



SUZAKU I/O Board  
LED/SW SIL00-U00  
Software Manual

Version 1.0.0

2006 年 8 月 11 日

株式会社アットマークテクノ

<http://www.atmark-techno.com/>

**SUZAKU** 公式サイト

<http://suzaku.atmark-techno.com/>

# 目次

1.	はじめに.....	1
1.1.	マニュアルについて.....	1
1.2.	フォントについて.....	1
1.3.	コマンド入力例の表記について.....	1
1.4.	数字の表記について.....	1
1.5.	謝辞.....	2
1.6.	注意事項.....	2
2.	作業の前に.....	3
2.1.	準備するもの.....	3
2.2.	接続方法.....	4
2.3.	開発環境の構築について.....	4
2.3.1.	FPGA データ.....	4
2.4.	インターフェースの表記について.....	5
3.	アプリケーションプログラム.....	7
3.1.	単色 LED.....	7
3.1.1.	概要.....	7
3.1.2.	単色 LED の状態表現.....	7
3.1.3.	使用法.....	8
3.2.	7セグメント LED.....	8
3.2.1.	概要.....	8
3.2.2.	7セグメント LED の状態表現.....	8
3.2.3.	使用法.....	9
3.3.	押しボタンスイッチ.....	10
3.3.1.	概要.....	10
3.3.2.	使用法.....	10
3.4.	ロータリコードスイッチ.....	10
3.4.1.	概要.....	10
3.4.2.	使用法.....	11
3.5.	シリアルポート.....	11
3.5.1.	概要.....	11
3.5.2.	tip.....	11
3.6.	アプリケーションの選択.....	12
4.	デバイスドライバ.....	13
4.1.	単色 LED.....	13
4.1.1.	概要.....	13
4.1.2.	システムコール.....	13
4.2.	7セグメント LED.....	15
4.2.1.	概要.....	15
4.2.2.	システムコール.....	15
4.3.	押しボタンスイッチ.....	16
4.3.1.	概要.....	16
4.3.2.	押しボタンスイッチの状態表現.....	17
4.3.3.	システムコール.....	17
4.4.	ロータリコードスイッチ.....	18
4.4.1.	概要.....	18
4.4.2.	システムコール.....	18
4.5.	シリアルポート.....	19
4.5.1.	概要.....	19

---

4.6. ドライバの選択.....	20
5. ハードウェア設定 .....	21
5.1. 7セグLED .....	21
5.2. 単色LED .....	21
5.3. スイッチ.....	21
5.4. ロータリコードスイッチ.....	21
6. 参考文献.....	22

## 表目次

表 1-1	使用しているフォント .....	1
表 1-2	表示プロンプトと実行環境の関係 .....	1
表 1-3	表示方法と基数の関係 .....	1
表 2-1	各種インターフェースの表記 .....	5
表 2-2	各セグメントの表記.....	<b>エラー! ブックマークが定義されていません。</b>
表 3-1	単色 LED のビットマップ .....	7
表 3-2	demo_led の使用法.....	8
表 3-3	セグメントのビットマップ (7セグメント LED) .....	8
表 3-4	7セグメント LED のビットマップ .....	9
表 3-5	demo-7seg の使用法.....	9
表 3-6	demo-sw の使用法.....	10
表 3-7	demo-rsw の使用法 .....	11
表 4-1	led デバイスドライバ.....	13
表 4-2	open システムコール (単色 LED) .....	13
表 4-3	close システムコール (単色 LED) .....	14
表 4-4	read システムコール (単色 LED) .....	14
表 4-5	write システムコール (単色 LED) .....	14
表 4-6	7セグメント led デバイスドライバ .....	15
表 4-7	open システムコール (7セグメント LED) .....	15
表 4-8	close システムコール (7セグメント LED) .....	15
表 4-9	read システムコール (7セグメント LED) .....	16
表 4-10	write システムコール (7セグメント LED) .....	16
表 4-11	押しボタンスイッチデバイスドライバ .....	16
表 4-12	押しボタンスイッチのビットマップ .....	17
表 4-13	open システムコール (押しボタンスイッチ) .....	17
表 4-14	close システムコール (押しボタンスイッチ) .....	17
表 4-15	read システムコール (押しボタンスイッチ) .....	18
表 4-16	ロータリコードスイッチデバイスドライバ.....	18
表 4-17	open システムコール (ロータリコードスイッチ) .....	18
表 4-18	close システムコール (ロータリコードスイッチ) .....	19
表 4-19	read システムコール (ロータリコードスイッチ) .....	19
表 4-20	シリアル通信設定 .....	19

## 目次

---

図 2-1 SIL00-U00 接続例 .....	4
図 2-2 FPGA データの書き換え .....	5
図 2-3 各種インターフェースの配置 .....	5
図 2-4 7セグメントLEDのセグメント .....	6
図 3-1 demo-ledの実行例 .....	8
図 3-2 demo_7segの実行例 .....	9
図 3-3 demo-swの実行例 .....	10
図 3-4 demo-rswの実行例 .....	11
図 3-5 tip コマンドの実行例 .....	11
図 3-6 アプリケーションの選択例 (menuconfig) .....	12
図 4-1 ドライバの選択例 (menuconfig) .....	20

