

# Armadillo-640 開発基礎セミナー

#### 第6部 拡張I/Fを使った開発の流れ(その1) (ハードウェア、カーネル編)

株式会社アットマークテクノ

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





■ 第1部 Armadilloについて ■ 第2部 Armadilloが動作する仕組み 第3部 Armadilloを動かしてみる 第4部 開発環境の構築 第5部 アプリケーションの作成 ■ 第6部 拡張I/Fを使った開発の流れ その1) ハードウェア、カーネル編 その2) アプリケーション編 第7部 イメージの作成 第8部 イメージの書き込み ■ 第9部 製品運用に向けて ■ 第10部 参考情報



# Armadilloにおける開発の要素

ハードウェア

 ・ 拡張I/Fを使った拡張基板 → GPIO接続のSWとLED
 I2C接続の加速度センサー

- カーネル/デバイスツリー
  - デバイスドライバの適用 → I2Cドライバの有効化確認
  - 拡張I/Fの端子機能の設定 → I2C端子の有効化
- アプリケーション
  - 既存のパッケージのインストール → C言語ビルド環境 Pythonパッケージ
  - 独自のアプリケーションの作成
    - → SWからデータを取得してLEDへ出力するプログラム 照度センサーからデータを取得するプログラム



# この章の概要

ハードウェア

 ・ 拡張I/Fを使った拡張基板 → GPIO接続のSWとLED
 I2C接続の加速度センサー

- カーネル/デバイスツリー
  - デバイスドライバの適用 → I2Cドライバの有効化確認
  - 拡張I/Fの端子機能の設定 → **I2C端子の有効化**
- アプリケーション
  - 既存のパッケージのインストール → C言語ビルド環境 Pythonパッケージ
  - 独自のアプリケーションの作成
    - → SWからデータを取得してLEDへ出力するプログラム 照度センサーからデータを取得するプログラム



# ハードウェア トグルスイッチとLEDの接続 I2C接続の加速度センサーとの接続



# Armadillo-640の拡張I/F

■ Armadilloの拡張I/Fは様々な機能を持っています。



CON9 CON14

CON9:GPIO、I2C、SPI、UART、SD等 CON14:GPIO、I2C、UART等

- 各機能のドライバはカーネルコンフィギュレーションで適用します。
- 各機能の端子設定はデバイスツリーで行います。



### 端子の確認方法

# Armadillo-640マルチプレクス表(※)で端子を確認します。 ※)Webサイトよりダウンロードできます。

Armadillo Armad	dilloとは 組み込みIoTボード IoTゲートウェイ DX向けシリーズ サポート	
ホーム   Armadillo-640		
	Armadillo-640	
	製品機要 製品仕様 製品ラインアップ 技術情報・ダウンロード	
	ドキュメントダウンロード	
ドキュメント マニュアル・カイド (ネス仕様書) アンテナリスト 変更履歴表 明品デークジート 回路図・形状図	コンテンツ Armadillo-640 製品マニュアル	
信頼性試験報告書 RoHS規制物資不使用証明書 マルチブレクス表(ピン配置)▼ キーワード	Armadillo標準ガイド〜組み込みLinuxの導入から製品化まで〜 Armadilloシリーズで組み込み機器の関発を行なうエンジニアに向けた開発ガイドブック	
絞り込み	Armadillo-640 マルチプレクス表	
	Armadillo-640 納入仕様書 Armadillo-640本体の納入仕様書	



