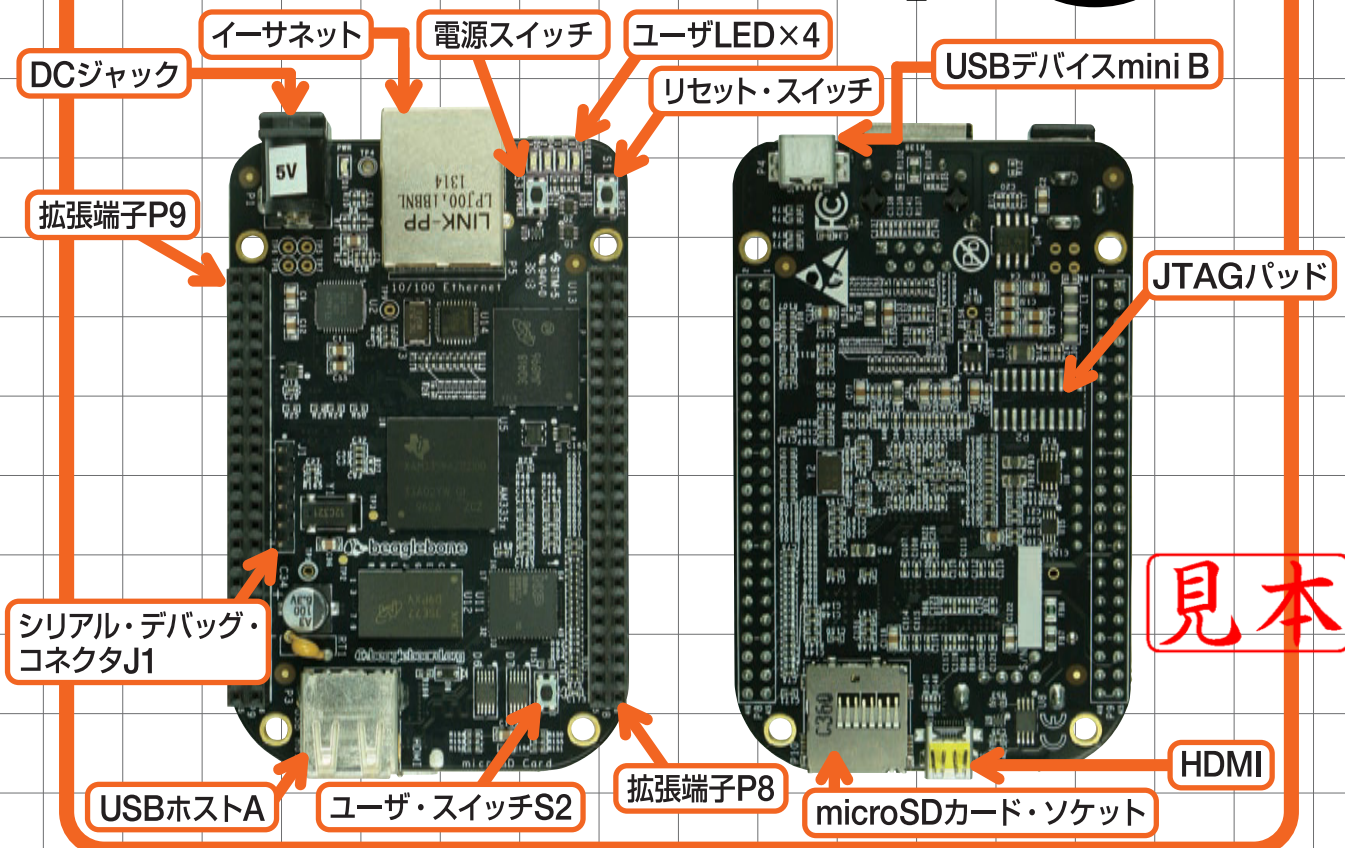
Interface増刊
インターフェース SPECIAL

ハードウェア制御の大本命！ Androidにも挑戦！

Linuxガジェット BeagleBone Black でI/O



CQ出版社

第3章

カーネルのデバイス・ファイルを直接制御

LED 点灯 / 消灯アプリケーションの製作

出村 成和

Androidが動作するようになったところで，次に，このBeagleBone Blackとrowboatならではの処理を行ってみます．

ここでは，BeagleBone Blackの拡張端子にあるGPIO（General Purpose Input/Output）にLEDを接続し，アプリケーションからLEDの点灯と消灯ができるようにします（**図1**）．

このようなGPIOの制御は，Androidのアプリケーション開発キットであるAndroid SDKではサポートされていません．そこで，アプリケーションから，Linuxカーネルのデバイス・ファイルを直接制御して実現します．