# 1.はじめに

## 1.1. マニュアルについて

このマニュアルは、SUZAKU I/O Board の LED/SW (SIL00-U00) に付属している以下のサンプルソフトウェアについて記載されています。

- アプリケーションプログラム
- Linux 用デバイスドライバ

ソフトウェアのカスタマイズなどの開発作業を行う際には、参考文献[1][2]もあわせてご覧ください。SUZAKU の機能を最大限に引き出すために、ご活用いただければ幸いです。

## 1.2. フォントについて

このマニュアルでは以下のようにフォントを使っています。

表 1-1 使用しているフォント

フォント例	説明
本文中のフォント	本文
[PC ~]\$ ls	プロンプトとユーザ入力文字列

## 1.3. コマンド入力例の表記について

このマニュアルに記載されているコマンドの入力例は、表示されているプロンプトによって、それぞれに対応した実行環境を想定して書かれています。「/」の部分はカレントディレクトリによって異なります。各ユーザのホームディレクトリは「~」で表わします。

表 1-2 表示プロンプトと実行環境の関係

プロンプト	コマンドの実行環境
[PC /]#	作業用 PC 上の特権ユーザで実行
[PC /]\$	作業用 PC 上の一般ユーザで実行
[SUZAKU /]#	SUZAKU 上の特権ユーザで実行
[SUZAKU /]\$	SUZAKU 上の一般ユーザで実行

## 1.4. 数字の表記について

このマニュアルに記載されている数字は、特に明記されている場合を除き表記方法によって異なった基数を表します。

表 1-3 表示方法と基数の関係

ベース	記載方法	備考
2 進数	10100101b	数字の後に”b”と記載
10 進数	165	0 から 9 までのアラビア数字
16 進数	0xA5	”0x”と数字の順に記載

## 1.5. 謝辞

SUZKAU で使用しているソフトウェアは Free Software / Open Source Software で構成されています。Free Software / Open Source Software は世界中の多くの開発者の成果によって成り立っています。この場を借りて感謝の意を示します。

## 1.6. 注意事項

本製品に含まれるソフトウェア(付属のドキュメント等も含みます)は、現状のまま(AS IS)提供されるものであり、特定の目的に適合することや、その信頼性、正確性を保証するものではありません。また、本製品の使用による結果についてもなんら保証するものではありません。

## 2.作業の前に

---

### 2.1.準備するもの

SIL00-U00 を使用する前に、以下のものを準備してください。

- SUZAKU  
SUZAKU-S シリーズまたは SUZAKU-V シリーズのいずれかの SUZAKU ボードです。
- 作業用 PC  
Linux もしくは Windows が動作し、1 ポート以上のシリアルポートを持つ PC です。
- シリアルクロスケーブル  
D-Sub9 ピン (メス - メス) の「クロス接続用」ケーブルです。
- D-Sub 9 ピン-10 ピン変換ケーブル  
D-Sub9 ピンと SUZAKU のピンヘッダ(10 ピン)を接続するケーブルです。
- 付属 CD-ROM (以降、付属 CD)  
SUZAKU に関する各種マニュアルやソースコードが収納されています。
- シリアルコンソールソフト  
minicom や Tera Term などのシリアルコンソールソフトです。作業用 PC にインストールしてください。
- 電源  
AC アダプタ 5V を使用してください。