# **GPIOに使用する端子**

#### Armadillo-640マルチプレクス表では、下記のような表記です。

CON9\_13ピン: GPIO3\_IO23 CON9\_28ピン: GPIO4\_IO09

						GPIU	
部品番号	ピン番号	信号名	ピン名	電圧グループ	own -	GPIO	T
_	12	PWRON	PWRON	VDD_SNVS_IN			ĺ
CON9	13	GPI03_I023	LCD_DATA18	VCC_3.3V		GPI03 I023	t
	14	GPI03_I024	LCD_DATA19	VCC_3.3V		GPI03 I024	
	15	GPI03_I025	LCD_DATA20	VCC_3.3V		GPI03 I025	t
	16	GPI03_I026	LCD_DATA21	VCC_3.3V		GPI03 I026	t
	17	GPI03_I027	LCD_DATA22	VCC_3.3V		GPI03 I027	t
	18	GPI03_I028	LCD_DATA23	VCC_3.3V		GPI03 1028	Ť
	19	GND					t
	20	VCC_3.3V					t
	21	USB2_DN	USB_OTG2_DN	-			t
	22	USB2_DP	USB_OTG2_DP	-			t
	23	USB2_VBUS	USB_OTG2_VBUS	USB_VBUS			t
	24	EXT_USB_EN	-	VCC_3.3V			Ť
	25	GPI04_I006	NAND_DATA04	VCC_3.3V		GPI04_I006	Ť
	26	GPI04_I007	NAND_DATA05	VCC_3.3V		GPI04_I007	Ť
	27	GP104_1008	NAND DATARS	VCC 3 3V		CP104 1009	Ť.
	28	GPI04_I009	NAND_DATA07	VCC_3.3V		GPI04_1009	Ť
	27 28	GP104_1008 GP104_1009	NAND_DATA06 NAND_DATA07	VCC_3_3V VCC_3_3V		GP104_1009 GP104_1009	



**I2C2** 

# I2Cに使用する端子2

#### Armadillo-640マルチプレクス表では、下記のような表記です。

CON9\_4ピン: I2C2\_SDA CON9\_6ピン: I2C2\_SCL

					<b>/チブレク</b>	ス機能
部品番号	ピン番号	信号名	ピン名	電圧グループ	1202	12
	4	GPI01 I021	UART2 RX C	VCC 3.3V	1262	120
	1	GPI01_I022	UART2_CTS	VCC_3.3V		
	2	GPI01_I023	UART2_RTS	VCC_3.3V		
.	3	GPI01_1017	HART1 RX C	VCC_3.3V		12C3
	4	GPI01_I031	UART5_RX_C	VCC_3.3V	I2C2_SDA	
CON9	5	GPI01_I016	UART1_TX_C	VCC_3.3V		12C3
	6	GPI01 I030	UART5 TX C	VCC 3.3V	I2C2 SCL	
	7	VCC_3.3V				
	8	VCC_3.3V				
	9	GND				
	10	GND				



### Armadillo-640 トグルスイッチとLEDの接続

#### ■ Armadillo-640とトグルSWとLEDを下記のように接続します。





### Armadillo-640と 照度センサー(TSL2561)の接続

Armadillo-640と照度センサー(TSL2561)搭載モジュール を下記のように接続することになります。





# カーネル/デバイスツリー I2Cドライバの設定



# カーネルのカスタマイズについて

- Armadillo-640のカーネルをカスタマイズする場合は、ATDE上 に展開したカーネル(第4章で実施)のソースコードで行います。
- ATDEで、カーネルのソースコードのディレクトリに移動します。

atmark@atde7:~\$ cd linux-v4.14-at5

カーネルの設定を「デフォルト」にするには、下記のコマンドを実行します。

atmark@atde7:~/linux-v4.14-at5\$ make ARCH=arm armadillo-640\_defconfig

このコマンドは、初めてカーネルをビルドする場合や、意図的にデフォルトに戻す場合に実行します。 カスタマイズ後は、このコマンドは実行する必要はありません。



カーネルコンフィギュレーション I2Cドライバの設定1

- ここでは、カーネルコンフィギュレーションの用法として、 I2Cドライバの設定を確認してみます。
- なお、Armadillo-640のカーネルコンフィギュレーションの デフォルトでは、I2Cドライバは有効になっています。 よって、ここでは、設定変更は行わず、設定確認のみを行います。





