ε-pTEX　取扱説明書

北川 弘典（H7K）*

2011 年 2 月 27 日，build 110227

1　はじめに

ε-pTEX は，東京大学理学部数学科 3 年生対象の 2007 年度の授業「計算数学 II」*1 において北川が作成したプログラムである．もともとは pTEX 3.1.10 を基盤として，ε-TEX 2.2 相当の機能や 10 進 21 桁の浮動小数点演算を追加したものであった．

しかし，本版においては，黒木氏などからのアドバイスもあり，ε-TEX をベースにして pTEX 拡張を組み込むという実装をとっていて，また，実装の中途半端さから，浮動小数点演算は削除している．

製作の動機や作業過程などについては，詳しくは [1] を参照して欲しいけれども，大雑把に言うと，動機は以下のように要約できる．

- pTEX は，TEX が持っている「レジスタ 1 種類につき 256 個まで」という制限をひきずっており，現状でも非常に多数のパッケージを読み込ませたりすると制限にぶち当たってしまう．
- 一方，ε-TEX 拡張ではこれが「レジスタ 1 種類につき 32768 個まで」と緩和されており，欧文で標準となっている pdfTEX やその後継の LuaTEX，及び XETEX でも ε-TEX の機能が取り込まれている．
- そうすると，pTEX だけが制限をレジスタ制限を引きずっているのは世界から取り残されることになるのではないか．

TEX Live 2010 についに pTEX が取り込まれた．そのため，今後の日本語 TEX 環境は角藤さんの W32TEX と TEX Live 2010 の 2 つに収束していくものと考えられる．そのため，本版では TEX Live 2010 にのみ対応している．

2　ε-TEX 拡張について

前に述べたように，ε-TEX は TEX の拡張の一つである．ε-TEX のマニュアル [9] には，開発目的が以下のように述べられている．

> The $\mathcal{NTS}$ project intends to develop an 'New Typesetting System' ($\mathcal{NTS}$) that will eventually replace today's TEX3. The $\mathcal{NTS}$ program will include many features missing in TEX, but there will also exist a mode of operation that is 100% compatible with TEX3. It will, necessarily, require quite some time to develop $\mathcal{NTS}$ to maturity and make it widely available.

* http://sourceforge.jp/projects/eptex/wiki/, e-mail: h_kitagawa2001(at)yahoo.co.jp

*1 http://ks.ms.u-tokyo.ac.jp/

> Meanwhile ε-TEX intends to fill the gap between TEX3 and the future NTS. It consists of a series of features extending the capabilities of TEX3.

NTS がどうなったのか僕は知らない．しかし，少なくとも ε-TEX 拡張自体は実用的な物であり，そのせいか ℵ (Aleph), pdfTEX, XETEX などの他の拡張にもマージされており，かなりの人が ε-TEX 拡張を使うことができるようになっている．

ε-TEX 拡張で追加される機能について，詳しくは [9] を参照して欲しい．しかしそうやって丸投げするのはよろしくなさそうな気もするので，ひとまず [1] 中の 4.2 節「ε-TEX の機能」を引用することにする（一部改変）：

> ε-TEX には Compatibility mode と Extended mode の 2 つが存在し，前者では ε-TEX 特有の拡張は無効になるのでつまらない．後者がおもしろい．
>
> 拡張機能を使うにはファイル名を渡すときに * をつけるかコマンドラインオプションとして -etex スイッチをつければいいが，ε-TEX 拡張に関わる追加マクロは当然ながらそれだけでは駄目である．「plain マクロ for ε-TEX」（etex.fmt というのが一番マシかな）では自動的に追加マクロである etex.src が呼ばれる．LATEX 下ではちょうど etex.src に対応した etex パッケージを読み込む必要がある．

■レジスタの増加

最初に述べたように，TEX では 6 種類のレジスタが各 256 個ずつ利用できる．それぞれのレジスタには\dimen75 などのように 0 ～ 255 の番号で指定できる他，予め別名の定義をしておけばそれによって指定することもできる．これらのいくつかは特殊な用途に用いられる（例えば\count0 はページ番号などのように）ことになっているので，さらに user が使えるレジスタは減少する．

