にマウスボタンを2回クリックすると, それをダブルクリックとして扱います. 何も指定しない場合は300ミリ秒です.

*Tgif*HandleSize:* 数
強調している図形の取手の大きさを2から6の間で指定します. 何も指定しない場合は2です.

*Tgif*HistoryDepth:* 数
いくつ前の操作までなかったことにし, やり直せるのかどうかを指定します.負の値を指定した場合は無制限に戻れます. 何も指定しない場合は-1です.

*Tgif*SaveTmpOnReturn: [true,false]*
「true」を指定すると, ドライバに制御を移す前に, 自動的に「tmpmodel」ファイルに図面をセーブします. 「false」を指定した場合は, 自動的にはセーブしません. 何も指定しない場合はtrueです.

*Tgif*ImportFromLibrary: [true,false]*
「true」を指定すると, 図形, ビットマップ, ビックスマップ, およびEPSファイルを取り込む際に, ファイルを現在のドメインに対応するディレクトリ中で探します. 「false」を指定した場合は起動時のカレントディレクトリ中で探します. 何も指定しない場合はfalseです.

*Tgif*WarpToWinCenter: [true,false]*
「true」を指定すると, ポップアップウィンドウを表示する際に, マウスカーソルがそこの位置まで飛びます. 「false」の場合はカーソルが飛ぶことはありません. 何も指定しない場合はtrueです (訳注: いわゆる「業界標準」の「Motifスタイルガイド」によれば, マウスカーソルをアプリケーションが勝手に移動させるのはユーザーを混乱させるので絶対にやってはいけない, ということになっています).

*Tgif*SaveCommentsInSaveNew: [true,false]*
「true」を指定すると, SaveNew (新しいファイルにセーブ) の際, 現在のファイル中の「%%」で始まるコメントを新しいファイル中にコピーします (訳注: falseを設定すると, あるファイルから読み込んだ図形を編集している場合でも, これを別のファイルにセーブするとコメントがないものとして処理するために, 新しいファイルにはコメントがない, ということになります). 何も指定しない場合はtrueです.

*Tgif*CanvasWindowOnly: [true,false]*
「true」を設定すると, 図面だけを表示します (これはtgifを用いた各種「デモンストレーション」の際に便利でしょう). 何も指定しない場合はfalseです.

*Tgif*UsePsAdobeString: [true,false,数1/数2]*
PSあるいはEPSファイルを出力する際に, 第1行目になんと出力するのかを指定します. 「true」を指定した場合は「%!PS-Adobe-2.0 EPSF-1.2」となります.「false」を指定した場合は単に「%!」となるだけです. 「数1/数2」を指定した場合は「%!PS-Adobe-数1 EPSF-数2」となります. 何も指定しない場合はfalseです.「PS-Adobe」という文字列があるとTranscriptが誤動作しますが, groffには必要です.

*Tgif*HalfToneBitmap: [true,false]*
「true」を指定すると, X11ビットマップファイルに印刷する際に Floyd-Steinbergの中間階調法を用います. 図面中に色を用いている場合に便利です. 何も指定しない場合はfalseです.

*Tgif*ThresholdBitmap: [true,false]*
「true」を指定すると, X11ビットマップファイルに印刷する際に, 単純な閾値アルゴリズムを用います. Xのリソース Tgif*HalfToneBitmapにtrueを指定した場合は, ここに何を設定しても関係ありません. 何も指定しない場合は falseです.

*Tgif*BitmapThreshold:* 閾値
Floyd-Steinberg中間階調アルゴリズム, あるいは単純な閾値アルゴリズムで用いる閾値として0ないし1の数値を指定します. この指定は, Xリソース中で Tgif*HalfToneBitmapあるいは Tgif*ThresholdBitmapのいずれかにtrueを指定した場合に有効になります. 何も指定しない場合は, Tgif*HalfToneBitmapに trueを指定してある場合には0.5, Tgif*ThresholdBitmapにtrueを指定してある場合には1.0 (1.0は完全な白以外の色を全て黒とすることに対応) です.

*Tgif*GroupedTextEditable: [true,false]*
「false」を指定すると, 文字列の編集は (最上位にある) 文字列図形に対してしかできなくなります. 「true」を指定すると (グループ化してある図形中の文字列など) どのような文字列でも編集できるようになります. 何も指定しない場合はfalseです.