#### 下記コマンド(make menuconfig)を実行すると、

atmark@atde7:~/linux-v4.14-at5\$ make ARCH=arm menuconfig

#### 下記のようなカーネルコンフィギュレーションの画面が開きます。

🔏 atde7-i386 - VMware Workstation 14 Player	<u>م</u>	
Player(₽) ▼      ▼ ⊕ 🔒 🖾 🖾		<b>«</b>
アクティビティ □ 端末 ▼      金曜日 18:26		± 40 € -
atmark@atde7: ~/linux-v4.14-at3		×
ファイル(F) 編集(E) 表示(V) 検索(S) 端末(T) ヘルプ(H) .config - Linux/arm 4.14-at3 Kernel Configuration		
Linux/arm 4.14-at3 Kernel Configuration Arrow keys navigate the menu. <enter> selects submenus submenus). Highlighted letters are hotkeys. Pressing includes, <n> excludes, <m> modularizes features. Press <e exit, <? > for Help,  for Search. Legend: [*] built-in Ceneral setup&gt; [] Enable loadable module support [*] Enable the block layer&gt; System Type&gt; Bus support&gt; Boot options&gt; CPU Power Management&gt; Floating point emulation&gt; Userspace binary formats&gt; (+) <pre> Kernel </pre></e </m></n></enter>	> (or emp ) <y> Sc&gt;<esc> [ ] ad &gt;</esc></y>	ty to

### 

#### Menuconfigでの操作方法

- 上下キー: メニューの選択
- 左右キー:動作の選択
- スペースキー:オプションの選択
- Enterキー:動作の決定
- yキー:\*を付ける。(選択中の設定を有効にする)
- nキー:\*を外す。(選択中の設定を無効にする)



#### Device Driversにカーソルを合わせて<Select>でEnterします。

🔏 atde7-i386 - VMware Workstation 14 Player 🦳 🗆	×
Player(P) 🕶 📕 🖛 🖨 🖾 🛱 🦄 🗰	
アクティビティ 回 端末 ▼ 金曜日 18:27 晶 ♥	) () <del>-</del>
atmark@atde7: ~/linux-v4.14-at3	×
ファイル(F) 編集(E) 表示(V) 検索(S) 端末(T) ヘルプ(H)	
.config - Linux/arm 4.14-at3 Kernel Configuration	_ []
Arrow keys navigate the menu. <enter> selects submenus&gt; (or empty submenus&gt;). Highlighted letters are hotkeys. Pressing <y> includes, <n> excludes, <m> modularizes features. Press <esc> to exit, <? > for Help,  for Search. Legend: [*] built-in [] Power management options&gt; [*] Networking support&gt; Perimeare Drivers&gt; Firmware Drivers&gt; File systems&gt; Kernel hacking&gt; Security options&gt; [] Virtualization [] Virtualization</esc></m></n></y></enter>	



#### ■ I2C supportにカーソルを合わせて<Select>でEnterします。

📽 atde7-i386 - VMware Workstation 14 Player 🛛 🗌	×
Player(₽) ▼ 📕 ▼ 🖶 🖾 🖾 🔯	« 📔 📰
アクティビティ □ 端末 ▼ 金曜日 18:28 鳥	●) () <del>、</del>
atmark@atde7: ~/linux-v4.14-at3	×
ファイル(F) 編集(E) 表示(V) 検索(S) 端末(T) ヘルプ(H)	
.config - Linux/arm 4.14-at3 Kernel Configuration	1
Arrow keys navigate the menu. <enter> selects submenus&gt; (or empty submenus&gt;). Highlighted letters are hotkeys. Pressing <y> includes, <n> excludes, <m> modularizes features. Press <esc><esc> to exit, <? > for Help,  for Search. Legend: [*] built-in [] [] Generic Target Core Mod (TCM) and ConfigFS Infrastructure [*] Network device support&gt; [] Open-Channel SSD target support&gt; Input device support&gt; Character devices&gt; ISPI support&gt; [] SPMI support [] HSI support [] PPS support [] PPS support</esc></esc></m></n></y></enter>	





#### ■ I2C support内の設定が[\*](有効)になっていることを確認します。

🔏 atde7-i386 - VMware Workstation 14 Player 🦳 🗌	×
Player( <u>P</u> ) 🕶 📕 💌 🛱 🔂 🧮 🔯	«   📄
アクティビティ ▷ 端末 ▼ 金曜日 18:29 4	●0 O -
atmark@atde7: ~/linux-v4.14-at3	×
ファイル(F) 編集(E) 表示(V) 検索(S) 端末(T) ヘルプ(H)	
.config - Linux/arm 4.14-at3 Kernel Configuration	1
<pre>submenus). Highlighted letters are hotkeys. Pressing <y> includes, <n> excludes, <m> modularizes features. Press <esc>Esc&gt; to exit, <?> for Help,  for Search. Legend: [*] built-in []</esc></m></n></y></pre>	