ε-TEX では，追加のレジスタとして番号で言うと 256 ～ 32767 が使用できるようになった．上の pdf によると最初の 0 ～ 255 と違って若干の制限はあるようだが，それは些細な話である．追加された（各種類あたり）$32768 - 256 = 32512$ 個のレジスタは，メモリの効率を重視するため sparse register として，つまり，必要な時に始めてツリー構造の中で確保されるようになっている．

■式が使用可能に

TEX における数量の計算は充実しているとは言い難い．例えば，

$$\texttt{\dimen123} \leftarrow (\texttt{\dimen42} + \texttt{\@tempdima})/2$$

という計算を元々の TEX で書こうとすると，

```
\dimen123=\dimen42\advance\dimen123by\@tempdima\dimen123=0.5\@tempdima
```

のように書かないといけない．代入，加算代入，乗算代入，除算代入ぐらいしか演算が用意されていない状態になっている（上のコードのように，$d_2 \mathrel{+}= 0.8d_1$ というような定数倍を冠することは平気）．

ε-TEX では，そのレジスタの演算に，他のプログラミング言語で使われているような数式の表現が使えるようになった．上の PDF では実例として

```
\ifdim \dimexpr (2pt-5pt)*\numexpr 3-3*13/5\relax + 34pt/2<\wd20
```

が書かれている．これは，

$$32\,\mathrm{pt} = (2\,\mathrm{pt} - 5\,\mathrm{pt})\,(3 - \mathrm{div}(3 \cdot 13, 5)) + \frac{34\,\mathrm{pt}}{2} < \texttt{\box20 の幅}$$

が真か偽かを判定していることになる．

■**\middle primitive**

TEX に\left, \right という primitive があり，それを使えば括弧の大きさが自動調整されるのはよく知られている．ε-TEX では，さらに\middle primitive が追加された．

具体例を述べる．

$$\left\{ n+\frac{1}{2} \middle| n\in\omega \right\} \left\{ n+\frac{1}{2} \middle| n\in\omega \right\}$$

これは以下の source で出力したものである：

```
\def\set#1#2{\setbox0=\hbox{$\displaystyle #1,#2$}%
\left\{\, \vphantom{\copy0}#1 \,\right|\!\left.\, %
    \vphantom{\copy0}#2 \,\right\}}
\def\eset#1#2{\left\{\, #1 \,\middle|\, #2 \,\right\}}
\[ \set{n+\frac12}{n\in \omega} \eset{n+\frac12}{n\in \omega} \]
```

両方とも集合の表記を行うコマンドである．TEX 流の\set では 2 つの\left, \right の組で実現させなければならず，そのために | の左側と右側に入る式の最大寸法を測定するという面倒な方法を使っている．その上，この定義では\textstyle 以下の数式（文章中の数式とか）ではそのまま使えず，それにも対応させようとすると面倒になる．一方，ε-TEX 流の\eset では，何も考えずに\left, \middle, \right だけで実現できる．

■ **TEX--XETEX** (TeX--XeT)

left-to-right と right-to-left を混植できるという機能であるらしい．ヘブライ語あたりの組版に使えるらしいが，よく知らない．ここでの RtoL は LtoR に組んだものを逆順にしているだけのような気がする．

とりあえず一目につきそうな拡張機能といったらこれぐらいだろうか．他にも tracing 機能や条件判断文の強化などあるが，そこら辺はパッとしないのでここで紹介するのは省略することにしよう．

ε-pTEX ではここに述べた代表的な機能を含め，ほとんどすべての機能を実装しているつもりである．ただ，TEX--XETEX を和文で使うと約物の位置がずれたり空白がおかしかったりするけれども，そこの修正は大変に思えるし，苦労して実装する意味があるのか疑問なので放置している．

pTEX 拡張では，TEX と比較して *dir_node* と disp_node という 2 種類の node が追加された．前者は，現在のリストの中に違う組方向の box を挿入する際に寸法を補正するために作られ，\hbox や\vbox のコンテナとなっている．また後者は，欧文文字のベースライン補正のために使われる．

ε-pTEX-110102 まではこれらの node も\lastnodetype の値として出力させるようにしたが，両者ともにユーザーが意識する必要はないことから，ε-pTEX-110227 以降では *dir_node* と disp_node は\lastnodetype の値として出力しないようにしている．

−1:	none (empty list)	5:	mark node	11:	glue node
0:	char node	6:	adjust node	12:	kern node
1:	hlist node	7:	ligature node	13:	penalty node
2:	vlist node	8:	disc node	14:	unset node
3:	rule node	9:	whatsit node	15:	math mode nodes
4:	ins node	10:	math node		