*Tgif*DefaultEPSScaling:* 比率
PSあるいはEPSファイルを取り込む際に, 全体の大きさをここに指定した比率倍にします. 【特記事項】の項で述べるように, 128点を25.4mmとして扱っていますが, PostScriptでは72点を25.4mmとして扱っています. このため, PostScript図形を正しい大きさで扱うためには, 1.778 (=128/72) を比率として指定しなければなりません. 何も指定しない場合は1です.

*Tgif*IntrCheckInterval:* 数
ここに指定した数の図形を描く度に, 割り込みがあるかどうかを調べます. したがって, 0あるいは負の値を指定すると, 割り込みはできなくなります. 何も指定しない場合は10です.

*Tgif*TiledPageScaling:* 比率
一般的にいってPostScript印刷装置を使って紙の全面に印刷することはできません (上下左右に余白が必要です). 複数ページをタイル状にして用いている場合, 印刷時に, 各ページをここに指定した比率倍にして印刷します. もしこの比率を1にしてしまうと, ページ境界付近を帯状に切り取ってしまったようになってしまいます. 何も指定しない場合は0.9です.

*Tgif*MaxColors:* 色数
最大の色数を指定します. Xのリソース定義中にColor0ないしColorMaxの指定がなければなりません (Maxは色数-1です).

*Tgif*Color#:* 色
各色番号にはどの色が対応しているのかを指定します.

*Tgif*DefaultColorIndex:* 色番号
ある色を見つける見つけられなかった場合, 代わってここに指定した番号の色を用います. 何も指定しない場合は0です.

*Tgif*ShortCuts:* 項目*1* 項目*2* ...
項目には, ある1文字 (大文字小文字の区別を付けられます) と (通常は非図形文字を押すことで起動している) 諸処理との対応を書きます. 詳しくは【短縮形】の項を参照して下さい.

*Tgif*MaxLineWidths:* 種類
線の太さのを何種類用意するのかを指定します. Xリソース中にLineWidth0ないしLineWidthMax, ArrowWidth0ないしArrowWidthMax, ArrowHeight0ないし ArrowHeightMaxの指定が必要ですが, もし指定がない場合はある標準値を用います (Maxは種類-1です). 何も指定しない場合は7種類です.

*Tgif*DefaultLineWidth:* 番号
Tgif起動時の線の標準の太さ (を表す**番号**) を設定します. 何も指定しない場合は0番です.

*Tgif*LineWidth#:* 太さ
ある番号に対応する線の太さを指定します. 標準設定では, (番号0ないし6が)それぞれ1, 2, 3, 4, 5, 6, 7に対応します.

*Tgif*ArrowWidth#:* 太さ
円弧, および開いたスプライン曲線に付ける矢印の「矢」の (矢印が上を向いているとした時の) 太さを指定します. 標準設定では, それぞれ8, 10, 12, 14, 18, 20, 22です.

*Tgif*ArrowHeight#:* 高さ
円弧, および開いたスプライン曲線に付ける矢印の「矢」の (矢印が上を向いているとした時の) 高さを指定します. 標準設定では, それぞれ3, 4, 5, 6, 7, 8, 9です.

*Tgif*DefaultLineType:* 線種
Tgif起動時の線の種類としてstrait (折れ線) あるいはspline (スプライン曲線) を指定します. 何も指定しない場合は「折れ線」です.

*Tgif*DefaultLineStyle:* 矢印
Tgif起動時の矢印の種類としてplain (矢印なし), right (線の終了点に矢印), left (線の開始点に矢印), both (線の開始点および終了点の両方に矢印) を指定します. 何も指定しない場合はplainです.

*Tgif*DefaultLineDash:* 破線
Tgif起動時の破線の種類として0から8の番号を指定します. 0は実線を, 1は目の最も粗い破線, 8は目の最も細かい破線を表します. 何も指定しない場合は0です.

*Tgif*MaxDomains:* 領域数
最大の領域数を指定します. Xのリソース定義中にDomain0ないしDomainMaxの指定がなければなりません (Maxは領域数-1です).