カーネルコンフィギュレーション I2Cドライバの設定6

- カーネルコンフィギュレーションが終了するまで、
   <Exit>でEnterします。
- ここでは、設定変更しませんでしたが、設定変更した場合は、 下記のように、設定保存の確認があります。





#### カーネルコンフィギュレーションの設定は、製品マニュアル等では、 下記のように表記されます。

#### UARTの場合

Device Drivers ---> Character devices ---> [\*] Enable TTY <TTY> Serial drivers ---> [\*] IMX serial port support <SERIAL\_IMX> [\*] Console on IMX serial port <SERIAL\_IMX\_CONSOLE>



# カーネル/デバイスツリー I2C端子の設定



# デバイスツリーの設定 (at-dtweb)

Armadillo-640 製品マニュアル

#### 20.3. Device Treeをカスタマイズする

https://manual.atmark-techno.com/armadillo-640/armadillo-640\_product\_manual\_ja-1.9.0/ch20.html#sect.customize-dts に記載されているat-dtwebをATDE上にインストールすることで、 容易に端子設定できます。

- 基本的な機能は、at-dtwebで端子設定できますが、 GPIOなどを独自に設定する場合は、ソースコードの設定が必要な 場合があります。
- 本セミナーでは、at-dtwebを使用してデバイスツリーを設定する 別の方法を説明します。



# デバイスツリーの設定 使用する端子にI2Cの機能を選択1

- Armadillo-640の拡張I/Fの端子は、複数の機能がマルチプレクス されています。
- 端子をどの機能で使用するかを設定するのが、デバイスツリーです。

Armadille	o-640 マル	チブレクス表							12C2	-	
部品番号	ビン番号	信号名	ピン名	電圧グループ	FCSP11	EC:	SPI FCSP13	FCSP14	ソルチブレク 12C2	12C 12C 12C3	IX6ULLの信号 12C4
1	1	VCC_3.3V									
CON14	3	GPI01_I020	UART2_TX_DATA	VCC_3.3V			ECSP13_SS0				I2C4_SCL
	4	GPI01_I021	UART2_RX_DATA	VCC_3.3V			ECSP13_SCLK				I2C4_SDA
	1	GPI01_I022	UART2_CTS_B	VCC_3.3V			ECSPI3_MOSI				
CON9	2	GPI01_I023	UART2_RTS_B	VCC_3.3V		2	ECSPI3_MISO				3
	3	GPI01_I017	UART1_RX_DATA	VCC_3.3V		8			12 17	I2C3_SDA	
	4	GP101_1031	UAR15_RX_DATA	VCC_3.3V		ECSP12_MIS0			12C2_SDA		
	5	GPI01_I016	UART1_TX_DATA	VCC_3.3V						I2C3_SCL	
	6	GPI01 I030	UART5 TX DATA	VCC 3.3V		ECSPI2 MOSI			12C2 SCL		
	7	VCC_3.3V									
	8	VCC_3.3V									
	9	GND									
	10	GND									



# at-dtwebの起動

■ 下記コマンドを実行して、

ATDE7 に at-dtweb パッケージをインストールします。

atmark@atde7:~\$ sudo apt-get update
atmark@atde7:~\$ sudo apt-get install at-dtweb

#### アクティビティをクリックし、「at-dtweb」と入力してatdtwebのアイコンをクリックして起動します。







# at-dtwebの起動

#### ■ ボードの選択画面が表示されるので、Armadillo-640を選択します。



コンフィギュレーション済みの Linux カーネルディレクトリを選択して、「OK」をクリックします。

Cancel	Root directory of kernel source		<b>Q</b> 0
3 最近開いたファイル			
<b>企</b> ホーム	名前	▼ サイズ	更新日時
	COPYING	18.7 kB	2月17日
\$ 9000-F	CREDITS	98.6 kB	2月17日
ドキュメント	Documentation		2月17日
■ ビデオ	M Kbuild	2.3 kB	2月17日
- C73	🛅 Kconfig	287 バイト	2月17日
日 音楽	MAINTAINERS	430.2 kB	2月17日
o 画像	Makefile	61.5 kB	2月17日
Ed bridge	README	722 /371	2月17日
司 フロッピー ディー	i arch		2月17日
	block		2月17日
+ 他の場所	🛅 certs		2月17日
	🛅 crypto		2月17日
	trivers		2月17日
	🛅 firmware		2月17日
	🛅 fs		2月17日
	🛅 include		2月17日
	init.		2月17日