## 2.2. 接続方法

下の図を参照して、シリアルクロスケーブル、電源、そして LAN ケーブルを SUZAKU に接続してください。

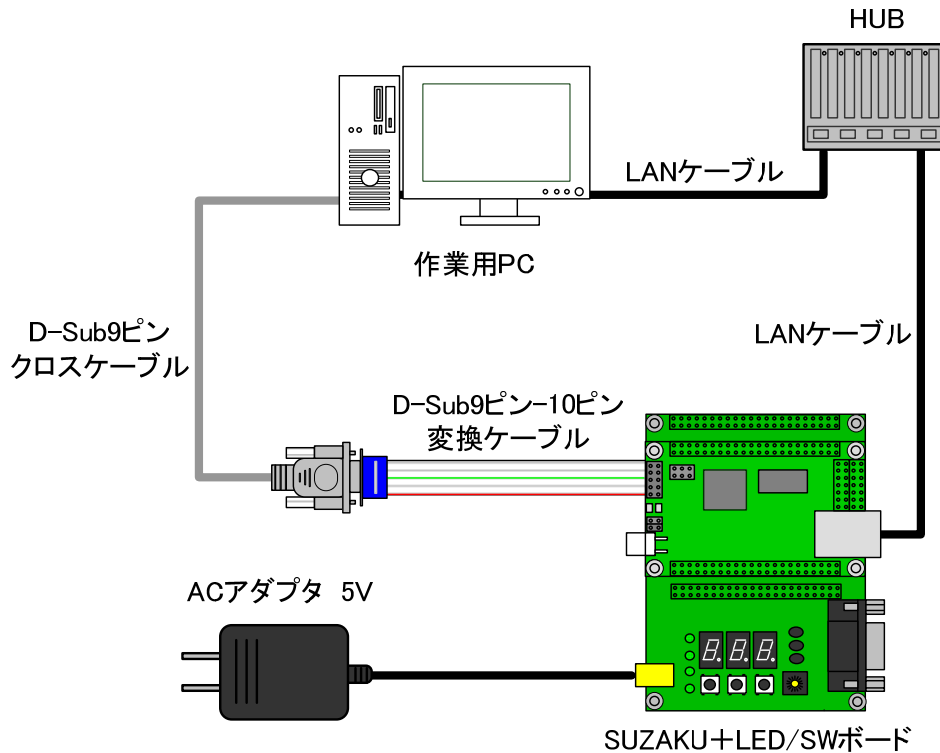


図 2-1 SIL00-U00 接続例



### 注意

SUZAKU ボードからは絶対に電源を供給しないでください。電源がショートし、機器を破損する可能性があります。SUZAKU ボードへ電源は、LED/SW ボードから供給されるようになっています。

## 2.3. 開発環境の構築について

SUZAKU でのクロス開発環境の構築については、参考文献[1][2]を参照してください。

### 2.3.1. FPGA データ

SUZAKU-S スターターキットとして SIL00-U00 を購入された場合、SUZAKU のフラッシュメモリに入っている FPGA のデータは 参考文献[3]用になっています。このため本書で説明しているアプリケーションおよびデバイスドライバを使う前に、フラッシュメモリ内の FPGA コンフィギュレーションデータを変更する必要があります。変更する FPGA データは、付属 CD の `suzaku-starter-kit¥image` ディレクトリに、`fpga-szXXX-sil-gpio_control.bin` という名のファイルです。szXXX の部分は、お使いの SUZAKU ボードの型番です。

FPGAのデータは以下のようにhermitコマンドで書き換えることが可能です。hermitの詳細については、参考文献[1]をご覧ください。

```
[PC ~/]$ hermit download -r fpga -i fpga-szXXX-sil-gpio_control.bin --force-locked
```

図 2-2 FPGA データの書き換え



### 注意

FPGA リージョンを異なったデータで書き換えたり、書き換えが異常終了した場合はSUZAKU および SIL00-U-00 の電源をいれないでください。最悪の場合、SUZAKU および SIL00-U00 を破壊する恐れがあります。書き換えに失敗した場合は、お使いのSUZAKU用ハードウェアマニュアルを参照し正しいデータに書き戻してから電源を入れてください。

## 2.4. インターフェースの表記について

SIL00-U00 の各種インターフェースの配置と、本マニュアルでの表記は以下のとおりです。

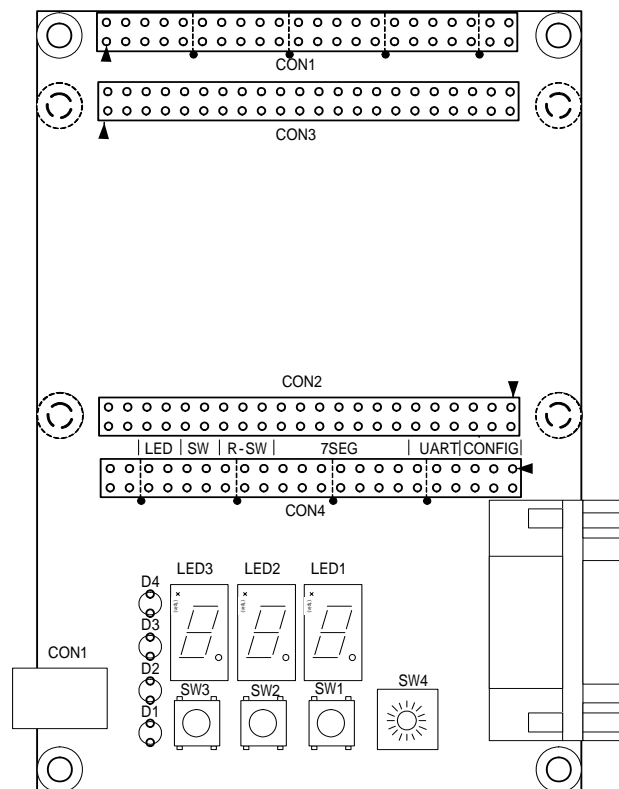


図 2-3 各種インターフェースの配置

表 2-1 各種インターフェースの表記

インターフェース	表記
単色 LED	D1, D2, D3, D4
7セグメント LED	LED1, LED2, LED3

押しボタンスイッチ	SW1, SW2, SW3
ロータリコードスイッチ	SW4

また、7セグメントLEDの各セグメントの位置と表記を下図に示します。

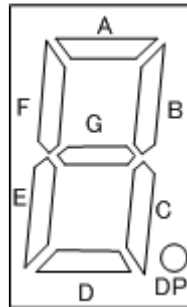


図 2-4 7セグメントLEDのセグメント

## 3. アプリケーションプログラム

この章では SIL00-U00 用に用意されているアプリケーションプログラムを説明します。すべてのアプリケーションは C 言語で記述されています。またソースコード自体の見通しが良くなるように、それぞれ単機能に特化した作りになっています。