*Tgif*Domain#:* 領域名
ある番号の領域の名前が何であるのかを指定します. その名前の領域が実際になんというパス名に対応しているのかを指定する方法については【環境変数】の項を参照して下さい.

*Tgif*DefaultDomain:* 数
tgif起動時の領域を設定します. 何も指定しない場合は0です.

*Tgif*ScaleRotateJapanese: [true,false]*
縦書きの日本語文字, および, 標準の大きさの日本語活字以外の日本語フォントを (図面上に) 表示する際, 文字を回転表示させるか, あるいは自動的に文字を拡大縮小表示させるのかを指定します. trueを指定すれば文字として表示しますし, falseを指定すればこれらを全て黒い四角形として表示します.何も指定しない場合はtrueです.

*Tgif*BaseRyuminFont:* 明朝用体用標準フォント
標準の明朝体用のフォントを指定します. 何も指定しない場合は14ドットのフォントを用います.

*Tgif*BaseGothicFont:* ゴシック体用標準フォント
標準のゴシック体用のフォントを指定します. 何も指定しない場合は14ドットのフォントを用います.

*Tgif*KanjiRyumin-#-75: 75dpi*用明朝体フォント
75dpi用の明朝体フォントについて, 標準フォントを自動的に拡大縮小したものではなく, 実際に存在するフォントを指定することもできます. #の部分はフォントのドット数で, 8, 10, 12, 14, 18, 24のいずれかです. 指定のフォントが見つからない場合は標準フォントを自動的に拡大縮小

したもので代用します.

*Tgif*KanjiGothic-#-75: 75dpi*用ゴシック体フォント
75dpi用のゴシック体フォントについて, 標準フォントを自動的に拡大縮小したものではなく, 実際に存在するフォントを指定することもできます. #の部分はフォントのドット数で, 8, 10, 12, 14, 18, 24のいずれかです. 指定のフォントが見つからない場合は標準フォントを自動的に拡大縮小したもので代用します.

*Tgif*KanjiRyumin-#-100: 100dpi*用明朝体フォント
100dpi用の明朝体フォントについて, 標準フォントを自動的に拡大縮小したものではなく, 実際に存在するフォントを指定することもできます. #の部分はフォントのドット数であり, 11, 14, 17, 20, 25, 34のいずれかです. 指定のフォントが見つからない場合は標準フォントを自動的に拡大縮小したもので代用します.

*Tgif*KanjiGothic-#-100: 100dpi*用ゴシック体フォント
100dpi用のゴシック体フォントについて, 標準フォントを自動的に拡大縮小したものではなく, 実際に存在するフォントを指定することもできます. #の部分はフォントのドット数であり, 11, 14, 17, 20, 25, 34のいずれかです.指定のフォントが見つからない場合は標準フォントを自動的に拡大縮小したもので代用します.

*Tgif*InitialKanjiFont:* 日本語フォント
Tgif起動時の日本語フォントとして, RyuminあるいはGothicのいずれかを指定できます. 何も指定しない場合は日本語フォントを選択してない (つまり英文字のみを用いる) 状態になります.

*Tgif*IMProtocol: [kinput,xim]*
日本語入力方法を指定します. kinputを用いる方法と, XIMを用いる方法のいずれかを利用できます. 何も指定しない場合は, 日本語の入力はできません.

*Tgif*ConvSelection: kinput*変換セレクション名
kinputの変換セレクション名を指定できます. 何も指定しない場合は「_JAPANESE_CONVERSION」を用いて*kinput2*と接続しますが, 例えば「JAPANESE_CONVERSION」を指定すれば, *kinput*や*sjx*などと接続することもできます.

*Tgif*Modifier: Modifier*
日本語入力方法としてXIMを用いる際のmodifierを指定します. 何も指定しない場合で, かつ国際化ライブラリとしてXsiを使用している場合は,「@im=_XWNMO」になります. 国際化ライブラリとしてXimpを使用している場合は, 必ずこのリソースに適切なmodifierを指定して下さい.

*Tgif*Lang:* 言語
日本語入力方法としてXIMを用いる際の言語を指定します. 何も指定しない場合で, かつ国際化ライブラリとしてXsiを使用している場合は「ja_JP.ujis」になります. 何も指定しない場合で, かつ国際化ライブラリとしてXimpを使用している場合は「ja_JP.EUC」になります.