図1　LEDを制御するAndroidアプリケーションの画面

使用するGPIO

まず，BeagleBone BlackのGPIOについて確認しましょう．GPIOは，拡張端子P8，拡張端子P9のどちらにも割り当てられています．ここでは，P9のGPIOを利用します．拡張端子P9の詳細は，**表1**のとおりです．

BeagleBone Blackの拡張端子はピンごとに役割は固定化されておらず，複数の役割が割り当てられています．これらの役割を切り替えるには，ソフトウェアでピンごとにモードを指定します．

ここでは，LEDを拡張端子P9の12番ピンのGPIO（GPIO1_28）を利用し，LEDを接続，制御します．

接続方法

今回は，拡張端子P9の12番ピン（GPIO1_28）を利用します．なお，BeagleBone Blackの拡張端子のI/O電圧は3.3Vとなっており，12番ピンからLED，抵抗（100 Ω）を接続し，P2（GND）に接続しています（**図2**，**写真1**）．

表1　拡張端子P9の配置（13番ピンまでの抜粋）

ピン番号	プロセッサのピン番号	名　称	モード0	モード1	モード2	モード3	モード4	モード5	モード6	モード7
1，2	GND									
3，4	DC_3.3V									
5，6	VDD_5V									
7，8	SYS_5V									
9	PWR_BUT									
10	A10	SYS_RESET								
11	T17	UART4_RXD	gpmc_wait0	mii2_crs	gpmc_csn4	rmii2_crs_dv	mmc1_sdcd		uart4_rxd_mux2	gpio0[30]
12	U18	GPIO1_28	gpmc_ben1	mii2_col	gpmc_csn6	mmc2_dat3	gpmc_dir		mcasp0_aclkr_mux3	gpio1[28]
13	U17	UART4_TXD	gpmc_wpn	mii2_rxerr	gpmc_csn5	rmii2_rxerr	mmc2_sdcd		uart4_txd_mux2	gpio0[31]

第2章

NFSとtftpを使ったネットワーク・ブートで開発効率アップ！

Linux SDK クロス開発環境の構築

石井 孝幸

本章では，カーネルのビルドを含めた作業環境を構築します．ここでは，テキサス・インスツルメンツのLinux SDK（ソフトウェア開発キット）を使用します．

テキサス・インスツルメンツのLinux SDKは，ARMのツールチェインを含んだ形でインストールでき，NFS（Network File System）やtftp（trivial file transfer protocol）の設定もセットアップ・スクリプトで実行してくれるので簡単です．

Linux SDKのダウンロードとインストール

● テキサス・インスツルメンツのLinux SDKの特徴

テキサス・インスツルメンツのLinux SDKは，テキサス・インスツルメンツのウェブ・サイトからダウンロードできます．執筆時点での最新版は，ti-sdk-am335x-evm-06.00.00.00で，Linuxカーネルのリビジョンは3.2.0となっています．

最新のカーネルとはいかないのですが，テキサス・インスツルメンツのLinux SDKは，仕組みや環境が割とシンプルで作業内容が把握しやすく，Beagle Bone Blackで使用しているSoC AM3358専用に作られているので動作がそれなりに安定しています．

そのため，PC Linuxを使用したクロス開発環境でARM用Linuxのカーネルやアプリケーションを開発する工程がどのようになっているか，といった学習に向いています．

● ハードディスクの増設

では早速，前章で構築したPC Linux（Ubuntu）にARM用のLinux環境を構築していくのですが，仮想PC環境でLinuxを使用されている方は，新しいハードディスクを増設されることをお勧めします．

もちろん，ここで言う新しいハードディスクとは仮想上のハードディスクで，実際には仮想PCに接続するディスク・イメージ・ファイルを新しく作ることを指します．

仮想PC環境でしたら，物理的なディスク容量が許す限り仮想ハードディスクを追加し放題です．ですので，最初にPC Linuxをインストールした仮想ディスクとは別に，これから試していくARM用のいろいろなLinux，Androidごとに個別に仮想ディスクを用意すれば，UbuntuやAndroidと切り替えるたびにPC Linuxをインストールしなおす手間が省けます．

割り当てるディスクの容量の目安は，使い方にもよるのですが，Linuxで10Gバイト，Androidで20Gバイト以上は必要になります．

用意する仮想ディスクのイメージを仮想環境内で使った分だけ増えていくタイプのものにして，余裕を見て32Gバイトくらいを割り当てるのがよいと思います．

● ハードディスクのフォーマット

新しいハードディスクを追加したらPC Linux上でそのハードディスクを使用できるようにフォーマットする必要があります．

まずは，ハードディスクを追加してPC Linuxを起動します．Ubuntuでは，Disk Utilityを使用するとGUIでディスクのフォーマットが可能です．

まず初めに，新規ハードディスクのデバイス名を確認します．Disk Utilityの左側の「Storage Devices」タブから***Hard Diskをクリックします（**図1**）．一つ目のハードディスクが/dev/sda，二つ目のハードディスクが/dev/sdbのように順に割り当てられます．