\currentiflevel における条件判断文とそれを表す数字との対応は，以下のようになっている．21 以降が，pTEX 拡張で追加された条件判断文に対応する．

1:	\if	8:	\ifmmode	15:	\iftrue	**22:**	\ifydir
2:	\ifcat	9:	\ifinner	16:	\iffalse	**23:**	\ifmdir
3:	\ifnum	10:	\ifvoid	17:	\ifcase	**24:**	\iftbox
4:	\ifdim	11:	\ifhbox	18:	\ifdefined	**25:**	\ifybox
5:	\ifodd	12:	\ifvbox	19:	\ifcsname		
6:	\ifvmode	13:	\ifx	20:	\iffontchar		
7:	\ifhmode	14:	\ifeof	**21:**	\iftdir		

3 「FAM256」パッチについて

FAM256 パッチは，掲示板 TEX Q & A の山本氏の書き込み [2] に刺激されて作ったものであり，以下に説明する Ω の一部機能[*2]を使えるようにするパッチである（この名称は便宜的なもの）．**本パッチは ε-TEX 拡張とは関係はありません．**

本版も含め，091003 版以降では，FAM256 パッチは標準で有効になる．ε-TEX 拡張とは違い，FAM256 パッチを有効にしてバイナリを作った場合，追加機能は extendend mode でなくても有効になっている．ただし，後に述べるように，本パッチではレジスタが 65536 個まで使えるようにしているが，それについてだけは extended mode の時に限り有効になる．

3.1 機能解説

本ドキュメントの最後のページ[*3]にちょっとしたサンプルを載せてある．

■数式フォント制限の緩和

Ω の大きな特徴としては，TEX 内部のデータ構造を倍の領域を用いるように改変し[*4]，TEX に従来から存在していた「256 個制限」を 2^{16} 個にまで緩和したことが挙げられる．同様に，Ω では（[2] にもあるように）数式フォントを同時に 256 個まで用いることができ，各フォントも 65536 文字まで許されるようになっている．

FAM256 パッチでは，中途半端だが，数式フォント 1 つあたりの使用可能文字数は 256 個のままで，同時に数式フォントを 256 個まで使えるようにしている．基本的には Ω と同様の方法を用いているが，内部でのデータ構造に違いがある（数字はすべて bit 幅）：

	category	family	char	math code	delimiter code
TEX	3	4	8	$3+4+8=15$	$3+2\cdot(4+8)=27$
Ω	3	8	16	$3+8+16=27$	$(3+8+16,\ 8+16)=(27,24)$
FAM256	3	8	8	$3+8+8=21$	$(3+8+8,\ 8+8)=(19,16)$

TEX に既存のコマンド類は互換性維持のために同じ動作とする必要があるので，16 番から 255 番のフォントを利用する際には別のコマンドが必要となる．（実装自体に Ω の流儀を使っているから）

[*2] LuaTEX ではこれらの機能は使えるはずである．見た感じだと XHTEX では使えないようだ．

[*3] ただし，ソースファイルで言えば fam256d.tex（本文）と fam256p.tex（preamble 部）に対応する．

[*4] 詳しい話は texk/web2c/texmfmem.h 中の共用体 memoryword の定義を参照．大雑把に言うと，1 つの「メモリ要素」に 2 つの 32 bit 整数を同時に格納できるようになっている．

FAM256 では，Ω のコマンド類を流用することにした．すなわち，FAM256 では，以下の primitive が追加されている[*5]．

- `\omathcode` ⟨*8-bit number*⟩ = ⟨*27-bit number*⟩
- `\omathcode` ⟨*8-bit number*⟩
- `\omathchar` ⟨*27-bit number*⟩
- `\omathaccent` ⟨*27-bit number*⟩
- `\omathchardef` ⟨*control-sequence*⟩ = ⟨*27-bit number*⟩
- `\odelcode` ⟨*8-bit number*⟩ = ⟨*27-bit number*⟩⟨*24-bit number*⟩
- `\odelimiter` ⟨*27-bit number*⟩⟨*24-bit number*⟩
- `\oradical` ⟨*27-bit number*⟩⟨*24-bit number*⟩