SIL00-U00 に搭載している各インターフェースによって、以下のようなアプリケーションプログラムが用意されています。

- 単色 LED: demo-led
- 7 セグメント LED: demo-7seg
- 押しボタンスイッチ: demo-sw
- ロータリコードスイッチ: demo-rsw

なお、これらのアプリケーションプログラムは、次章で説明するデバイスドライバを使用します。実行するには、デバイスドライバ(「[4.6. ドライバの選択](#)」参照)およびアプリケーション(「[3.6. アプリケーションの選択](#)」参照)を選択後、イメージを作成し、Flashメモリの書き換えを行なってください。

### 3.1. 単色 LED

#### 3.1.1. 概要

この章では単色 LED を制御するアプリケーションプログラムについて説明します。単色 LED は SIL00-U00 の左側に一列に並んでいます。この単色 LED を操作するアプリケーションプログラムの名前は demo-led です。demo-led は /bin にインストールされています。このアプリケーションプログラムは後述するオプションの組み合わせによって任意の単色 LED の操作や、状態を表示することが可能です。

たとえば demo-led に引数として 0 を与えることで、すべての単色 LED を消灯することができます。逆に引数として f を与えることで、すべての単色 LED を点灯させることができます。

#### 3.1.2. 単色 LED の状態表現

demo-led では SIL00-U00 に搭載されている 4 つの単色 LED の状態を数字で表します。各単色 LED の点灯を 1、消灯を 0 とするバイナリ表現に置き換えると、下記の図のように 4 bit ですべての単色 LED の状態を表すことができます。

上記の例のように 0x0(0000b)はすべての単色 LED が消灯している状態を、0xf(1111b)ですべての単色 LED が点灯している状態を表わします。また、D1 と D3 のみ点灯される状態は 0x5(0101b)と表現します。

表 3-1 単色 LED のビットマップ

3	2	1	0
D4	D3	D2	D1

### 3.1.3. 使用法

demo\_led の使用法について説明します。

表 3-2 demo\_led の使用法

書式	demo-led [options] [ctrl]
説明	プログラム demo-led は、単色 LED を制御するアプリケーションプログラムです。引数 ctrl は制御コードで、単色 LED の状態表現で指定します。オプションの指定により任意の LED 毎の制御が行えます。ctrl の指定は任意で、未指定時は、全 LED の状態を出力します。
オプション	-l <u>N</u> 対象の LED を指定します。N は、1(D1) ~ 4(D4)の数字を指定します。未指定時は、全 LED が対象です。 -u                使用法を表示します。 -v                バージョンを表示します。

# demo-led 0	(全 LED を消灯)
# demo-led f	(全 LED を点灯)
# demo-led f	(全 LED の状態を表示)
# demo-led -l 2 0	(LED2 を消灯)
# demo-led -l 4 1	(LED4 を点灯)
# demo-led l 4 1	(LED4 の状態を表示)

図 3-1 demo-led の実行例

## 3.2.7 セグメント LED

### 3.2.1. 概要

SIL00-U00 の中央に位置する 3 つの 7 セグメント LED を制御するアプリケーションプログラムについて説明します。このアプリケーションプログラムの名前は demo-7seg です。demo-7seg は他のアプリケーションプログラム同様 /bin にインストールされています。

demo-7seg を引数無しで実行することで、すべての 7 セグメント LED の状態を表示することができます。

### 3.2.2. 7 セグメント LED の状態表現

demo-7seg では SIL00-U00 に搭載されている 7 セグメント LED の状態を数字で表します。各 7 セグメント LED には名前が示す通り 7 つのセグメント LED が搭載されており、これらのセグメントには図 2-4 が示す通り A から G の名前が与えられています。また、小数点を表す 8 つめのセグメントは DP と呼ばれます。

これら 8 つのセグメントを demo-7seg では下表のようにバイナリで表現しています。1 がセグメント LED の点灯を、0 がセグメント LED の消灯を表します。

表 3-3 セグメントのビットマップ(7 セグメント LED)

7	6	5	4	3	2	1	0
DP	G	F	E	D	C	B	A

例えば 2 という数字を表現する場合を考えます。2 は、A, B, G, E, D の 5 つのセグメント LED の点灯で表現することができます。その際の状態は 0x5b(0101 1011b)と表します。

上記のように一つの 7 セグメント LED を表すには 8bit 必要になります。SIL00-U00 に搭載されている 7 セグメント LED は 3 つありますので、すべての 7 セグメント LED を表現するために 24 bit 使用します。

