

目次

AD Converter の利用	1
設定	1
NuttX Configuration	1
ファームウェアの書き込み	2
動作テスト	2
値の(電圧への)変換	3
値の(温度への)変換	4
プログラミング	5

AD Converter の利用

XG-50 の AD Converter には、外部電源(バッテリー[ch1]・ワイド電源[ch2]) を接続しています。
 加えて SoC 内部の Vref・温度センサーも有効化しています。

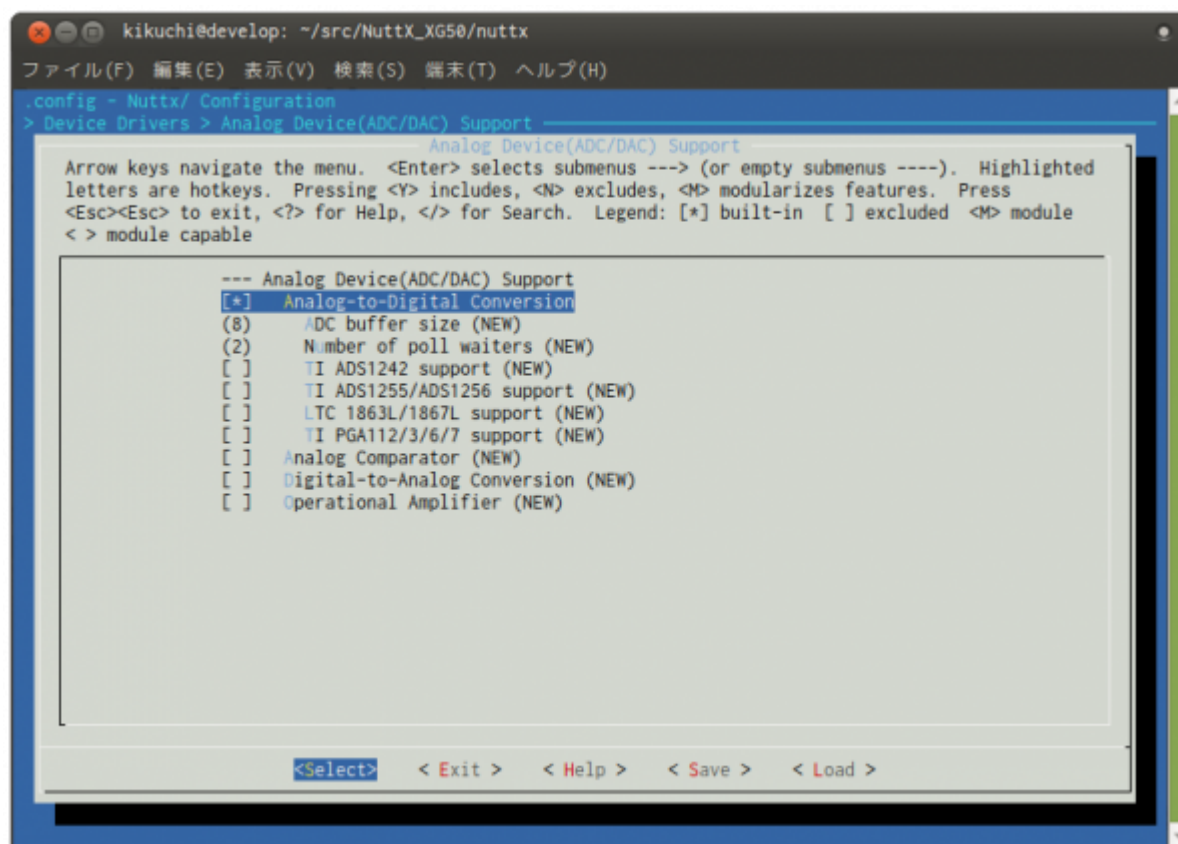
Channel	入力	Note
0	VREFINT	SoC 内部基準電圧源
1	バッテリー電圧	アッテネータ経由 ¹⁾
2	ワイド電源電圧 ²⁾	アッテネータ経由 ³⁾
17	温度センサー	SoC 内蔵温度センサー