ここで，27 bit とか 24 bit の自然数の意味については，上の表の Ω の行を参照して欲しい．上に書いた FAM256 での実装から，character code の指定に使われる 16 bit の数値で，実際に使われるのは下位 8 bit であり，上位 8 bit は無視される．なお，`\odelcode`⟨*8-bit number*⟩ として delimiter code を取得しようとしても，現時点のパッチでは，うまく動作しない[*6]．

当然ながら，LaTeX において数式 fam を 16 個以上使うには，`\omathchar` などのコマンドに対応したマクロを使う必要がある．実験的と書かれてはいるが，山本氏による「最低限のパッケージ」[4] が手っ取り早いような気がする．

■無限のレベル

TeX では，glue の伸縮量に 3 つの無限大のレベルが存在した：`fil`, `fill`, `filll` であり，`l` が多いほど無限大のオーダーが高い．Ω では，「inter-letter spacing のために」`fi` という，有限と `fil` の中間にあたる無限大のレベルが付け加えられ，`\hfi`, `\vfi` という 2 つの primitive も追加された．FAM256 パッチでは，この無限大レベル `fi` も採用することにした．

実装方法は，大まかには Ω で `fi` の実装を行っている change file `omfi.ch` の通りであるのだが，これに pTeX や ε-TeX に伴う少々の修正を行っている．

- コマンド`\pagefistretch` を新たに定義している．
- `\gluestretchorder`, `\glueshrinkorder` の動作を ε-TeX のそれと合わせた．具体的には，ある適当な glue `\someglue` の stretch 幅を ⟨*stretch*⟩ とおくとき，

$$\texttt{\textbackslash gluestretchorder\textbackslash someglue} = \begin{cases} 0 & \langle \textit{stretch} \rangle \text{ が高々 } \texttt{fi} \text{ レベルの量} \\ 1 & \langle \textit{stretch} \rangle \text{ がちょうど } \texttt{fil} \text{ レベルの無限量} \\ 2 & \langle \textit{stretch} \rangle \text{ がちょうど } \texttt{fill} \text{ レベルの無限量} \\ 3 & \langle \textit{stretch} \rangle \text{ がちょうど } \texttt{filll} \text{ レベルの無限量} \end{cases}$$

  となっている．内部では `fi` レベルが 1，`fil` レベルが 2，……として処理している．

■レジスタについて

Ω では（前にも書いたが）データ構造の変更が行われ，それによってレジスタが各種類あたり 65536 個使えるようになっている．

一方，ε-TeX では，256 番以降のレジスタを専用の sparse tree に格納することにより，32767 番までのレジスタの使用を可能にしていた．この tree 構造を分析してみると，65536 個までレジスタを拡張

[*5] Ω では ⟨*8-bit number*⟩ のところが ⟨*16-bit number*⟩ になっている．

[*6] 51 bit 自然数を返さないといけないですからねえ．やる気があれば検討してみます．

するのはさほど難しくないことのように思われた．具体的には，tree の階層を 1 つ増やしてみた（だから，おそらく各種類あたり $16 \cdot 32768 = 524288$ 個まで使えるとは思うが，これはきりが悪い）．そこで，FAM256 パッチでは ε-TEX 流の方法を用いながらも，レジスタをさらに 65536 個まで増やしている．

4 LaTeX3 対応と pdfTeX

開発中の LaTeX3 では，ε-TEX 拡張の他に，pdfTEX で導入された\pdfstrcmp（又はその同等品）が必要となっており，もはや純粋な ε-TEX ですら LaTeX3 を利用することはできない状況である ([5, 6, 7])．そのため，ε-pTEX の 101231 版以降では，\pdfstrcmp を実装した．角藤さんによる W32TEX での実装が参考になった[*7]ので，角藤さんにはお礼を申し上げたい．\pdfstrcmp は，2 つの引数をとり，それぞれの内容を文字列化したものを先頭バイトから順番に比較するという動作を行うものである．110102 版以降では，このときの和文文字の符号化に（ε-pTEX の内部漢字コードにかかわらず）UTF-8 を採用している．

また，pdfTEX では，「指定箇所の dvi/pdf における出力位置」を，\pdfsavepos, \pdflastxpos, \pdflastypos を利用して知ることができる．例えば，ルビの行頭/行末/行中形自動判定に応用がきくだろう．そのため，ε-pTEX 110227 版以降では，この primitive を実装した．

