測定データを USB で取得しグラフ化

GP-IB　　USB　　82357B

1. はじめに

測定器のリモート制御とデータ転送のためのインターフェースは、近年は USB に移行している。USB は、その取扱いが簡便でかつあらゆる PC の標準 IO となっているため、できれば USB に統一したいところである。しかし、数十年前に製造された測定器もまだ現役であり、当時の標準であった GP-IB や RS-232C への対応が欠かせない現状である。

筆者の使用している測定器においても、あるものは USB になっているが、多くは GP-IB または RS-232C しか使えない。

Agilent(Keysight)、Contec が GP-IB USB 変換器を発売しており、個人レベルでも EasyGP-IB による RS-232C への変換などの対応策がある。しかし、前者は高額であり、後者はユニットの自作が必要である。たまたま Agilent の純正 GP-IB USB 変換器である 82357B の偽造品 (と思われる品物) が安価で Web 市場にあるので、これを使ってみることにした。

結論から言うと、入手した 82357B が動作上の支障はなく使えるのである。Agilent が特に意匠権を主張しているようには見えないので、(道義的なうしろめたさを感じながら) これを使用する例を紹介する。

2. 測定器と PC のインターフェース

図 1 は筆者の現用している測定器と 82357B を介したプログラミング環境の構成図である。

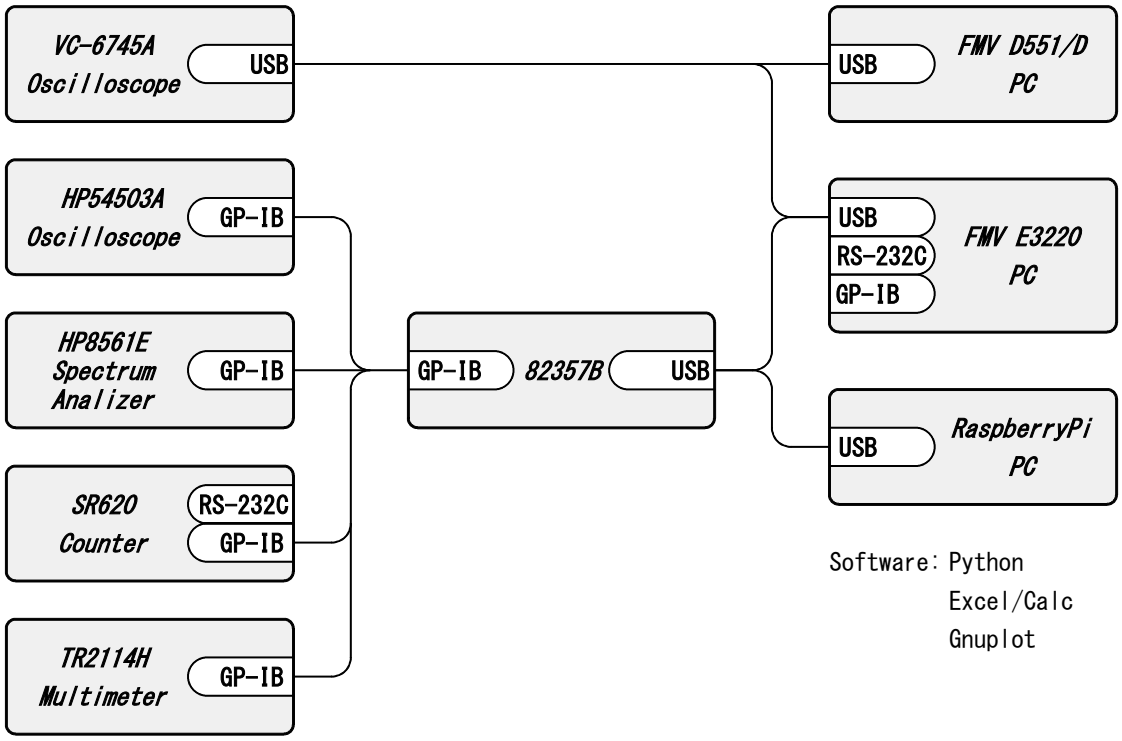

図 1: 82357B 周りの接続

図で分かるとおり、PC の外部接続には USB しか使用しない。USB RS-232C 変換ケーブルと呼ばれる簡便な道具もあるが、この構成ではその出番はない。

3. データ取得とグラフ化のコンセプト

測定器メーカが測定器専用のユーティリティソフトを提供している例が多い。これを利用すると手軽にデータをグラフ化して表現することができので便利ではあるが、デザインが固定されているので電子出版などへの引用には困ることがある。

プログラミングの機能は、遠隔制御・遠隔測定・データ転送が含まれている。しかし、実験室での測定結果をグラフ化するなどの個人的用途では、測定器のパネルを手動操作して測定を完了し、結果のみを PC 上にファイルとして取り込むだけで足りる。したがって、複雑なプログラムを組むまでもなく、データ転送に特化して利用するのが効率的である。

そこで、次項の手順により、生データを転送・編集することとする。

