Rechnen mit TEX

Leah Neukirchen

<leah@vuxu.org>

BayTEX 2026, München

2026-08-08

Motivation

Normale Dokumente haben nur geringe Anforderungen an Arithmetik:

- Zähler (Seitenzahlen, Überschriften, Fußnoten etc.)
- Berechnungen für den Schriftsatz/Layout

Für spezielle Anwendungen ist nützlich:

- in Rechnungen: Salden und Steuern
- wissenschaftliche Anwendungen
- Mathematische Übungsaufgaben
- Spielereien

Register in TEX82

Der klassische TEX-Kern bietet uns:

- 256 Register \count*N* von $-2^{31} + 1$ bis $2^{31} - 1$.
- 256 Register \dimen*N* von $-2^{30} + 1$ bis $2^{30} - 1$ sp (*scaled points*).

Andere Längeneinheiten werden im Fixpunktformat dargestellt (denn 1982 gab es noch kein IEEE 754 und selten FPU), z. B. ist

$$1\,\text{pt} = 65536\,\text{sp}.$$

Die größte darstellbare Länge ist damit

$$1073741823\,\text{sp} = 16383.99999237\,\text{pt} \approx 5.76\,\text{m}.$$

Rechnen in TEX82

TEXbietet einige Rechnenprimitiven an:

- `\advance ⟨counter⟩ by …`
- `\multiply ⟨counter⟩ by …`
- `\divide ⟨counter⟩ by …`

Subtraktion fehlt. Man kann nur mit Ganzzahlen multiplizieren und dividieren.

Beispiel: Divisionsrest

```
\def\remainder#1modulo#2{%
  \count@=#1%
  \divide#1 by #2%
  \multiply#1 by #2%
  \multiply#1 by -1%
  \advance#1 by \count@}

\newcount\a\newcount\b
\a=67\b=8

$\the\a\ \equiv\
\remainder \a modulo \b%
\the\a \pmod{\the\b}$
```

$67 \equiv 3 \pmod{8}$

ε-TEX

Quasi alle Engines heutzutage haben die Erweiterungen aus ε-TEX, und damit:

- 32678 Register
- `\numexpr`
- `\dimexpr`

Hauptsächlich schönere Syntax, kompliziertere Rechnungen werden einfacher; für Zwischenergebnisse sind keine Register notwendig:

```
\mycounter=\numexpr{1+2*3-4}
```

(mit „Punkt vor Strich“ und Subtraktion!)

Falls man kein ε-TEX hat: intcalc bietet ähnliches.

fp

Das Paket fp ist recht alt und unterstützt *plain* TEX, LATEX.

Zahlenbereiche: -99999999999999999.99999999999999999 bis +99999999999999999.99999999999999999 (18 Stellig)

- Exponentialfunktionen
- Trigonometrie
- Polynomgleichungen bis zum 4. Grad

```
\FPeval{result}{sin(3.5)/2 + 0.0002}
\result
```

bigintcalc

Mit dem Paket bigintcalc kann man mit beliebig großen Ganzzahlen arbeiten.

- Grundrechnenarten
- Arithmetische Shifts
- Potenzen

```
\(50! = \bigintcalcFac{50}\)
\(67^{42} = \bigintcalcPow{67}{42}\)
```

expl3 / l3int

LaTeX 3 bietet die TeX-Primitiven in der expl3-Syntax an:

```
\int_eval:n {⟨int expr⟩}
\int_div_round:nn {⟨int expr1⟩} {⟨int expr2⟩}
\int_div_truncate:nn {⟨int expr1⟩} {⟨int expr2⟩}
\int_max:nn {⟨int expr1⟩} {⟨int expr2⟩}
\int_min:nn {⟨int expr1⟩} {⟨int expr2⟩}
\int_mod:nn {⟨int expr1⟩} {⟨int expr2⟩}

\int_add:Nn ⟨integer⟩ {⟨int expr⟩}
\int_sub:Nn ⟨integer⟩ {⟨int expr⟩}
\int_decr:N ⟨integer⟩
\int_incr:N ⟨integer⟩
```

Komischerweise fehlt `\int_mul`, aber es gibt ja *.

expl3 / l3fp

Das LaTeX 3-Paket `l3fp` bietet *dezimale* Fließkommazahlen an.