- \pdfpagewidth, \pdfpageheight: ページの「幅」「高さ」．
  ここで言う「幅」は「字送り方向」のことではない．
- \pdflastxpos, \pdflastypos: \pdfsavepos が置かれた場所の，dvi における出力位置を返す．原点は紙の（物理的な意味の）左下隅であり，y 軸は（物理的な）上方向に向かって増加する．

pdfTEX では，組方向は 1 つのみだが，pTEX ではそうでないので，\pdflastxpos, \pdflastypos の値の座標系を「物理的な」向きとすべきか，それとも「組方向に応じた」向きとすべきかは悩みどころである．110227 版では，上の説明の用に物理的な向きとしたが，将来の版では変わるかもしれない．

なお，pdfTEX には（主要機能の pdf 出力や micro-typesetting の）他にも，便利な primitive がいくつか実装されている ([11])：

- \ifincsname：\csname ... \endcsname の内部にいるか判定する
- \pdfcreationdate：本来は pdf の作成日時を表す文字列を取得するものだが，これを用いて実行時刻の「秒」を知ることができる（通常の TEX では分単位までしかわからない）．
- \vadjust pre：本来，\vadjust は現在の行の「直後」に box を挿入する命令だが，このように pre をつけると「直前」に挿入できる．
- \pdfuniformdeviate 他：乱数．MetaPost で採用されている「引き算法」を用いているようだ．

これらの ε-pTEX への実装は，やろうという計画はあるが，今は見送っている．

[*7] W32TEX での実装は，XeTEX のものを元にしているので，対応する primitive の名称は\strcmp である他，後に述べている\vadjust pre 等も組み込まれている．困った時には，ファイルの最初にでも
\ifdefined\strcmp\else\let\strcmp=\pdfstrcmp\fi とでも書けば，\strcmp の名前で使える．

5　互換性

ε-TEX, pTEX との互換性をはかるのに有効な手段としては，まず TRIP test がある．[1] にも書いたが，これは TEX のソースの全行を実行するようなテストソースであり，TRIP test の実行結果が違えば，どこかの動作が違っていることが分かるという仕掛けである．

幸いにも，ε-pTEX の（compatibility mode における）TRIP test の実行結果は，pTEX のそれとほとんど同じであり，違う点は以下のみであった：

- Memory usage の違い：

  Memory usage before: x&y; after: z&w; still untouched: u

  のところで，x, z が 4 多く，u が 4 少ない．
- （FAM256 パッチのおかげで）数式フォントが 256 個同時に扱えるようになったことによる，エラーの未発生やエラーメッセージの違い．
- ログの最後に出てくる，メモリ総使用量．

また，extended mode と compatibility mode による出力の違いを検証すると，Memory usage の違いやメモリ総使用量の他には，e-TRIP の説明書 p. 4 の項目 4. にある違いしか見受けられなかった，

次に，ε-TEX には，ε-TEX によって拡張された部分を調べる，同様の e-TRIP test が存在する．pTEX 系列では，\showbox などでいちいち組方向を表示するので，それを

```
$ alias sep='sed "s/, yoko direction//;s/yoko direction, //;\
  s/yoko(math) direction, //"'
```

という alias を利用することで log file から除去し，それと ε-TEX での e-TRIP test の出力と比べた結果，

- Memory Usage の違い（略）
- レジスタの使用可能個数を 65536 個/種類としたことによるエラーの未発生
- （和文文字のため）character code に 256 以上も許すことによるエラーの未発生．

という違いを得た．

すると，最後に，pTEX 特有の拡張部分のテストをしたくなるわけだが，北川の知る限りにおいては，そのようなテストソースは公式には存在しないようである．しかし，やはり気になる問題であり（さる方面から圧力もきたので），個人的に作ってみたものが ptex-qtrip である．詳細な説明はそちらのドキュメントに譲る．ともかく，これを動作させて log file を比べてみると，メモリ関連や FAM256 関連以外は

```
-% split2 to -0.01802,6.94444 p=-10000
+% split2 to -0.01806,6.94444 p=-10000
```

以外の違いしかなかった．わずか 0.00004 の違い（単位がもし point だとしたら 3 sp）だが，どこか気になるところである．

上の箇所もあるし，また ptex-qtrip 自体がまだ完全なものではなのだが，pTEX と ε-pTEX は実用上においては互換と言えそうな気もしなくもない．

6　最後に

[1] にも書いたような気がするが，ε-pTEX は単につなぎでしかない，と思っている．upTEX, ε-pTEX によってある程度の新機能が利用可能になっている間に，XETEX や LuaTEX といった次世代の TEX に移行する準備を整えていくのが良いのではないだろうか？