4. 82357B のユーティリティの使用

測定器と PC のハードウェア接続を点検するには、Keysight が提供する IOLibraries Suite を使う。次のダウンロードサイトから取得・インストールする。使用できる OS は Windows、MAC または Linux(Ubuntu) であり、Ver1.5 は 32 ビット OS 用で Ver1.8 は 64 ビット用の違いがある。

```
https://www.keysight.com/ja/pd-1985909/io-libraries-suite?cc=JP&lc=jpn
```

これでインストールされた Connection Expert によって接続されている測定器のデバイス ID・アドレスなどの情報を得ることができる。

IOLibraries Suite は常駐するが、実際の測定器との通信には特に必要としない。後述するとおり、通信接続は Python などが行う。

5. 測定器ごとの具体的な手順

5.1 VC-6745A Oscilloscope の場合

(1) USB ケーブルで USB 接続
(2) VC-6745A で観測し、STRAGE HOLD に設定
(3) メーカー製の Slink.exe を起動
(4) スコープ 波形取り込み 一回
(5) ファイル 波形の保存 テキスト形式
(6) CSV
(7) テキストエディタで、コンマ>>スペース変換、不要個所の削除
(8) gnuplot.exe を起動
(9) load 'vc-6745a.fmt'
(10) print 'vc-6745a.pdf'

[参考] vc-6745a.fmt の内容例

```
\#A parameter file to draw "vc-6745a.dat" for GNUPLOT (c)2020 JA5FP
#set terminal wxt
set title "Wave form on VC-6745A   (c)2020 JA5FP" font "Arial, 10"
set grid back
set xrange [-0.0008:0.0010]
set xtics -0.0008, 0.0002, 0.0010 nomirror
set xlabel "time(s)"
set yrange [-2:2]
```

```
set ytics -2, 0.5, 2 nomirror
set ylabel "voltage(V)"
set key left
plot 'vc-6745a.dat' u 1:2 w l t 'CH1', '' u 1:3 w l t 'CH2'
```