# デバイスツリーの設定 (at-dtwebの画面)



※コンソール出力用



# デバイスツリーの設定 (at-dtwebの画面)





「Device Tree build!」が表示されたらビルド完了



# デバイスツリーの設定 使用する端子にI2Cの機能を選択2

- ビルドが終了すると、arch/arm/boot/dts/以下に DTS/DTB が作成されています。
- arch/arm/boot/dts/armadillo-640-expansion-interface.dtsi
- arch/arm/boot/dts/armadillo-640-at-dtweb.dtb



# カスタマイズしたデバイスツリー

#### Armadillo-640への書き込み



# Armadillo-640に カーネル/デバイスツリーを書き込む1

#### ■ これまでの手順で生成したデバイスツリー

arch/arm/boot/dts/armadillo-640-at-dtweb.dtb

をArmadillo-640に書き込むことにって、I2C端子

CON9\_4ピン:I2C2\_SDA CON9\_6ピン:I2C2\_SCL が有効になります。



### Armadillo-640に デバイスツリーを書き込む2

#### ■ カスタマイズしたDTBイメージをArmadillo上に用意しておきます。

root@armadillo:~# **Is** armadillo-640-at-dtweb.dtb

eMMCの第2パーティションを/mnt/ディレクトリにマウントします。

root@armadillo:~# mount /dev/mmcblk0p2 /mnt

 DTB を a640.dtb にリネームして/mnt/boot/ディレクトリにコ ピーします。

root@armadillo:~# cp armadillo-640-at-dtweb.dtb /mnt/boot/a640.dtb

■ マウントした eMMC の第 2 パーティションをアンマウントします。

root@armadillo:~# umount /mnt



# I2Cが有効になっているかを確認1

#### ■ I2Cが有効になると

/dev/i2c-1 (末尾の-1は、I2Cのポートによって変わります。) のようにデバイスファイルとして見えるようになります。

現時点では、I2Cを有効にしたカーネルイメージ/デバイスツリー で起動していないので、/dev/i2c-1は見えません。

root@armadillo:~# ls /dev/i2c\* /dev/i2c-4



# I2Cが有効になっているかを確認2

- I2Cを有効にしたカーネルイメージ/デバイスツリーで再起動します。
- haltコマンドを実行して、System haltedのログがでるまで待ちます。

root@armadillo:~# halt (略) [ 891.779603] reboot: System halted

- Armadillo-640の電源をOFFにします。
- Armadillo-640の電源をONにしてログインします。

=>というプロンプトが出るところでは、 boot と入力してEnterを押します。 ログイン名、パスワード共に、root でログインします。

■ I2Cのデバイスファイルが存在することを確認します。

root@armadillo:~# ls /dev/i2c\*
/dev/i2c-1

※I2C2のデバイスファイル名は「i2c-1」になります。



# Armadillo-640とSW/LED、 照度センサーの接続



### Armadillo-640と SW/LED、照度センサーの接続

- カーネルとデバイスツリーでI2C端子を有効化したことにより、 I2C接続の照度センサー(TSL2561)と接続可能になりました。
- haltコマンドを実行して、System haltedのログがでるまで待ちます。

root@armadillo:<sup>~</sup># halt (略) [ 891.779603] reboot: System halted

Armadillo-640の電源をOFFにします。



### Armadillo-640 トグルスイッチとLEDの接続

#### ■ Armadillo-640とトグルSWとLEDを下記のように接続します。





### Armadillo-640と 照度センサー(TSL2561)の接続

Armadillo-640と照度センサー(TSL2561) 搭載モジュール を下記のように接続します。





### Armadillo-640と SW/LED、照度センサーの接続

- Armadillo-640の電源をONにしてログインします。 =>というプロンプトが出るところでは、boot と入力してEnterを押します。 ログイン名、パスワード共に、root でログインします。
- 以上で、Armadillo-640と照度センサー(TSL2561)が I2Cで通信できるようになりました。