環境変数

TGIFPATH
この環境変数には,【ファイル】の項で述べたファイルの置いてあるディレクトリ名を設定して下さい.

TGIFICON
この環境変数には, tgifをアイコン化したときの (アイコンとして表示する)図面ファイルの名前を指定できます. 標準設定では「tgificon」です. 「/」で始まっている場合は絶対パス名として解釈します. そうでない場合は $TGIFPATH/$TGIFICONというファイルを探します.

TGIF_[ドメイン名]
Xリソースとして定義した各ドメインごとに, そのドメインのシンボルファイルを探すディレクトリ名を**TGIF_ドメイン名**というXリソースに設定できます. 設定の仕方はシェルの*PATH*環境変数の設定方法 (訳注: パス名を「:」区切って並べる方法) と同じです(但し, 領域名が*Examples*であるが*TGIF_Examples*という名前の環境変数が設定されていない場合, (訳注: カレントディレクトリを探すのではなく) コンパイル時にTGIF_PATHとして設定した値を用います).例えば, ドメイン*DEFAULT*の部品を/tmp/tgif/symbols中から探すように指定するのなら

ば, 以下のshコマンドを実行すれば言いわけです.

```
TGIF_DEFAULT=/tmp/tgif/symbols:.
export TGIF_DEFAULT
```

ファイル

$TGIFPATH/tgificon.objにtgifの標準的なアイコンを定義してあります.

$TGIFPATH/keys.objに, 非図形文字とそれを押すことで起動できる処理の対応をまとめてあります.

PROLOGあるいはCによるドライバの例

tgifと一緒に, 3つのPrologのプログラムを, 簡単なドライバの例として配布しています. (図面ファイルおよびシンボルファイル両方の形式の) tgifのファイルを解釈するための述語を*tgif.pl*に書いてあります.外部関数呼びだしの機能を利用して, QuintusとかSISCtusとかいったProlog処理系とやりとりするための述語を*frontend.pl*に書いてあります. frontend.plを使うためには, *frontend11.o*を用意しておく必要があります (makefileの中の*frontend11.o:*という部分をコメントにしてあるので, それを外しておきます). Solve() (あるいは #s) コマンドでtgifから一時的に抜け出した時に, 現在作図している図面中の全ての図形のIDを書いたファイルを出力するプログラムを,最後の*testdrive.pl*に書いてあります. tgifのファイルの構造を解析するのもいいでしょう (古いtgifによって作ったファイルとの互換性を保つために, 現在のファイルの構造は複雑なものになっており, そのためファイルの構造についての解説は書いていません).

非常に簡単な, Cで書いたドライバ*testdrive.c*も, やはりtgifと一緒に配布しています. これはProlog版のドライバと同じ動作をするものです. このファイルには, メモリ中の図形と属性をどのように動かしたらよいのかの参考になるコードも書いてあります (これはtgif.cの中にはありません).

日本語化機能

文字列に日本語文字を含められます (内部的には日本語文字はUJIS表現になっています). 文字列に日本語文字を含める場合は, 必ず日本語フォントを指定して下さい. 日本語フォントを指定していない場合は, 入力した日本語文字を日本語文字として認識しません.

図面ファイルに図形を収める際のフォント符号は, 日本語フォントの符号を 1000倍したものに英語フォントの符号を加えたものとして定義します. また日本語フォントの符号が0の場合は日本語を用いないものとして扱います. 従って図面ファイル中のフォント符号が1000よりも小さいということは, 日本語フォントを用いないということを意味しています. また図面ファイル中の文字列は,順序数が126以下の英文字はそのまま, 順序数が160以上の英文字は頭に「\」を付した順序数の8進表記で, 日本語文字はUJIS表記でそのまま表します.

図面上の縦書きの日本語文字は, 黒い四角形あるいは回転した文字として表示します. いずれの表示方式にするのかは, TextStyleメニュー中の「Scale/Rotate Japanese」で指定できます.

日本語フォントとしては「Ryumin (明朝体)」, 「Ryumin-V (明朝体縦書き)」,「Gothic (ゴシック体)」, および「Gothic-V (ゴシック体縦書き)」を指定できます. 但し日本語文字を太字にしたりイタリック体にしたりすることはできません.