参考文献

[1] 北川 弘典，「計算数学 II　作業記録」，2008.
https://sourceforge.jp/projects/eptex/document/resume/ja/1/resume.pdf 他

[2] 山本 和義，「数式 fam の制限と luatex」，掲示板「TEX Q & A」52744 番書き込み，2009.2.12，
http://oku.edu.mie-u.ac.jp/~okumura/texfaq/qa/52744.html

[3] 山本 和義，「Re: 数式 fam の制限と luatex」，掲示板「TEX Q & A」52767 番書き込み，2009.2.16，
http://oku.edu.mie-u.ac.jp/~okumura/texfaq/qa/52767.html

[4] 山本 和義，「数式 fam 拡張マクロ for e-pTeX 等」，掲示板「TEX Q & A」52799 番書き込み，
2009.2.21，http://oku.edu.mie-u.ac.jp/~okumura/texfaq/qa/52799.html

[5] 河原，「パッケージとディストリビューションについて」，掲示板「TEX Q & A」55464 番書き込み，2010.12.16，http://oku.edu.mie-u.ac.jp/~okumura/texfaq/qa/55464.html

[6] 角藤 亮，「Re: パッケージとディストリビューションについて」，掲示板「TEX Q & A」55478 番書き込み，2010.12.19，http://oku.edu.mie-u.ac.jp/~okumura/texfaq/qa/55478.html

[7] zrbabbler，「LaTeX3 と expl3 パッケージ」，ブログ「マクロツイーター」内，2010.12.22，
http://d.hatena.ne.jp/zrbabbler/20101222/1293050561

[8] 角藤 亮，「Re: e-pTeX 101231」，掲示板「TEX Q & A」55528 番書き込み，2011.1.1，
http://oku.edu.mie-u.ac.jp/~okumura/texfaq/qa/55528.html

[9] The $\mathcal{NTS}$ Team. *The ε-TEX manual.*
http://www.tug.org/texlive/Contents/live/texmf-dist/doc/etex/base/etex_man.pdf

[10] J. Plaice, Y. Haralambous. *Draft documentation for the Ω system*, 1999.
http://www.tug.org/texlive/Contents/live/texmf-dist/doc/omega/base/doc_1.8.tex

[11] Hàn Thế Thành et al. *The pdfTEX user manual*, 2010.
ftp://ctan.tug.org/tex-archive/systems/pdftex/manual/pdftex-l.pdf

Test source for FAM256 patch

本ソースは山本和義氏による「数式 fam の制限と luatex」（qa:52744）中のコードをベースにしたものである.

■ **More than 16 math font families.**

A**BCDEFGHIJKL fam = 19**

AaAbAcAdAeAfAgAhAiAjAkAlAmAnAoAp fam35

AaAbAcAdAeAfAgAhAiAjAkAlAmAnAoAp fam51

AaAbAcAdAeAfAgAhAiAjAkAlAmAnAo**Ap fam67**

Aa fam68 Ab fam69 Ac fam70 Ad fam71 roman

■**\omathchar etc.**

\mathchar"7F25 : %, \omathchar"7420125 : %

meaning of \langle: macro:->\delimiter "426830A ,

meaning of \lx: macro:->\odelimiter "4450068"030001

\lx : h, \bigl\lx :), \Bigl\lx :)

\the\mathcode‘\f: 7166, \the\omathcode‘\f: 7010066（どちらも 16 進に変換した）

t ⟩⟩))) e e

√, p $\sqrt{a}$, p■ $\sqrt{\int_V f\,d\mu}$, $\sqrt{\int_V f\,d\mu}$

\odelcode primitive による **delimiter code** の取得はうまく動かない：
\the\delcode‘\|: 618, \the\odelcode‘\|: −1（どちらも 10 進）

■ **Infinite level “fi”**

■(fi)■(fil)■(fill)■		(filll)		■
■(fi)■(fil)■		(fill)		■
■(fi)■		(fil)		■
■	(fi) ■	(fi)	(fi)	■

■ **65536 registers**

fuga! a 漢字仮名sdf TEX ほげほげTEX

589