**表 3-4 7 セグメント LED のビットマップ**

31 ~ 24	23 ~ 16	15 ~ 8	7 ~ 0
---------	---------	--------	-------

(空)    LED3    LED2    LED1

### 3.2.3. 使用法

**表 3-5 demo-7seg の使用法**

書式	demo-7seg [options] [ctrl]
説明	プログラムdemo-7segは、7 セグメントLEDを制御するアプリケーションプログラムです。引数 <u>ctrl</u> は制御コードで、LEDの状態を 16 進数で指定します (3.2.2. 7 セグメントLEDの状態表現を参照)。オプションの指定により任意のLEDやセグメント毎の制御も行えます。
オプション	-l <u>N</u> 制御する LED の番号を指定します。 <u>N</u> には 1 (LED1) ~ 3 (LED3)を指定できます。未指定時は、全 LED が対象です。 -u                使用法を表示します。 -v                バージョンを表示します。

# demo-7seg	(全 LED の状態出力)
4f666d	
# demo-7seg 4f66fd	(LED3: 「3」点灯, LED2: 「4」点灯, LED1: 「5」点灯)
# demo-7seg -l 1 6d	(LED1: 「5」点灯)

**図 3-2 demo\_7seg の実行例**

### 3.3. 押しボタンスイッチ

#### 3.3.1. 概要

押しボタンスイッチの押下状態を確認するサンプルアプリケーション demo-sw について説明します。demo-sw は SIL00-U00 の手前側に 3 つ並んでいる押しボタンスイッチの状態を標準出力に表示します。後述するオプションを使用することで、指定したスイッチの状態のみ表示することも可能です。

#### 3.3.2. 使用法

demo-sw の使用法について説明します。

表 3-6 demo-sw の使用法

書式	demo-sw [options]
説明	プログラム demo-sw は、押しボタンスイッチの押下状態を出力するアプリケーションプログラムです。オプションによりボタン毎の状態を出力することができます。
オプション	<p>-l <u>N</u>      監視するスイッチの番号を指定します。<u>N</u>には 1 (SW1) ~ 3 (SW3)を指定できます。未指定時は、全スイッチが対象です。</p> <p>-u          使用法を表示します。</p> <p>-v          バージョンを表示します。</p>

```
# demo-sw
SW3:[ ] SW2:[ ] SW1:[ ]
SW3:[ ] SW2:[ ] SW1:[ ]
SW3:[ ] SW2:[*] SW1:[ ]      SW2 が押された
:
```

図 3-3 demo-sw の実行例

demo-sw を停止する場合は、Ctrl + C キーで実行を中断してください。

### 3.4. ロータリコードスイッチ

#### 3.4.1. 概要

ロータリコードスイッチの状態を表示するサンプルアプリケーション demo-rsw について説明します。ロータリコードスイッチはつまみを回すことで状態を変化させるスイッチのことで、SIL00-U00 は 押しボタンスイッチの右側に 16 の状態を表現できるロータリコードスイッチを搭載しています。

demo-rsw は 16 の状態を一定間隔で標準出力に表示します。

### 3.4.2. 使用法

demo-rsw の使用法について説明します。

表 3-7 demo-rsw の使用法

書式	demo-rsw [options]
説明	プログラム demo-rsw は、ロータリコードスイッチの状態を表示するアプリケーションプログラムです。スイッチが示す位置を標準出力に書き出します。
オプション	-u                    使用法を表示します。 -v                    バージョンを表示します。

```
# demo-rsw
0
0
4
:
```

図 3-4 demo-rsw の実行例

demo-rsw を停止する場合は、Ctrl + C キーで実行を中断してください。

## 3.5. シリアルポート

### 3.5.1. 概要

シリアルポート用の特別なアプリケーションは用意されていません。既存の UNIX コマンド等で利用する例を説明します。

SIL00-U00 のシリアルポートは、一般的なシリアルポートと同じですので、一般的な Linux のシリアルポート用アプリケーションプログラムが動作します。ここでは、tip と呼ばれるシリアル通信ツールを使って説明します。

### 3.5.2. tip

tip は、シンプルなシリアル通信プログラムです。ここではそれぞれのシリアルポートに別のシリアル通信ソフトウェアが接続されていると仮定します。tip を起動するには使用するシリアルポートや転送レートなどいくつかの通信設定が必要です。指定値については、「表 4-20 シリアル通信設定」を参照してください。

以下に、シリアルポートに ttyS1、転送レート 115200bps を指定した、tip コマンドの実行例を示します。

```
[SUZAKU ~/]# tip -s 115200 -l /dev/ttyS1
```

図 3-5 tip コマンドの実行例

tip コマンドの終了は、「~.」と入力します。



## 3.6. アプリケーションの選択

本章で紹介した各種アプリケーションは、全て uClinux-dist の user ディレクトリに含まれています。ご利用の際には、make menuconfig で、アプリケーションを追加する必要があります。

```
[PC ~/uClinux-dist]$ make menuconfig
Main Menu
  Kernel/Library/Defaults Selection --->
    [*] Customize Vendor/User Settings

Main Menu
  Miscellaneous Applications --->
    --- SUZAKU I/O Board LED/SW Sample Application
    [*] demo-led
    [ ] demo-7seg
    [ ] demo-sw
    [ ] demo-rsw
```

図 3-6 アプリケーションの選択例 (menuconfig)