日本語入力方法としては, kinputを用いる方法と, XIMを用いる方法を選択でき, 起動時オプションあるいはXのリソースIMProtocolに指定します.

日本語の入力としてkinputを用いる場合, キーボード上で<コントロール>と空白キーを同時に押すことで日本語を入力できるようになります.

関連項目

jlatex(1L), **latex**(1L), **lpr**(1), **env**(1), **X**(1), **dvi2ps**(1), **dvips**(1), **sh**(1), **ksh**(1), **bash**(1), **csh**(1), **pbmplus**(1), **bitmap**(1), **pixmap**(1), **XPM**(1), **xgrabsc**(1), **xloadimage**(1), **xsnap**(1), **sxpm**(1), **xv**(1), **pstoepsi**(1)

特記事項

「ドライバを起動」するべく (不意に) 指示してしまった場合, 現在製図中の図面を一旦「tmpmodel.obj」ファイルに書き込み, ドライバを起動してから直ちに制御を元に戻します. もし製図中の図面がシンボルファイルの場合は,製図中の図面を一旦「tmpmodel.sym」に書き込みますが, ドライバには制御を移しません.

カットバッファからの貼り込み操作は, tgifからの中身に対しても, tgif以外の (例えば*xterm*といった) アプリケーションからの中身に対しても動作します. もしカットバッファの中身がtgif以外からのものの場

合,単純にASCII文字列として扱い, 現在の文字に関する設定にしたがって文字列として貼り込むことになります. もしカットバッファの中身がtgifによって作られたものの場合は, 現在の設定は全て無視します (中身をそのまま貼り込みます).

なお, カットバッファ中に日本語文字が含まれていても取り込めます (「^[$@」あるいは「^[$B」, および「^[(J」あるいは「^[(B」との間にある文字列が日本語文字列として, これをUJISに変換してから取り込みます. 逆にカットバッファに文字列を書き込む場合は, JISに変換してから書き込みます. この際KIN/KOUTとしては「^[$B」および「^[(B」を用います).

選択部分 (X selection, たとえば*xterm*で右ボタンを押しながらマウスを使って反転表示した部分) から文字列を取り込むこともできます. これにはEditメニュー中のSelectionPasteを使うか, あるいは^#Yを押して下さい.なお, ^#Yの「Y」は大文字でなければなりません (小文字のyの場合, つまり ^#yはカットバッファからの貼り込みになります).

文字の大きさはXの**画面上の**大きさです. 印刷すると, 指定の大きさよりも小さいものになっているはずです. たとえば, 24ポイントの文字を印刷すると, 13.5ポイントになっているはずです. これは, tgifでは128画素を 25.4mmとして扱っているのに対して, PostScriptでは72画素を25.4mmとして扱っているためです.

格子線間隔は絶対的なものではありません. 格子線間隔は図面表示上の画素数で決まっているので, 図面を拡大表示すると, それに応じて格子線間隔も代わってしまいます. 例えば図面を最大に拡大表示した状態で, 格子線間隔が画素数にして16になっていたとしましょう. 図形を配置したり, 動かしたり,伸縮したりする際 (格子線に合わせて操作する場合は), 操作の最小単位が図面上の16画素になります. ここで図面の拡大率を1段階下げても, やはり操作の最小単位は16画素ですが, これは最大に拡大した場合の32画素単位の操作に対応しているわけです.

文字列の行間として負の値を設定してある場合, 文字列の一部を反転表示した際に (XOR処理のために) 表示がいささかおかしくなることがあります.文字列の行間として1000以上, あるいは-100以下の値を設定した場合, (訳注:数字が隣にはみ出してしまい) 選択欄の表示がおかしくなります. 表示を元に戻すには, 一旦tgifウィンドウを閉じてから再度開き直して下さい.