設定

NuttX Configuration

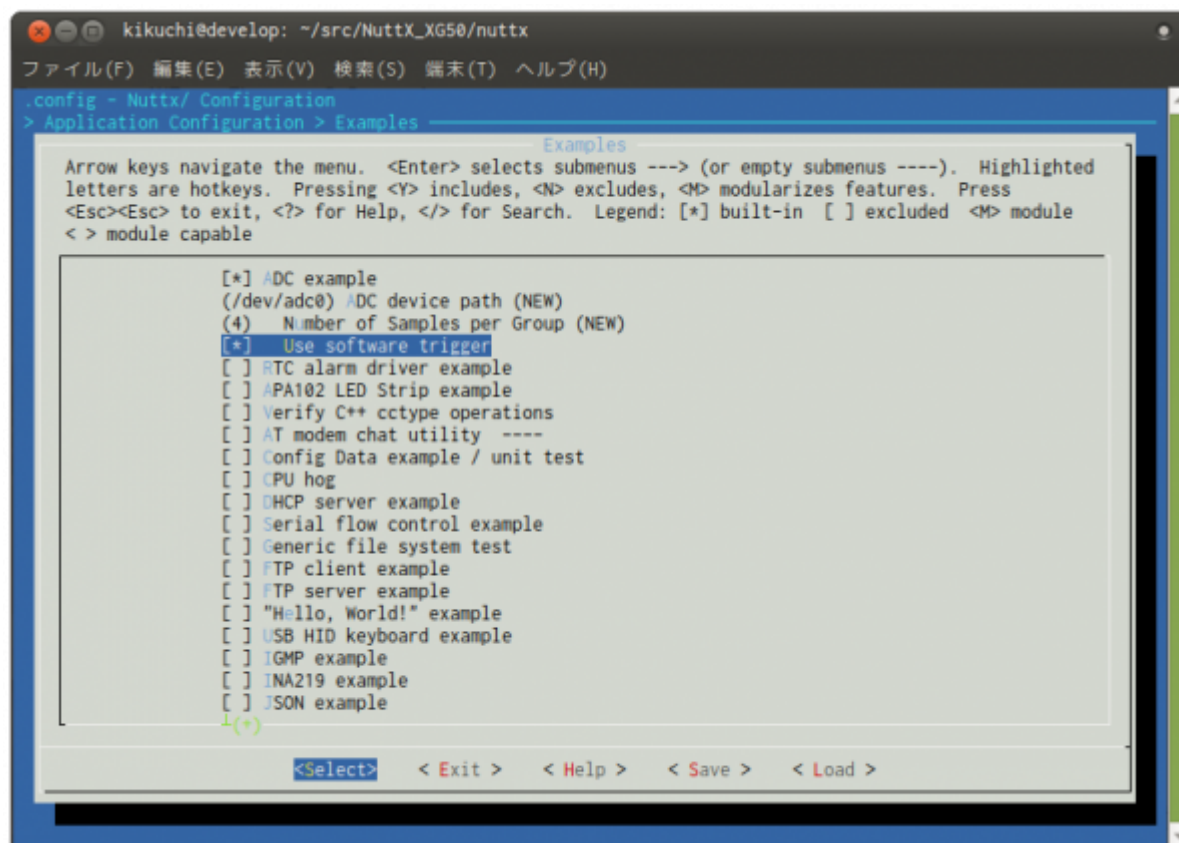
AD Converter を有効化するため、**make menuconfig** で NuttX の構成を変更します。

- Device Drivers → Analog Device(ADC/DAC) Support 有効化
- Device Drivers → Analog Device(ADC/DAC) Support → Analog-to-Digital Conversion 有効化



- Application Configuration → Examples → ADC example 有効化
 - ADC device path → `"/dev/adc0"`
 - Number of Samples per Group → 4

- Use software trigger 有効化



config を保存し、build します。

ファームウェアの書き込み

[ファームウェアの書き込みと動作](#) を参考に、ファームウェアを XG-50 に書き込みます。

動作テスト

書き込んだファームウェアを実行すると、下のように **adc** というコマンドが使用できるようになります。

NuttShell (NSH)

nsh> help

help usage: help [-v] [<cmd>]

[dirname	false	mkfatfs	pwd	time
?	date	free	mkfifo	reboot	true
basename	dd	help	mkrd	rm	uname
break	df	hexdump	mh	rmdir	umount
cat	dmesg	kill	mount	set	unset
cd	echo	ls	mv	sh	usleep

```

cp          exec      mb      mw      sleep      xd
cmp         exit      mkdir   ps      test
Builtin Apps:
  adc <-----
  cu
  i2c
  sudoku
nsh>

```

さっそく実行してみます。

```

nsh> adc
adc_main: g_adcstate.count: 1
adc_main: Hardware initialized. Opening the ADC device: /dev/adc0
Sample:
1: channel: 0 value: 1498
2: channel: 1 value: 3483
3: channel: 2 value: 0
4: channel: 17 value: 936
nsh>

```

Channel 0, 1, 2, 17 の 4 つの値を取ることができました。

値の(電圧への)変換

上記 AD 変換で取得した値は、アナログ電源(VDDA) を基準にした相対値⁴⁾となっています。
STM32L4 の Reference Manual にあるとおり、実際の電圧を求めるには計算を行う必要があります。

計算式は下記のとおりです。

$$V_{\text{CHANNEL_X}} = \frac{V_{\text{DDA}}}{\text{FULL_SCALE}} \times \text{ADC_X_DATA}$$