## 4. デバイスドライバ

この章では SIL00-U00 に実装されている各種インターフェースを制御するための Linux 用デバイスドライバについて説明します。

### 4.1. 単色 LED

#### 4.1.1. 概要

SIL00-U00 には、単色 LED ( 緑 ) が 4 個実装されています。ここで説明するデバイスドライバは、4 個の単色 LED を個別または全て同時に、制御 ( 点灯、消灯 ) することができます。

表 4-1 led デバイスドライバ

ドライバ ID	sil-led
ドライバ名	SUZAKU I/O Borad -LED/SW- LED
デバイスファイル名	/dev/silled (全部)
	/dev/silled1 (D1)
	/dev/silled2 (D2)
	/dev/silled3 (D3)
	/dev/silled4 (D4)
ソースファイル所在	linux-2.4.x/drivers/char/sil-led.c

#### 4.1.2. システムコール

本ドライバで用意されているシステムコールは、open・close・read・write の 4 種類です。それぞれについて説明します。

##### open

表 4-2 open システムコール(単色 LED)

書式	int open(const char *pathname, int flags);
説明	デバイスをオープンします。オープンに成功した場合、新しいファイルディスクリプタを返します。
引数	pathname オープンするデバイスファイル名 flags ファイルアクセスモード O_RDONLY, O_WRONLY, O_RDWR のどれかひとつを指定します。それぞれ読み込み専用、書き込み専用、読み書き用にファイルをオープンすることを要求します。
戻り値	成功した場合は新しいファイルディスクリプタを返し、エラーが発生した場合は-1 を返します。

close

表 4-3 close システムコール(単色 LED)

書式	int close(int fd);
説明	デバイスをクローズします。
引数	fd           ファイルディスクリプタ
返り値	成功した場合は 0 を返し、エラーが発生した場合は-1 を返します。

read

表 4-4 read システムコール(単色 LED)

書式	ssize_t read(int fd, void *buf, size_t count);
説明	デバイスからデータを読み込みます。最大 <u>count</u> バイトをバッファ <u>buf</u> へ読み込みます。 読み込みデータは、readシステムコールを呼び出した時点の単色LEDの状態を示します。単色LEDの状態については、 <a href="#">3.1.2. 単色LEDの状態表現</a> を参照してください。
引数	fd           ファイルディスクリプタ buf          読み込みデータを格納するバッファ count       読み込みデータのバイト数
返り値	成功した場合は読み込んだバイト数を返し、エラーが発生した場合は-1 を返します。

write

表 4-5 write システムコール(単色 LED)

書式	int write(int fd, const void *buf, size_t count);
説明	デバイスへデータを書き込みます。バッファ <u>buf</u> から最大 <u>count</u> バイト分のデータをデバイスへ書き込みます。 書き込みデータには、制御したい単色LEDの状態を示す文字を指定します。単色LEDの状態は、 <a href="#">3.1.2. 単色LEDの状態表現</a> を参照してください。
引数	fd           ファイルディスクリプタ buf          書き出しデータを格納するバッファ count       書き出しデータのバイト数
返り値	成功した場合は書き込んだバイト数を返し、エラーが発生した場合は-1 を返します。

## 4.2.7 セグメント LED

### 4.2.1. 概要

SIL00-U00 には、7 セグメント LED が 3 個実装されています。ここで説明するドライバは、3 個の 7 セグメント LED を個別にまたは全部同時に、制御（点灯、消灯）することができます。

表 4-6 7 セグメント led デバイスドライバ

ドライバ ID	sil-7seg
ドライバ名	SUZAKU I/O Board -LED/SW- 7SEG
デバイスファイル名	/dev/sil7seg (全部)
	/dev/sil7seg1 (LED1)
	/dev/sil7seg2 (LED2)
	/dev/sil7seg3 (LED3)
ソースファイル所在	linux-2.4.x/drivers/char/sil-7seg.c

### 4.2.2. システムコール

本ドライバで用意されているシステムコールは、open・close・read・write の 4 種類です。それぞれについて説明します。

#### open

表 4-7 open システムコール(7 セグメント LED)

書式	int open(const char *pathname, int flags);
説明	デバイスをオープンします。オープンに成功した場合、新しいファイルディスクリプタを返します。
引数	pathname オープンするデバイスファイル名 flags ファイルアクセスモード O_RDONLY, O_WRONLY, O_RDWR のどれかひとつを指定します。それぞれ読み込み専用、書き込み専用、読み書き用にファイルをオープンすることを要求します。
返り値	成功した場合は新しいファイルディスクリプタを返し、エラーが発生した場合は-1 を返します。

#### close

表 4-8 close システムコール(7 セグメント LED)

書式	int close(int fd);
説明	デバイスをクローズします。
引数	fd ファイルディスクリプタ
返り値	成功した場合は 0 を返し、エラーが発生した場合は-1 を返します。