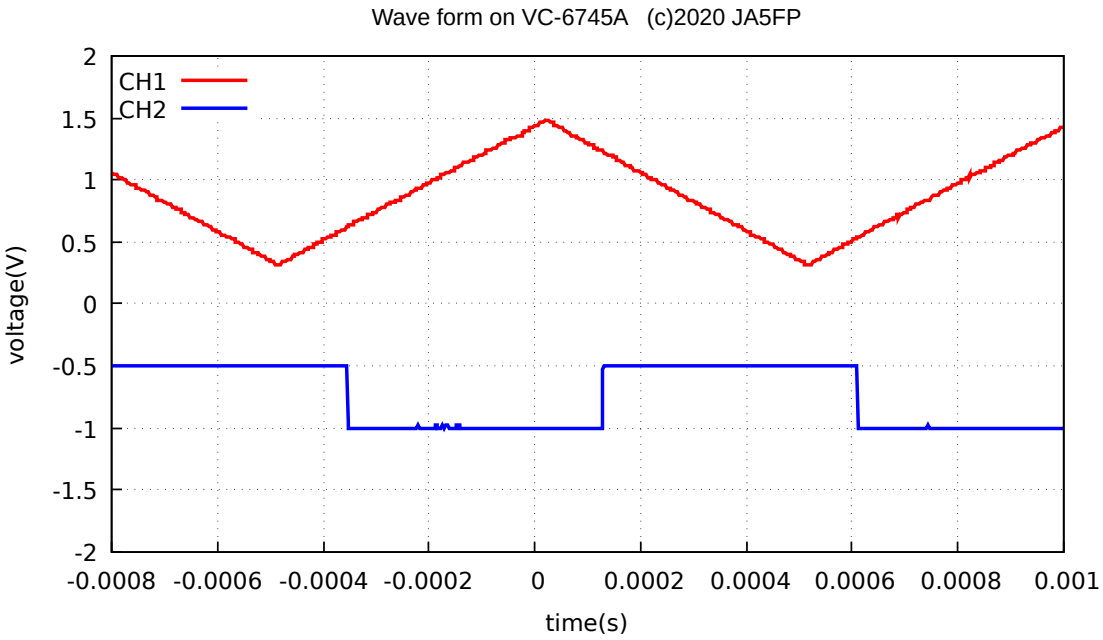

図 2: VC-6745A のデータから作成した pdf

5.2 HP54503A Digitizing Oscilloscope の場合

(1) 82357B で GP-IB 接続
(2) HP54503A で観測し、WAVEFORM SAVE
(3) PC にインストールしてある python.exe を起動
(4) import visa
(5) rm=visa.ResourceManager()
(6) dev=rm.open_resource('GPIB0::7')
(7) store=open('hp54503a.txt', 'a')
(8) dev.write(':MEAS:SOUR WMEM1')
(9) dev.write(':MEAS:VPP?')
(10) line=dev.read()
(11) print('Ch1\n'+line)
(12) store.write('Ch1\n'+line)
(13) dev.write(':MEAS:SOUR WMEM3')
(14) dev.write(':MEAS:VPP?')
(15) line=dev.read()
(16) print(Ch3\n +line)
(17) store.write('Ch3\n'+line)

(18) store.close()
(19) dev.close() までで'hp54503a.txt' を取得
(20) テキストエディタで、コンマ>>スペース変換、不要個所の削除
(21) Excel または Calc で整形し'hp54503a.dat' と変名
(22) gnuplot.exe を起動
(23) load 'hp54503a.fmt'
(24) print 'hp54503a.pdf'

[参考] hp54503a.fmt の内容例

```
#A parameter file to draw "hp54503a.dat" for GNUPLOT (c)2020 JA5FP
#set terminal wxt
set title "Wave form on HP54503A   (c)2020 JA5FP" font "Arial, 10"
set grid back
set xrange [0.0:5.0]
set xtics 0.0, 0.5, 5.0 nomirror
set xlabel "time(ms)"
set yrange [-2:2]
set ytics -2, 0.5, 2 nomirror
set ylabel "voltage(V)"
set key left
plot 'hp54503a.dat' u 1:2 w l t 'CH1', '' u 1:3 w l t 'CH2'
```