(Reference Manual より)

ここで VDDA は 3.3[V]、FULL_SCALE は 4095 なのでそれを当てはめて Channel 1 (バッテリー入力電圧) を計算すると、

$$V_{\text{CHANNEL1}} = \frac{3.3}{4095} \times 3483 \times 1.1 = 3.087 \text{ [V]}$$

となります。

VREFINT を利用した計算

STM32L4 の VREFINT を利用し、VDDA の電圧に依存しない計算方法です。

$$V_{\text{CHANNEL_X}} = \frac{3.0 \text{ V} \times \text{VREFINT_CAL} \times \text{ADC_X_DATA}}{\text{VREFINT_DATA} \times \text{FULL_SCALE}}$$

ここで、各変数は下記の通りです。

変数	内容
VREFINT_CAL	VREFINT calibration value ⁵⁾ (0x1FFF75AA - 0x1FFF75AB)
ADC_DATA	AD 変換結果
VREFINT_DATA	Channel 0 AD変換結果
FULL_SCALE	4095

手元のチップでは、**VREFINT_CAL** の値は 0x0678 でした。

```
nsh> mh 0x1fff75aa
1fff75aa = 0x0678
nsh>
```

この値を上記式に当てはめて計算すると、

```
In [8]: ((3 * 0x0678 * 3483) / (1498 * 4095)) * 1.1
Out[8]: 3.1028589034463536
```

となります。

試しに電圧計で電源ピンのところで計測してみると 3.094 [V] でした。
どちらの方法を使用しても 0.3% 程度の誤差⁶⁾で計測できるようです。

値の(温度への)変換

CH17 の温度センサーの値から温度への変換式は下記のとおりです。

$$\text{Temperature (in } ^\circ\text{C)} = \frac{110^\circ\text{C} - 30^\circ\text{C}}{\text{TS_CAL2} - \text{TS_CAL1}} \times (\text{TS_DATA} - \text{TS_CAL1}) + 30^\circ\text{C}$$

ここで、各変数は下記になります。

変数	内容
TS_CAL1	温度センサーキャリブレーション値 ⁷⁾ @30°C (0x1FFF75A8 - 0x1FFF75A9)
TS_CAL2	温度センサーキャリブレーション値 ⁸⁾ @110°C (0x1FFF75CA - 0x1FFF75CB)
TS_DATA	温度センサー ADC 出力値

TS_CAL1, **TS_CAL2** とともに VDDA が 3.0V のときの値となっていますが XG-50 の VDDA は 3.3V のため、

『**VREFINT** を利用した計算』を用いて換算する必要があります。

$$\begin{aligned} \text{Temperature (in } ^\circ\text{C)} &= \frac{110\ ^\circ\text{C} - 30\ ^\circ\text{C}}{\text{TS_CAL2} - \text{TS_CAL1}} \times \\ &\left(\frac{\text{VREFINT_CAL} \times \text{TS_DATA}}{\text{VREFINT_DATA}} - \text{TS_CAL1} \right) + 30\ ^\circ\text{C} \end{aligned}$$

手元のチップで **TS_CAL1**, **TS_CAL2** を確認してみると、

```
nsh> mh 0x1fff75a8
      1fff75a8 = 0x0410
nsh> mh 0x1fff75ca
      1fff75ca = 0x051b
```

それぞれ 0x0410, 0x051b となっているため、それを用いて計算すると、

```
In [18]: VREFINT_CAL=0x0678

In [19]: TS_DATA=936

In [20]: TS_CAL1=0x410

In [21]: TS_CAL2=0x51b

In [22]: VREFINT_DATA=1498

In [23]: ((110 - 30) / (TS_CAL2 - TS_CAL1)) * ((VREFINT_CAL * TS_DATA) /
VREFINT_DATA - TS_CAL1) + 30
Out[23]: 28.419065620577733
```

28.4 となりました。

プログラミング

apps/examples/adc/adc_main.c を参照してください。

¹⁾

10/11 にしているため、真の値を求めるには 1.1 倍する必要があります

²⁾

5V 36V

³⁾

1/12 にしているため、真の値を求めるには 12倍 する必要があります

⁴⁾

12bit ADC なので、0 ~ 4095

⁵⁾

メーカーにて出荷時にキャリブレーションした結果を不揮発領域に書き込んであります

⁶⁾

電圧計が正しいかどうかは置いておいて

7) , 8)

VDDA 3.0V

From:

<https://ma-tech.centurysys.jp/> - **MA-X/MA-S/MA-E/IP-K Developers' Wiki**

Permanent link:

https://ma-tech.centurysys.jp/doku.php?id=xg_series_devel:use_adc:start

Last update: **2019/01/07 16:16**