【TGIF画面上の欄の種類と操作】の項で述べたように, 制約付き移動を指定してある場合に図形を移動すると, その内部に両端が含まれている折れ線があれば, 例えそれを選択していないとしても, その折れ線も図形と一緒に移動します. しかし, 以下のような場合に奇妙に振舞っているように見える事があります. 例えばある図形を指しているような折れ線があったとしましょう. その図形を動かすと, その図形を指したままになるよう, その折れ線も図形の移動につれて「変形」します. ところが, この結果たまたまその折れ線のもう一方の端もその図形の内側に入ってしまった場合に, さらにその図形を動かすと, (図形を指し示していた点だけが図形につれて動かすのではなく) その折れ線全体が動いてしまうことになります. 折れ線の両端の線分の動きも奇妙なものに見えるでしょう. こういった時には, 制約無し移動を指定したほうがいいと思います.

制約付き移動に関してもう一つ分かりにくい振舞があります. ある図形を複製した直後は制約付き移動を一時的に解除します. 複製して作った図形を好きな場所に配置できる方が便利だろうと思われるためです. 複製して作った図形の移動以外の, 何らかの操作をおこなうと, 複製を「落ち着かせる」効果があるものとみなします. 一旦「落ち着く」と, 再び制限つき移動が有効になります.

図形を操作禁止にしても, 削除することはできます.

注意事項

solidでないペンを指定しcourier体の文字をApple LaserWriterで印刷しようとすると, (文字を単独で出力する場合にはうまくいくようなのですが) 場合によってうまくいかないことがあるようです. PostScriptマニュアルに書かれているように, Courier体は「stroked font」であり, character pathを構成することが「難しい」のです. しかし, ghostscriptやdxpsviewでは Courier体もうまく印刷できます. この問題をどのように解決したらよいのかわかりません. Solidでないペンを必要とし, 多色の図面を印刷しようという場合には, Courier体を使わないことをお勧めします.

円弧に付けた矢印の「矢」はあまり「鋭く」ありません (「矢」の位置が開いたスプライン曲線とは異なっているからです) (訳注: 円弧の場合, 「矢」位置の関係で, 矢の先端がつぶれて見えます).

大きな拡大率を指定している場合に扇型を伸縮しようとすると, 内部計算における切捨て誤差のために, 変則的に変形する事があります.

異常終了してしまった場合, 作図途中の図面を「EmergencySave.obj」(部品を作成中の場合は「EmergencySave.sym」) という名前のファイルに保存しようとします. 大抵の場合「EmergencySave.obj」を読み込むことで, 作図途中の図面を取り戻す事ができます. もっとも (tgifの欠陥が原因で) 図形が壊れてしまったために異常終了してしまった, という場合は, おそらくファイル「EmergencySave.obj」も壊れているでしょう.

内部から他のアプリケーションを起動した際に, 起動したときに与えた文字列の最後の文字が「&」でなく, しかも起動したアプリケーションが終了しないと, tgif上で何の作業もできなくなってしまいます. こういう場合は, kill(1)を用いてHUPシグナルをtgifのプロセスに送信してください. 作図途中の図面を「EmergencySave.obj」という名前のファイルに保存します.

日本語入力方法としてXIMを利用している場合に, 余りに長い (日本語文字で 128字以上の) 文字を一度に確定しようとすると, 文字列の一部がかけてしまうことがあります.

英語版tgif-2.16との相違点

英語版のtgif-2.16と日本語版が準拠しているtgif-2.13との相違点は以下の通りです.

新しい瞬間切替えのための属性*href=*を追加しました
: 属性*href=*にURLを指定することで, httpあるいはftpプロトコルによってアクセスできる「.obj」ファイルに, 瞬間切替えができるようになりました.

内部コマンドの引数に算術式を使えるようになりました
: 内部コマンドの一部には, 引数として算術式を使えるようになりました.

*Navigate*メニューを追加しました
: Navigateメニューは複数ファイルの間を行き来したり, ホットリストをアクセスするための新しいメニューです. 新しいXのリソース*HotListFileName*も追加になっています.

*hyperspace*モードを追加しました
: 他のファイルへの瞬間切替えを定義してある図形や, 他のアプリケーションの起動を定義してある図形の上にカーソルを動かすと, カーソルの形が矢印から手の形に変わるhyperspaceモードが追加になっています.

*-text*オプションを追加しました
: *-text*オプションを指定することで, 図面中の文字列だけを抜きだしたファイルを作成でき, 図面中の文の綴り検査などを行なえるようになりました.