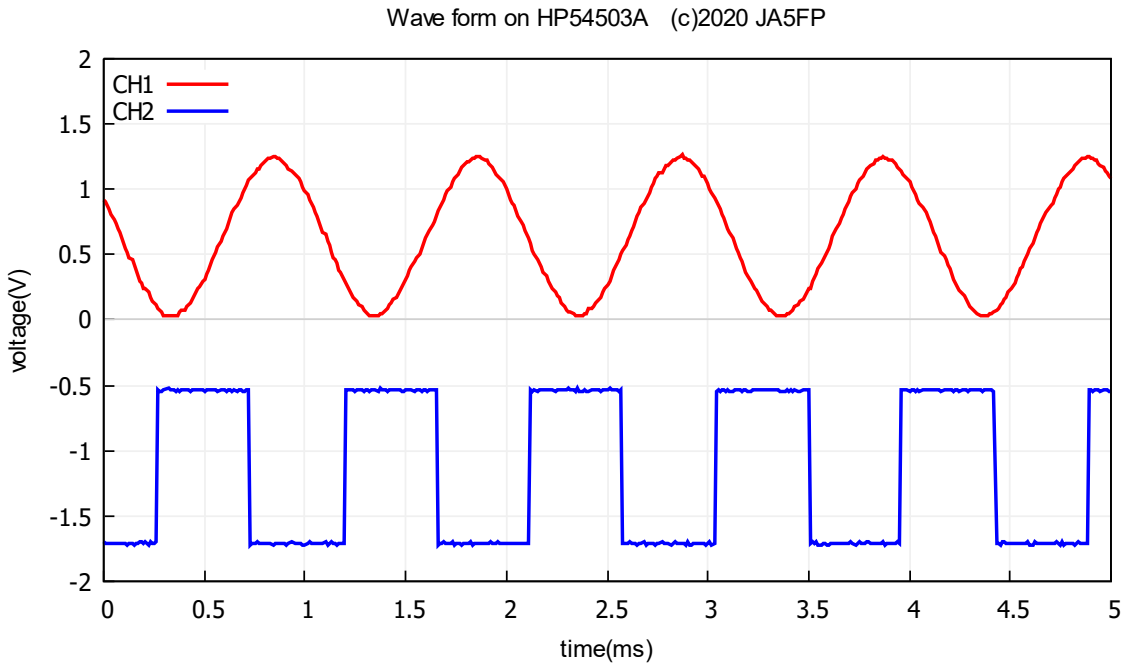

図 3: HP54503A のデータから作成した pdf

5.3 HP8561E Spectrum Analizer の場合

(1) 82357B で GP-IB 接続
(2) HP8561E で観測し、Save TRACE A および TRACE B
(3) PC にインストールしてある python.exe を起動
(4) import visa
(5) rm=visa.ResourceManager()
(6) dev=rm.open_resource('GPIB0::18')
(7) store=open('hp8561e.txt', 'a')
(8) dev.write('TRA?')
(9) line=dev.read()
(10) print('TRA\n'+line)
(11) store.write('TRA\n'+line)
(12) dev.write('TRB?')
(13) line=dev.read()
(14) print(TRB\n +line)
(15) store.write('TRB\n'+line)
(16) store.close()
(17) dev.close() までで'hp8651e.txt' を取得
(18) テキストエディタで、コンマ>>スペース変換、不要個所の削除
(19) Excel または Calc で整形し'hp8651e.dat' と変名
(20) gnuplot.exe を起動
(21) load 'hp8561e.fmt'
(22) print 'hp8561e.pdf'

[参考] hp8651e.fmt の内容例

```
#A parameter file to draw "hp8561.dat" for GNUPLOT (c)2020 JA5FP
set terminal wxt
set title "Spectrum of HP8561E's CAL 300MHz    (c)2020 JA5FP" font "Arial, 11"
set grid back
set xrange [299.5:300.5]
set xtics 299.5, 0.1, 300.5 nomirror
set ytics nomirror
set xlabel "frequency (MHz)"
set ylabel "power (dBm)"
set yrange [-120:0]
set key left
set label 1 "Device:HP8561E" at 300.1, -5.5
plot 'hp8651e.dat' u 2:3 t 'BW=10kHz' w l lw 2,'' u 2:4 t 'BW=100kHz' w l lw 2
```