Zusätzliche Operationen:

- Quadratwurzeln
- Exponentialfunktionen
- Trigonometrie

Zahlenbereiche: $\pm 0..10^{16} \cdot 10^{-10000..10000}$

```
\ExplSyntaxOn\fp_to_decimal:n{sin(3.5)/2 + 2e-3}
```

Nutzbar auch in LaTeX 2 mit Paket xfp und `\fpeval`.

Zahlen formatieren mit siunitx

Mittels siunitx kann man z. B. die l3fp-Zahlen konfigurierbar darstellen:

```
\ExplSyntaxOn
\fp_set:Nn \l_zahl {123456789.3456}
\fp_use:N \l_zahl
\num{\fp_use:N \l_zahl}
\num[exponent-mode=engineering]{\fp_use:N \l_zahl}
\num[round-mode=places,round-precision=2]{\fp_use:N \l_zahl}
```

123456789.3456

123 456 789.3456

$123.456\,789\,345\,6 \times 10^{6}$

123 456 789.35

didec

Das Paket didec rechnet mit Zahlen die genau 2 Dezimalstellen haben (für Geldbeträge etc.)

Verwendet l3fp.

```
\didecnew{summe}
\didecadd{summe}{11.22}
\didecadd{summe}{333.55}
\didecmulfp{summe}{1.19}        % korrekt gerundet
\didecuse{summe}
```

xintexpr

Für ε-TEX, LATEX, und ConTEXt LMTX gibt es xintexpr, das mit beliebig großen Zahlen und beliebigen Nachkommastellen rechnen kann.

Ganzzahloperationen sind deutlich schneller als mit bigintcalc.

```
\xintSetDigits*{62}
\xintfloateval{cos(3Pi/17)*sin(1)^0.123 + log(3.42e5)}

% Das dauert viel zu lange mit bigintcalc:
400!=\xintiieval{400!}
```

Es unterstützt maximal 62 Nachkommastellen Genaugigkeit.

Hat sehr viele Funktionen (bis zu Arrays, Satz von Bezout, Kettenbrüchen etc.), 750 Seiten Handbuch.

pgfmath

PGF, die Grafik-Engine von TikZ braucht viele Berechnungen. Standardmäßig werden TEX-Register verwendet.

Mit `\usetikzlibrary{fpu}`: $\pm 10^{324}$, mit 4 bis 6 Stellen Genauigkeit.

LuaTeX

Wer LuaTeX verwendet, kann natürlich auch mittels Lua rechnen:
endlich IEEE 754 Doubles!

```
\newcount\mycount
\newdimen\mydimen

\directlua{
  tex.count['mycount'] = 6 * 7
  tex.dimen['mydimen'] = 6.7 * 4.8 * 65536
  tex.print(math.sin(7.8))
}
```

Leider gibt es ohne Weiteres keine großen Ganzzahlen o. Ä. und nur wenige mathematische Funktionen.

ConTEXt LMTX

Die neue Engine LMTX von ConTEXt hat eine Variante von `\numexpr` und `\dimexpr`, die auch abgeschnittene Division `:` und modulo `;` anbietet.

Zusätzlich gibt es Lua-Module:

`xmath` bietet noch mehr mathematische Funktionen der C-Standardbibliothek an.

`xcomplex` bietet die mathematische Funktionen der C-Standardbibliothek für komplexe Zahlen an.

`xdecimal` implementiert dezimale Fließkommazahlen beliebiger Genauigkeit.

Zusammenfassung

Kurzgesagt:

- Mit LaTeX verwendet man am besten `l3int`/`l3fp`.
- Sonst, oder wenn man die Funktionalität braucht, xintexpr.

Vielen Dank!

Noch Fragen?