**read****表 4-9 read システムコール(7 セグメントLED)**

書式	ssize_t read(int fd, void *buf, size_t count);
説明	デバイスからデータを読み込みます。最大 count バイトをバッファ buf へ読み込みます。 読み込みデータは、readシステムコールを呼び出した時点の 7 セグメントLEDの状態を示します。状態については、「 <a href="#">3.2.2. 7 セグメントLEDの状態表現</a> 」を参照してください。
引数	fd           ファイルディスクリプタ buf           読み込みデータを格納するバッファ count        読み込みデータのバイト数
返り値	成功した場合は読み込んだバイト数を返し、エラーが発生した場合は-1 を返します。

**write****表 4-10 write システムコール(7 セグメントLED)**

書式	int write(int fd, const void *buf, size_t count);
説明	デバイスへデータを書き込みます。バッファ buf から最大 count バイト分のデータをデバイスへ書き込みます。 書き込みデータには、対象の 7 セグメントLEDの制御状態を示す文字を指定します。状態については、「 <a href="#">3.2.2. 7 セグメントLEDの状態表現</a> 」を参照してください。
引数	fd           ファイルディスクリプタ buf           書き出しデータを格納するバッファ count        書き出しデータのバイト数
返り値	成功した場合は書き込んだバイト数を返し、エラーが発生した場合は-1 を返します。

## 4.3. 押しボタンスイッチ

### 4.3.1. 概要

SIL00-U00 には、押しボタンスイッチが 3 個実装されています。ここで説明するドライバは、3 個の押しボタンスイッチの状態を取得することができます。

**表 4-11 押しボタンスイッチデバイスドライバ**

ドライバ ID	sil-sw
ドライバ名	SUZAKU I/O Board -LED/SW- SW
デバイスファイル名	/dev/silsw (全部)
	/dev/silsw1 (SW1)
	/dev/silsw2 (SW2)
	/dev/silsw3 (SW3)
ソースファイル所在	linux-2.4.x/drivers/char/sil-sw.c

### 4.3.2. 押しボタンスイッチの状態表現

本デバイスドライバでは、押しボタンスイッチの状態を次のように表現します。1つの押しボタンスイッチの状態として、押下と開放の2つを定義し、それぞれを1と0で表します。

SIL00-U00には、3つの押しボタンスイッチが搭載されており、各スイッチの状態を下表に示す並びで整列させることで、3ビットの2進数とみなすことができます。この2進数を16進数に変換した値を、全スイッチの状態として使用します。

表 4-12 押しボタンスイッチのビットマップ

2	1	0
SW3	SW2	SW1

例えば、SW1とSW3が押下、その他を開放とした全スイッチの状態は、「5」と表現されます。

### 4.3.3. システムコール

本ドライバで用意されているシステムコールは、open・close・readの3種類です。それぞれについて説明します。

#### open

表 4-13 open システムコール(押しボタンスイッチ)

書式	int open(const char *pathname, int flags);
説明	デバイスをオープンします。オープンに成功した場合、新しいファイルディスクリプタを返します。
引数	pathname オープンするデバイスファイル名 flags ファイルアクセスモード O_RDONLY を指定します。読み込み専用でファイルをオープンすることを要求します。
返り値	成功した場合は新しいファイルディスクリプタを返し、エラーが発生した場合は-1を返します。

#### close

表 4-14 close システムコール(押しボタンスイッチ)

書式	int close(int fd);
説明	デバイスをクローズします。
引数	fd ファイルディスクリプタ
返り値	成功した場合は0を返し、エラーが発生した場合は-1を返します。

**read****表 4-15 read システムコール(押しボタンスイッチ)**

書式	ssize_t read(int fd, void *buf, size_t count);
説明	デバイスからデータを読み込みます。最大 count バイトをバッファ buf へ読み込みます。 読み込みデータは、readシステムコールを呼び出した時点の押しボタンスイッチの状態を示します。押しボタンスイッチの状態は、「 <a href="#">4.3.2. 押しボタンスイッチの状態表現</a> 」を参照してください。
引数	fd           ファイルディスクリプタ buf           読み込みデータを格納するバッファ count        読み込みデータのバイト数
返り値	成功した場合は読み込んだバイト数を返し、エラーが発生した場合は-1 を返します。

**4.4. ロータリコードスイッチ****4.4.1. 概要**

SIL00-U00 には、ロータリコードスイッチが 1 個実装されています。ここで説明するドライバは、ロータリコードスイッチの状態を取得することができます。

**表 4-16 ロータリコードスイッチデバイスドライバ**

ドライバ ID	sil-rsw
ドライバ名	SUZAKU I/O Board -LED/SW- RSW
デバイスファイル名	/dev/silrsw
ソースファイル所在	linux-2.4.x/drivers/char/sil-rsw.c

**4.4.2. システムコール**

本ドライバで用意されているシステムコールは、open・close・read の 3 種類です。

**open****表 4-17 open システムコール(ロータリコードスイッチ)**

書式	int open(const char *pathname, int flags);
説明	デバイスをオープンします。オープンに成功した場合、新しいファイルディスクリプタを返します。
引数	pathname    オープンするデバイスファイル名 flags        ファイルアクセスモード O_RDONLY を指定します。読み込み専用でファイルをオープンすることを要求します。
返り値	成功した場合は新しいファイルディスクリプタを返し、エラーが発生した場合は-1 を返します。