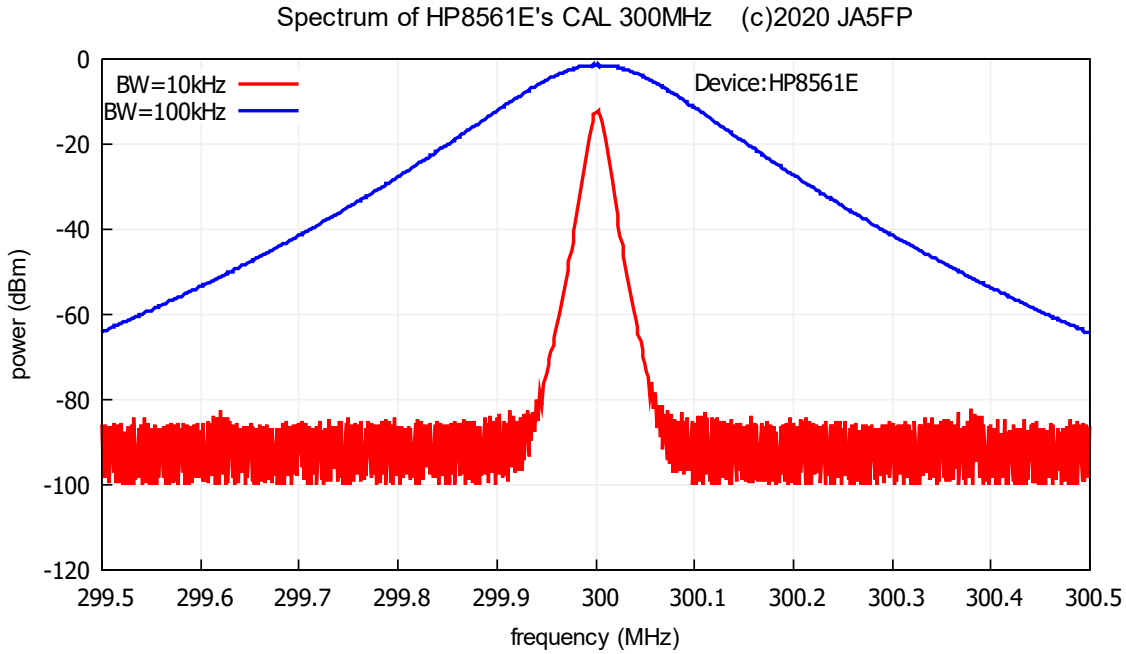

図 4: HP8561E のデータから作成した pdf

5.4 SR620 Counter の場合

(1) 82357B で GP-IB 接続
(2) SR620 で観測
(3) python.exe "sr620.py"で"sr620.txt"にデータ取得
(4) テキストエディタで、コンマ>>スペース変換、不要個所の削除
(5) Excel または Calc で整形し'sr620.dat' と変名
(6) gnuplot.exe を起動
(7) load 'sr620.fmt'
(8) print 'sr620.pdf'

[参考 1] sr620.py の内容例

```
#coding utf-8
#A parameter file to draw "sr620.dat" for GNUPLOT  (c) 2020 JA5FP
import visa
rm=visa.ResourceManager()
rm.list_resources()
dev=rm.open_resource("GPIB0::16")
store=open('sr620.txt', 'a')
xvalue=datetime.datetime.now()
past="{0:%y-%m-%d %H:%M}".format(xvalue)
present=past
while(1)
    while(present==past):
```

```
        xvalue=datetime.datetime.now()
        present="{0:%y-%m-%d %H:%M}".format(xvalue)
        print("waiting")
        time.sleep(5)
    past=present
    dev.write("MEAS?0\n")
    time.sleep(0.1)
    line=dev.read()
    print(present+' '+line)
    store.write(present+' '+line)
store.close()
dev.close()
```