*EPSI*形式のファイルを作成できます
: EPS図形の場合, 枠からその位置を知ることはできますが, 中身を (tgif上で)見ることはできませんでした. EPSIは, EPSに中身のビットマップを組み込むことで, tgif上で中身を見ることができるように拡張したEPSです.

メニュー欄を追加しました
: 一番上の欄のすぐ下にメニュー欄を追加しました. マウス操作によってプルダウン形式のメニューを表示します.

表示欄内の文字列左右にずらすことができるようになりました
: シフトあるいはコントロールキーとマウス操作を組み合わせることで, 表示欄内の文字列を左右にずらすことができるようになりました.

状態欄を追加しました
: 図面の下に状態欄を追加しました. マウスの左, 中央, 右ボタン操作をすると何が起こるのか, メニュー項目は何か, といったことを表示します.

短縮系として起動できる処理を追加しました
: 短縮系として起動できる処理として, *ToggleShowMeasurement*, *ToggleSmoothHinge*, *ToggleShowMenubar*, *ToggleShowStatus*を追加しました.

アニメーションのための内部命令を追加しました
: アニメーションのための内部命令として, *select_obj_by_name*, *unselect_all_obj*, *move_selected_obj_relative*, *repeat*, *hyperjump*, *make_cgi_query*, *wait_click*, *sleep*, *begin_animate*, *end_animate*, *set_redraw*, *set_selected_obj_color*, *set_selected_obj_fill*, *set_selected_obj_pen*, *set_selected_obj_line_width*, *set_selected_obj_spline*,

set_selected_obj_arrow, *set_selected_obj_dash*, *inc*, *dec*, *shuffle_obj_to_top*, *disable_undo*, *enable_undo*, *get_drawing_area*, *set_selected_obj_bbox*, *move_selected_obj_absolute*, *assign*, *strcpy*, *while*, *if*を追加しました.

その他にも内部命令を追加しました
: 内部命令として, *update_xbm_child*, *update_xpm_child*, *read_file_into_attr*, *write_attr_into_file*, *append_attr_into_file*, *EmbedEPSFile*を追加しました.

標準以外の *(*英語*)* フォントを使えるようになりました

*X*リソースに指定できる項目を追加しました
: Xリソースに指定できる項目として, *InitialFontSize*, *MsgFontSize*, *RulerFontSize*, *DefaultFontSize*, *FontSizes*, *AdditionalFonts*, *FontNamePrefix*, *HasAlternateDefaultFonts*, *TGIFPATH*, *TGIFICON*, *StickyMenuSelection*, *PSBopHook*, *PSEopHook*, *ShowMeasurement*, *MinimalEPS*, *InitialPrintInColor*, *InitialShowGrid*, *InitialSnapOn*, *NoMenubar*, *NoStatusWindow*, *ReverseMouseStatusButtons*, *MinimalMenubar*, *ColorBgInPrintingColorPS*, *InitialPaperSize*, *UpdateChildUsingAlignment*, *ScrollBarWidth*, *DomainPath*, *@@@Viewer*, *UsePaperSizeStoredInFile*, *TiffEPSI*, *EPSIExportExtension*, *CanChangeAttrColor*を追加しました.

*X*リソースに指定できる項目の一部はもはや使いません
: Xリソースに指定できる項目の内, *InitialFontDPI*, *InitialFontSizeIndex*, *Domain*, *TGIF_Domain*は使わなくなりました.

編集メニューに追加があります
: 編集メニューに, *SetSelLineWidth()*と*AddColor()*の項目が追加になっています.

*Special*メニューに追加があります
: Specialメニューに, *ImportAttrs()*の項目が追加になっています.

選択部分からの文字列の取り込みはできません
: 選択部分からの文字列の取り込み機能は日本語版に固有の機能なので, tgif-2.16にはこの機能はありません.

訳語についての注

置く
: Instantiate. Tgifでは算体中心 (オブジェクト指向) の用語が用いられており, これもその一つです. 用語としてそのまま流用するならば「具体化」とすべきなのでしょうが, 「部品を図面上で具体化する」といわれても, 何のことだか全くわかりません. Tgifにおいてinstantiateとは, 部品中のシンボルを図面上に転写することを表しており, 「部品を置く」という表現のほうがその雰囲気を表しているように思われるので, ここではinstantiateを「置く」としています. 「記号」, 「シンボル」および「部品」の項も参照して下さい.

