Thermistor Table

This worksheet develops a thermistor temperature table in the general format used by Marlin firmware. The objective here, though, is to produce a smaller table with improved numerical precision. Specifically, the worksheet generates a source code file suitable for inclusion in the mpmd_marlin_1.1.x project (https://github.com/aegean-odyssey/mpmd_marlin_1.1.x), a derivative of Marlin firmware carefully crafted for the Monoprice MP Mini Delta 3d printer.

```
STATUS: EXPERIMENTAL$

SENSOR: "TDK EPCOS B57561G1104+000";
NUMBER: 1;
```

(SENSOR) *TDK EPCOS B57561G1104+000*

(NUMBER) *1*

1 Single Beta Model

```
SINGLE_BETA: R = Ro·exp(β·((1/((t/TS)+K))-(1/(To+K))));
u(t) := subst(SINGLE_BETA, AS·R/(R+Rp));
define(u(t), u(t))$

define(f(x), rhs(first(solve(x=u(t), t))));
```

$$\left(\frac{1}{\frac{t}{TS}+K}-\frac{1}{To+K}\right)\beta$$

(SINGLE_BETA) $R = Ro\ \%e$

$$u(t) := subst\left(SINGLE\_BETA, \frac{AS\ R}{R+Rp}\right)$$

$$f(x) := \frac{TS\ To\ \beta+(-K\ TS\ To-K^2\ TS)\ log\left(\frac{Rp\ x}{AS\ Ro-Ro\ x}\right)}{\beta+(To+K)\ log\left(\frac{Rp\ x}{AS\ Ro-Ro\ x}\right)}$$

1.1 Parameters

```
β : 4092;
Tβ : [0,100];
Ro : 100000;
To : 25;
```

(β) *4092*

(Tβ) *[0,100]*

(Ro) *100000*

(To) *25*

2 Steinhart-Hart Model

Source: https://en.wikipedia.org/wiki/Steinhart–Hart_equation

```
∛(x) := x^(1/3)$

STEINHART_HART: [ R=exp(∛(y-x) - ∛(y+x)), /* where */
    y=√((B/(3·C))^3+x^2), x=(1/(2·C))·(A-1/(t/TS+K)) ];

u(t) := subst(STEINHART_HART, AS·R/(R+Rp));
define(u(t), u(t))$

define(f(x), block([Q:1/T = A + B·log(R) + C·log(R)^3],
    rhs(first(solve(subst([T=(t/TS)+K, R=Rp·x/(AS-x)], Q), t)))));
```

(STEINHART_HART)
$$[R=\%e^{(y-x)^{1/3}-(y+x)^{1/3}},y=\sqrt{x^2+\frac{B^3}{27\,C^3}},x=\frac{A-\frac{1}{\frac{t}{TS}+K}}{2\,C}]$$

$$u(t):=subst\left(STEINHART\_HART,\frac{AS\,R}{R+Rp}\right)$$

$$f(x):=-\frac{C\,K\,TS\,log\left(\left(\frac{Rp\,x}{AS-x}\right)^3\right)+B\,K\,TS\,log\left(\frac{Rp\,x}{AS-x}\right)+(A\,K-1)\,TS}{C\,log\left(\left(\frac{Rp\,x}{AS-x}\right)^3\right)+B\,log\left(\frac{Rp\,x}{AS-x}\right)+A}$$

2.1 Coefficients A, B, C

```
R1: 100000;
T1: 25;

R2: R1·0.012967;
T2: 160;

R3: R1·0.0026693;
T3: 240;

block([L1,L2,L3,Y1,Y2,Y3,γ2,γ3,K:273.15],
    L1:log(R1), L2:log(R2), L3:log(R3),
    Y1:1/(T1+K),Y2:1/(T2+K),Y3:1/(T3+K),
    γ2:(Y2-Y1)/(L2-L1),
    γ3:(Y3-Y1)/(L3-L1),
    C: (γ3-γ2)/(L2-L1)·(1/(L1+L2+L3)),
    B: γ2-C·(L1^2+L1·L2+L2^2),
    A: Y1-(B+L1^2·C)·L1)$

A: A,numer;
B: B,numer;
C: C,numer;
```