[参考 2] sr620.fmt の内容例

```
#coding utf-8
#A parameter file to draw "sr620.dat" for GNUPLOT  (c) 2020 JA5FP
set terminal wxt
set title "Frequency drift of Audio Oscillator   (c) 2020 JA5FP"
set grid back
set xdata time
set format x "%H:%M"
set timefmt "%H:%M"
set xtics "00:00","00:15", "01:00" nomirror
set xrange ["00:00":"01:00"]
set xlabel "hour:minute (H:M) on 2018/09/30"
set yrange [1:1.7]
set ytics 1, 0.1, 1.7 nomirror
set ylabel "Δ frequency (kHz) from 100kHz"
set key left
plot "sr620.dat" u 2:3 w p t "  MS-9150"
```

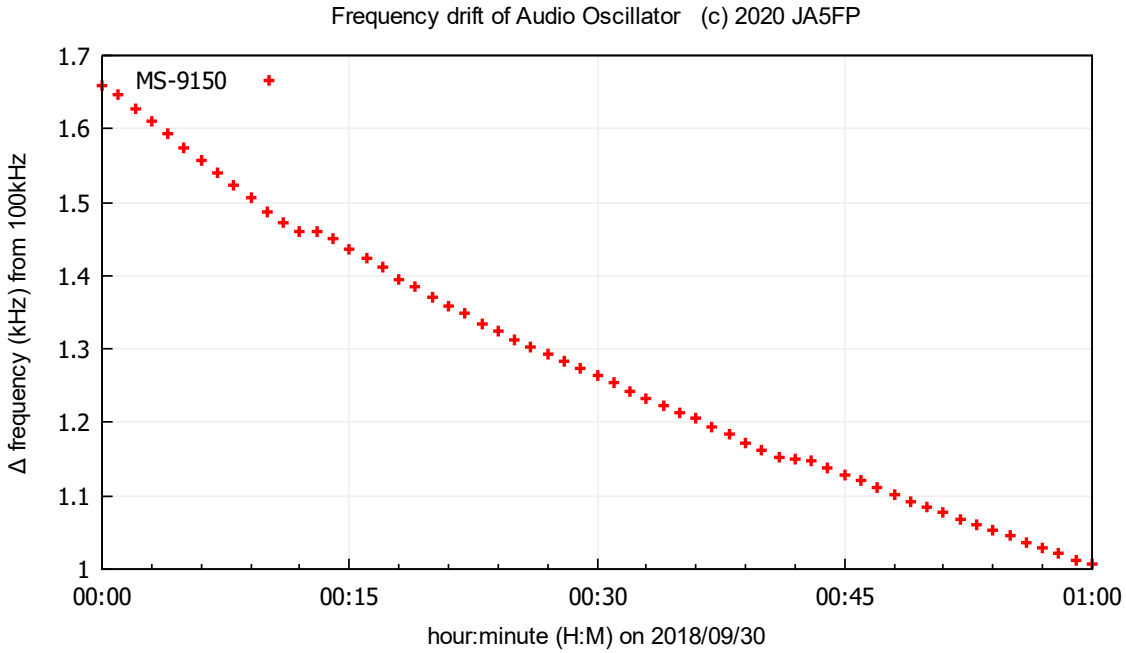

図 5: SR620 のデータから作成した pdf

5.5 TR2114H Multi Meter の場合

(1) 82357B で GP-IB 接続
(2) TR2114H で観測
(3) python.exe "tr2114h.py"で"tr2114h.txt"にデータ取得
(4) テキストエディタで、コンマ>>スペース変換、不要個所の削除
(5) Excel または Calc で整形し'tr2114h.dat' と変名
(6) gnuplot.exe を起動
(7) load 'tr2114h.fmt'
(8) print 'tr2114h.pdf'