記号
: Icon. 原文では, iconは3つの文脈で用いられています. 1) 部品のシンボルを図面上に転写したもののこと, 2) 状態を表している, 選択欄中の「絵」, 3) Xウィンドウ上の「アイコン」. すべてを「アイコン」とするとわかりにくくなると考え, 1) を記号, 2) を印, 3) をアイコンとして区別しています.「置く」, 「シンボル」および「部品」の項も参照して下さい.

シンボル
: Symbol. Tgifでは, 部品は, それを象徴している部分と, それについての解説部分からなり, 前者を表現部, 後者を定義部と呼んでいます. いずれも tgifで扱うことのできる図形ですが, 表現部は, その部品を象徴した「シンボル」として扱い, その部品を積み上げる際に図面上に転写します. 部品の「シンボル」 (表現部) を図面上に転写したものが記号です. 「置く」,「記号」および「部品」の項も参照して下さい.

TGIF
: 米国において, 1週間の仕事が終わって週末を迎えるうれしさを表す意味で「Thank God it's Friday」の略語として「T.G.I.F」が使われることがあるそうです. Tgifの名前の由来はわかりませんが, おそらくこれに引っかけているのでしょう. ということは, 一文字ずつ区切って「ティー・ジー・アイ・エフ」と読むのが正しいのかも知れません.

デフォルト拡大率
: Default zoom. Tgifを起動した時の拡大率のことをデフォルト拡大率といいます.

取手
: Handle. 図形の上やまわりに表示されている小さな黒い四角のことです. これをマウスで「つまむ」ことにより, 図形の大きさを変えたりできます. 単純に「ハンドル」と呼ばれていることもあるかもしれません.

ドメイン
: Domain. シンボルをまとめておくディレクトリのことをtgifでは「ドメイン」と呼んでいます.

ドラッグ
: Drag. ある図形のうえでマウスのボタンを押したままマウスを引きずる操作のことをドラッグといいます.

部品
: Building-block. 直訳すれば積み木. 製図の基本単位として「積み木」というのはわかりにくいので, 「部品」としました. 「置く」, 「記号」および「シンボル」の項も参照して下さい.

文字列
: Text. 文字列と言うとstringのことだと言われてしまうかも知れません. Tgifにおいてtextは, 単に文字列だというだけでなく, それに対してフォントや, 右寄せ, センタリングといった属性を定義できたりするものなので, 厳密に言えば区別するべきかも知れません. しかし, 両者を厳密に区別しなければ理解できないような文脈はなく, しかも「そこの文字列もうちょっと下にずらしてみてよ」というほうが「そこのテキストもうちょっと下にずらしてみてよ」というよりも普通に使われていると考え, textは文字列とすることにしました.

操作禁止
: Lock. Lockした図形はそれ以上操作できなくなるので, ここでは操作禁止と訳しています.

著作権

著作権の詳細についてはファイル「Copyright」を参照して下さい.

*PostScript*はAdobe Systems Incorporatedの商標です.

作者

William Chia-Wei Cheng (william@cs.UCLA.edu)

日本語化は上野 洋一郎 (ueno@cs.titech.ac.jp)

翻訳

Japanese Translation Copyright © MCMXCV by IMOTO Takashi, INAMURA Hiroshi, and HASEGAWA Takasi, Kanagawa, Japan.

このマニュアルの日本語への翻訳は井本貴之 (imoto@nttspch.hil.ntt.jp),稲村浩 (inamura@isl.ntt.jp), 長谷川 敬 (hasegawa@exa.onlab.ntt.jp) です.

この翻訳は, 原著作者William Chia-Wei Cheng (william@cs.ucla.edu) の許可に基づき配布するものです. この翻訳の利用, 配布に関しては, 上記著作権表示, ならびにこの著作権に関する注意書きが含まれている限りにおいて, GPL (GNU Geleral Public License) に準拠して取り扱われるべきものとします.

なお, 訳文に対するご意見などありましたら, ご連絡いただけると助かります.また配布などについてのご意見なども, もしありましたらお寄せいただければ恐縮です.