(R1) 100000

(T1) 25

(R2) 1296.7

(T2) 160

(R3) 266.93

(T3) 240

(A) $6.345049548092097\ 10^{-4}$

(B) $2.319042773596469\ 10^{-4}$

(C) $3.251279739207511\ 10^{-8}$

3 Printer Configuration

```
K:273.15$  /* °C to °K  */
```

```
Rp : 4700;
```

(Rp) 4700

```
x_label : "ADC (16x)"$
y_label : "T °C (x100)"$
AS : 16 ·1024;
TS : 100;
```

(AS) 16384

(TS) 100

4 Piece-wise Linear Approximation

```
CCF(f,xa,fa,xb,fb,δ,ε) := block([xm,fm],
    xm: (xa+xb)/2, fm: apply(f,[xm]),
    if (xb-xa) <= ε or abs((fa+fb)/2 -fm) <= δ then return([[xa,fa]]),
    append(CCF(f,xa,fa,xm,fm,δ,ε), CCF(f,xm,fm,xb,fb,δ,ε)))$

PLA(f,xa,fa,xb,fb,d,e) := map(round,
    append([[0,32767]],CCF(f,xa,fa,xb,fb,d,e),[[xb,fb]],[[AS-1,-32768]]))$

hi : 300;
lo : 3;
RANGE:[x, u(hi·TS), u(lo·TS)]$
TABLE:PLA(f,u(hi·TS),f(u(hi·TS)),u(lo·TS),f(u(lo·TS)),TS/2,2),numer;
ENTRIES:length(TABLE);
/*table_form(append([[x_label,y_label]], TABLE))$*/
```