[参考 1] tr2114h.py の内容例

```
#coding utf-8
# tr2114h.py   (c) 2020 JA5FP
import visa
import datetime
import time
rm=visa.ResourceManager()
rm.list_resources()
dev=rm.open_resource('GPIB0::20')
store=open("tr2114h.txt", 'a')
xvalue=datetime.datetime.now()
past='{0:%y-%m-%d %H:%M}'.format(xvalue)
present=past
```

```
while(1):
    while(present==past):
        xvalue=datetime.datetime.now()
        present='{0:%y-%m-%d %H:%M}'.format(xvalue)
        time.sleep(15)
    past=present
    line=dev.read()
    print(present+line)
    store.write(present+line+'\n')
store.close()
dev.close()
```

[参考 2] tr2114h.fmt の内容例

```
#coding utf-8
# Gnuplot format file for 'tr2114h.dat' (c) 2020 JA5FP
# Usage: load "tr2114h.fmt"
set title "cooling down characteristic  (c) 2020 JA5FP"
set grid back
set xdata time
set format x "%H:%M"
set timefmt "%H:%M"
set xtics "15:00", "00:30", "19:00" nomirror
set xrange ["15:00":"19:00"]
set xlabel "hour:minute (H:M) on 2020/06/28"
set yrange [20:100]
set ytics 20, 5, 100 nomirror
set ylabel "temprature (°C)
set key left
plot "tr2114h.dat" u 2:3 w l t "TR2114H"
```

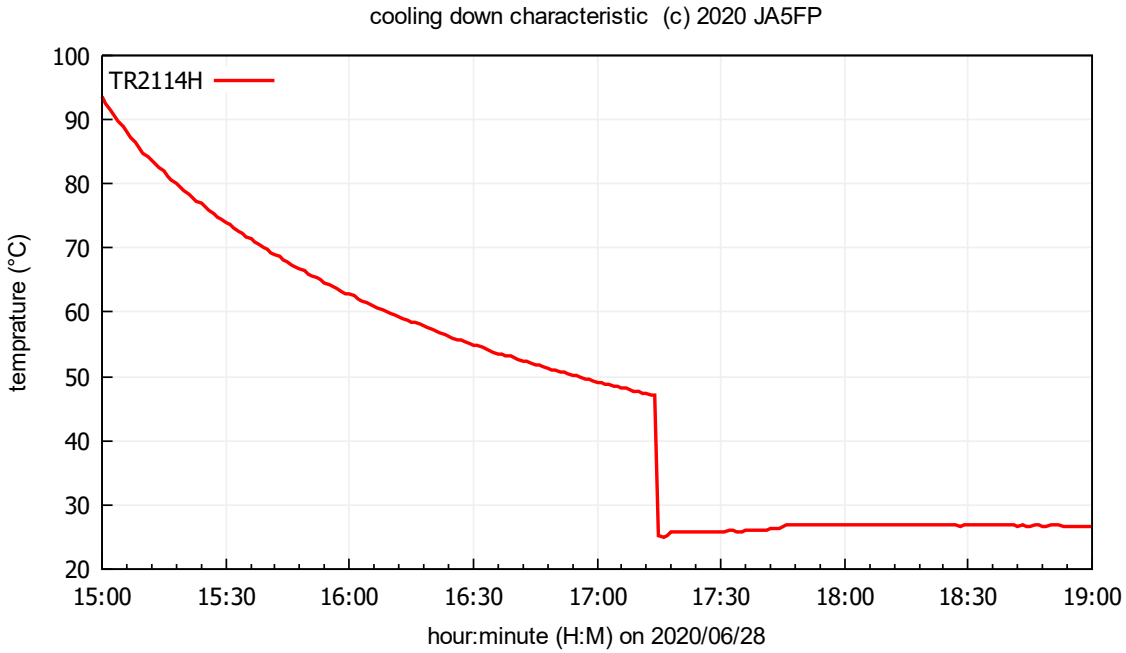

図 6: TR2114H のデータから作成した pdf

(修正:2021.10.6)