**close****表 4-18 close システムコール(ロータリコードスイッチ)**

書式	int close(int fd);
説明	デバイスをクローズします。
引数	fd           ファイルディスクリプタ
返り値	成功した場合は 0 を返し、エラーが発生した場合は-1 を返します。

**read****表 4-19 read システムコール(ロータリコードスイッチ)**

書式	ssize_t read(int fd, void *buf, size_t count);
説明	デバイスからデータを読み込みます。最大 count バイトをバッファ buf へ読み込みます。 読み込みデータは、read システムコールを呼び出した時点のロータリコードスイッチの状態を示します。ロータリコードスイッチの状態は、0~15 の 16 段階を 16 進数で表した文字として表現されます。例えば、状態 0 は '0'、状態 10 は'a'となります。
引数	fd           ファイルディスクリプタ buf          読み込みデータを格納するバッファ count       読み込みデータのバイト数
返り値	成功した場合は読み込んだバイト数を返し、エラーが発生した場合は-1 を返します。

## 4.5. シリアルポート

### 4.5.1. 概要

uartlite 用のデバイスドライバを使用しています。シリアル通信の各種設定値は、下表のようになっています。

**表 4-20 シリアル通信設定**

項目	設定値
転送レート	115200bps
データ長	8bit
ストップビット	1bit
パリティ	なし
フロー制御	なし
デバイスファイル名	/dev/ttyS1



## 4.6. ドライバの選択

この章で紹介した各種デバイスドライバは、全て付属 CD の uClinux-dist に含まれています。ご利用の際には、使用したいデバイスドライバを make menuconfig で追加し、再度ビルドする必要があります。

```
[PC ~/uClinux-dist]$ make menuconfig
Main Menu
  Kernel/Library/Defaults Selection  --->
    [*] Customize Kernel Settings

Main Menu
  Character devices  --->
    [*] SUZAKU I/O Board LED/SW
    [ ] Led Support
    [*] 7 segment led support
    [ ] Switch support
    [ ] Rotaly code switch support
    [ ] RS232C support
```

図 4-1 ドライバの選択例 (menuconfig)

## 5. ハードウェア設定

---

このマニュアルに記載されているアプリケーションプログラム、およびデバイスドライバは以下の GPIO コアの設定で確認されています。

なお、BASEADDR は SUZAKU-S と SUZAKU-V シリーズによって異なります。SUZAKU-S シリーズでは 0xFF000000、SUZAKU-V シリーズは 0xF0000000 となります。

### 5.1.7 セグ LED

```
PARAMETER HW_VER = 3.01.b
PARAMETER C_GPIO_WIDTH = 24
PARAMETER C_IS_BIDIR = 0
PARAMETER C_BASEADDR = BASEADDR + 0xFFD000
PARAMETER C_HIGHADDR = BASEADDR + 0xFFD1FF
```

### 5.2. 単色 LED

```
PARAMETER HW_VER = 3.01.b
PARAMETER C_GPIO_WIDTH = 4
PARAMETER C_IS_BIDIR = 0
PARAMETER C_BASEADDR = BASEADDR + 0xFFD200
PARAMETER C_HIGHADDR = BASEADDR + 0xFFD3FF
```

### 5.3. スイッチ

```
PARAMETER HW_VER = 3.01.b
PARAMETER C_GPIO_WIDTH = 3
PARAMETER C_IS_BIDIR = 0
PARAMETER C_ALL_INPUTS = 1
PARAMETER C_BASEADDR = BASEADDR + 0xFFD400
PARAMETER C_HIGHADDR = BASEADDR + 0xFFD5FF
PARAMETER C_INTERRUPT_PRESENT = 1
```

### 5.4. ロータリコードスイッチ

```
PARAMETER HW_VER = 3.01.b
PARAMETER C_GPIO_WIDTH = 4
PARAMETER C_IS_BIDIR = 0
PARAMETER C_ALL_INPUTS = 1
PARAMETER C_BASEADDR = BASEADDR + 0xFFD600
PARAMETER C_HIGHADDR = BASEADDR + 0xFFD7FF
PARAMETER C_INTERRUPT_PRESENT = 1
```

## 6. 参考文献

---

1. 『uClinux-dist Developers Guide』, (株)アットマークテクノ
2. 『SUZAKU Software Manual』, (株)アットマークテクノ
3. 『SUZAKU スターターキットガイド (FPGA 開発編)』, (株)アットマークテクノ
4. 『OPB General Purpose Input/Output (GPIO) Datasheet』, Xilinx

改訂履歴

Ver	年月日	改訂内容
1.0.0	2006/08/11	・初版発行

SUZAKU I/O Board -LED/SW- Software Manual

2006年8月11日 version 1.0.0

---

株式会社アットマークテクノ

060-0035 札幌市中央区北5条東2丁目 AFTビル6F

TEL:011-207-6550 FAX:011-207-6570

---