/dev/sdaはUbuntuがインストールされているハードディスクなので，フォーマットしてしまうと

第4章

U-Bootのカスタマイズやドライバの作成に必須！

JTAGアダプタ+OpenOCDを使ったデバッグ環境の構築

袴田 祐幸／菅原 大幸

先代のBeagleBoneと比べてスペックが格段にアップし，価格も約半分の5,000円程度になったBeagle Bone Black．出荷時には，Ångström Linuxがプレインストールされており，ハードウェアを意識することなくアプリケーションを作成することが可能で，デバッガを利用しないで済んでしまうケースも多いと思います．

しかし，周辺機能のハードウェアなどを拡張し，U-Bootのカスタマイズやドライバなどの作成が必要となった場合に，どうやってデバッグするの？デバッガは何を使えばいいの？どんなソフトウェアを用意したらいいの？などと，お困りの方も多いと思います．

そこで本章では，BeagleBone BlackにJTAGアダプタを接続してデバッグを行う方法と手順を紹介します．今回は安価なJTAGアダプタとオープン・ソースのOpenOCDとEclipseを使用して，実際にU-Bootのデバッグを行います．

JTAGの機能とBeagleBone Black

● JTAGによるオンチップ・デバッグ

JTAGは，元々デバイスや基板の検査を行うために定められたバウンダリ・スキャン・テストの標準規格（IEEE1149.1）ですが，近年はこのJTAGの機能を利用し，オンチップ・デバッグを行う方法が一般的になっています．

オンチップ・デバッグとは，CPUに内蔵されたデバッグ機能を使って，プログラムのデバッグを行うための機能で，プログラムの実行や停止，メモリやCPUの内部レジスタの参照などを行うことができます．

● JTAG端子がないBeagleBone Black

ところで，先代のBeagleBoneを使用していた方のなかで新たにBeagleBone Blackを購入し，「さぁ，デバッグをしてみよう」と思ったとき，「ん？」と違和感を覚えた方は少なからずいると思います．

それは，BeagleBone Blackには先代のBeagleBoneにあったデバッグ用のUSBポートがなくなっているからです．

実は，先代のBeagleBoneでは，JTAG端子をUSBに変換するデバッグ用の回路が搭載されていて，USBケーブルを接続するだけでデバッグが可能でしたが，BeagleBone Blackではコストダウンのためか削除されてしまいました．

しかし，BeagleBone BlackにはCPUのJTAG端子を引き出したコネクタが用意されているので，ここにJTAGアダプタを接続することでオンチップ・デバッグを行うことができます．

デバッグに必要なハードウェアとソフトウェア

● デバッグに必要なハードウェア

BeagleBone Blackをデバッグする上で必要なハードウェアは，前述したJTAG端子を利用するための「JTAGデバッガ」です．

JTAGデバッガは世の中に数知れずあります．例えば，テキサス・インスツルメンツの純正JTAGデバッガには「XDS100エミュレータ」，ARM社の純正JTAGデバッガには「U-LINK」や「DSTREAM」などがあります．それぞれ機能や価格はまちまちですが，基本的にはJTAG端子を利用したJTAGデバッグが目的です．

今回は，JTAGアダプタとして「HJ-LINK/USB」（アルファプロジェクト）を使用します．

第1章

拡張端子を使って外部機器と接続

GPIO，A-Dコンバータ，PWM，I²Cの使い方

芹井 滋喜

BeagleBone Blackは，Linuxを搭載した名刺サイズのシングル・ボード・コンピュータです（**写真1**）．

名刺サイズの小型基板ながらLinuxを搭載し，USBホスト/デバイス，イーサネット，HDMI，microSDカード・スロット，拡張端子などを搭載しており，購入してすぐに開発を始めることができます．

BeagleBone Blackには豊富なサンプルがあり，SoC（Sitara AM3359）の内蔵周辺モジュールを使う際には，これらのサンプルが役立ちます．また，拡張端子からGPIOなどを使用することもできます．