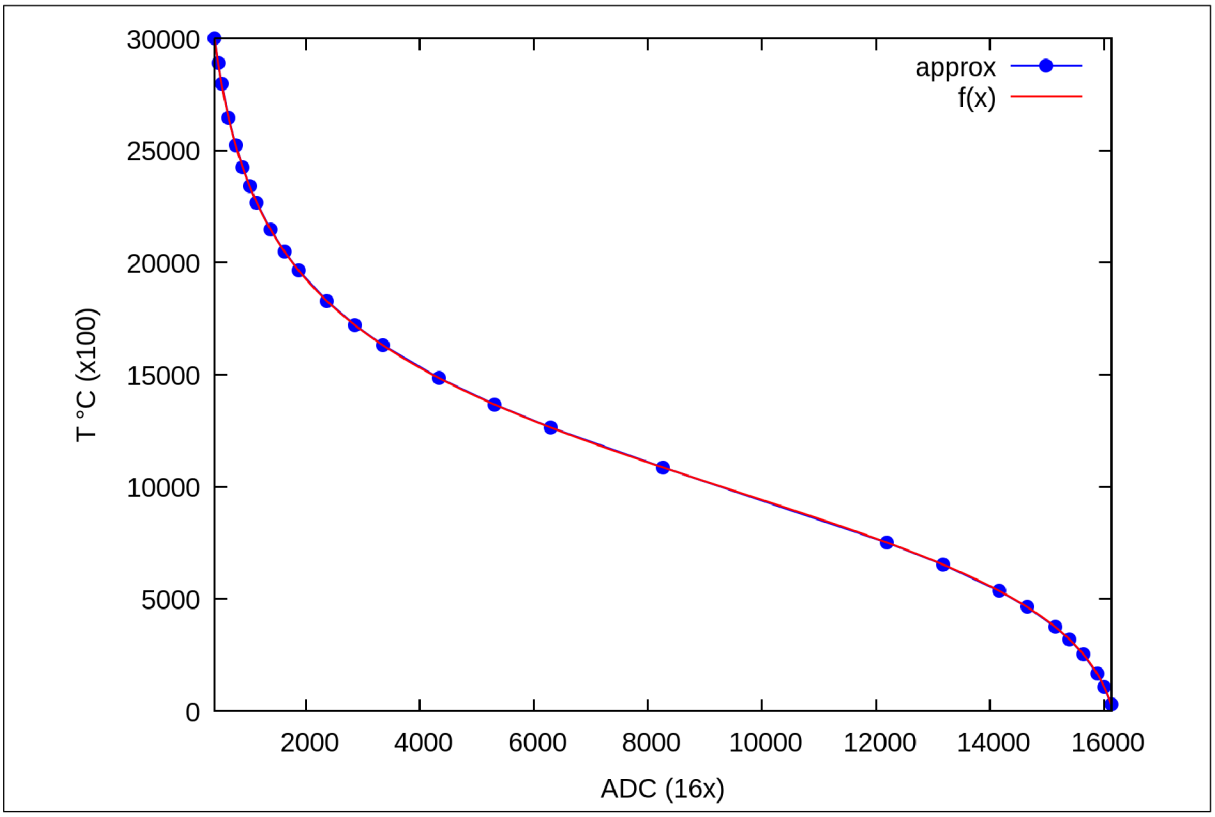
(hi) 300

(lo) 3

(TABLE) [[0,32767],[402,30000],[463,28898],[525,27963],[648,26443],[770,25239],[893,24247],[1016,23406],[1139,22677],[1385,21463],[1630,20475],[1876,19643],[2368,18293],[2859,17216],[3351,16316],[4333,14852],[5316,13664],[6299,12641],[8265,10864],[12197,7512],[13179,6532],[14162,5359],[14654,4640],[15145,3753],[15391,3205],[15637,2537],[15882,1658],[16005,1074],[16128,300],[16383,−32768]]

(ENTRIES) 30

```
wxplot2d([[discrete,TABLE],f(x)],RANGE,[style,[linespoints,1,6],[lines]],
    [legend,"approx","f(x)"],[xlabel,x_label],[ylabel,y_label],
    [gnuplot_preamble,"set terminal pngcairo fontscale 3 linewidth 3 size 1800,1200"])$
```



5 C Source File

```
LIMIT: 31$
WARNING: "The table length of ~d entries exceeds our ~d limit."$
if ENTRIES > LIMIT then warning(printf(false, WARNING, ENTRIES, LIMIT))$

FILENAME: printf(false,"thermistortable_~d.h", NUMBER)$
with_stdout(FILENAME, block([so:true],
    printf(so,"/* CUSTOM THERMISTOR TABLE ~d~%~%", NUMBER),
    printf(so,"~tSENSOR: ~a~%~%", SENSOR),
    printf(so,"~tBβd/~d: ~f~%", Tβ[1], Tβ[2], β),
    printf(so,"~tRo: ~dΩ (resistance @~d°C)~%", Ro, To),
    printf(so,"~%"),
    printf(so,"~tSteinhart-Hart Coefficients~%"),
    printf(so,"~t~fΩ @~f°C, ~fΩ @~f°C, ~fΩ @~f°C~%", R1,T1, R2,T2, R3,T3),
    printf(so,"~tA: ~g~%", A),
    printf(so,"~tB: ~g~%", B),
    printf(so,"~tC: ~g~%", C),
    printf(so,"~%"),
    printf(so,"~tRp: ~dΩ (pull-up resistor)~%", Rp),
    printf(so,"~tAS: ~d (10-bit A/D oversampling)~%", AS/1024),
    printf(so,"~tTS: ~f (temperature scaling)~%", TS),
    printf(so,"~%"),
    printf(so,"~ttemperature range: ~d°C - ~d°C~%", lo, hi),
    printf(so,"~ttable entries: ~d (~d line segments)~%", ENTRIES, ENTRIES-1),
    printf(so,"~tfit tolerance: ±~f°C~%", 1/2),
    printf(so, "*/~%~%"),
    printf(so,"//const float tempscale_~d = ~f;~%~%", NUMBER, TS),
    printf(so,"const short tablesize_~d = ~d;~%~%", NUMBER, ENTRIES),
    printf(so,"const short temptable_~d[][2] = {~%", NUMBER),
    map(lambda([s], printf(so, "~t{ ~d, ~d }, ~%", s[1], s[2])), TABLE),
    printf(so,"};~%")))$;
```