本稿では，BeagleBone Blackの拡張端子を使ったGPIO，A-Dコンバータ，PWM，I²Cの使い方を解説します．

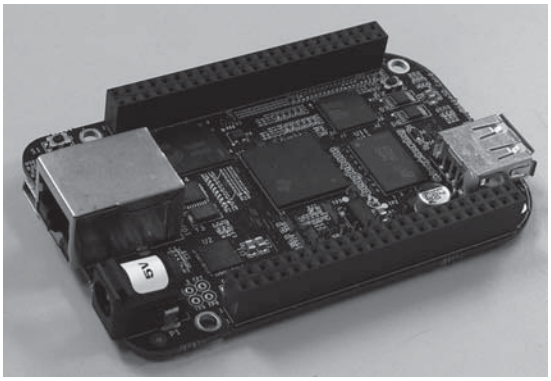


写真1　BeagleBone Blackの外観

BeagleBone Black セットアップ

● PCに繋ぐとUSBマス・ストレージ・デバイスとして認識される

BeagleBone Blackは，通常，USBでPCに接続してから使用します．

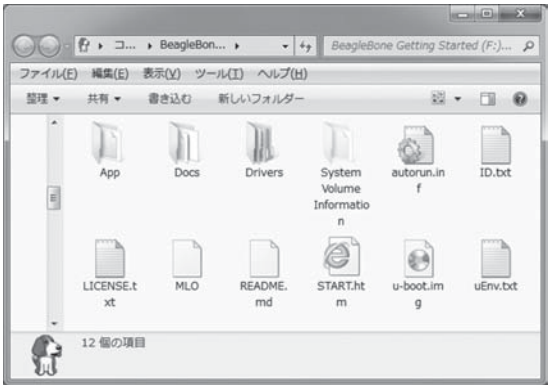


図1　BeagleBone Blackのマス・ストレージの内容

BeagleBone BlackをUSBでPCに接続すると，USBマス・ストレージ・デバイスとして認識されます．**図1**は，Windows PCにBeagleBone Blackを接続したときのフォルダの状態です．

BeagleBone BlackのUSBは複合デバイスとなっており，PCからはいくつかのUSBデバイスが認識されますが，マス・ストレージはWindows標準のドライバが使用できます．自動でドライバがロードされるため，最初の接続からフォルダを開くことができます．

● マス・ストレージ以外のドライバのインストール

マス・ストレージ以外のドライバは，以下の手順で別途ロードする必要があります．

BeagleBone Blackのフォルダには，START.htmという名前のHTMLファイルがあり，ウェブ・ブラウザで開くことができます．**図2**は，START.htmの画面です．

START.htmのStep #2の項目に，**図3**のように，対応OSの一覧が表示されています．

該当OSの「USB Drivers」セル内の文字列をクリッ

第4章

AM3359内蔵マルチチャネル・オーディオ・インターフェース

McASP＋外付けコーデックを使ったオーディオ入出力

袴田 祐幸／菅原 大幸

本章では，BeagleBone Blackにアナログ・オーディオ入出力機能を追加する方法を紹介します．BeagleBone Blackのオーディオ出力はHDMI経由なので，従来のアナログ入力タイプのヘッドホンやスピーカを直接接続することができません．

オーディオ入出力を追加する方法として，USBスピーカやUSBマイクを接続する方法もありますが，今回はAM3359が内蔵しているオーディオ用のインターフェース「McASP」を使って回路を拡張し，アナログ・オーディオの入出力ができるようにします．

I2SやS/PDIFなどをサポートするMcASP

McASP（Multichannel Audio Serial Port）とは，テキサス・インスツルメンツのCPUやDSPに搭載されている汎用オーディオ・シリアル・ポートで，I2SやS/PDIFのほか，複数の通信フォーマットをサポートしています．

McASPの信号は，基本的にクロック，データ，フレーム同期信号で構成されます．**表1**に，AM3359のMcASPの端子一覧を記載します．

McASPはあくまでディジタル・データの通信を行うためのものなので，アナログ・データとの変換を行うためにオーディオ・コーデックICに接続して使用します．

ハードウェアの構成

今回は，BeagleBone Black拡張ボード「XG-BBEXT」（アルファプロジェクト）の構成を例に説明します．

「XG-BBEXT」では，オーディオ・コーデックICにテキサス・インスツルメンツのTLV320AIC3106を使用しています．オーディオ・コーデックの選定にあたっては，McASPに対応していることと，Linux用のドライバが公開されていることを基準としました．公開されているドライバを利用することで，ソフトウェアの開発負担を大きく軽減することができます．

図1に，AM3359とTLV320AIC3106の接続図を示します．BeagleBone BlackとTLV320AIC3106は，McASPとI^2Cで接続します．

● McASPの接続

今回は，McASPのマスタ・クロックにはCPUの内部クロックを使用するので，マスタ・クロック入出力端子（McASP0_AHCLKX/AHCLKR）は未接続としま

表1　McASP端子一覧

機　能	説　明	本章での使用例
McASPx_AXR[3:0]	オーディオ・データ信号の送受信	McASP0_AXR2 出力に設定して使用 McASP0_AXR0 入力に設定して使用
McASPx_ACLKX	送信ビット・クロック	McASP0_ACLKX 入力に設定して使用
McASPx_FSX	送信フレーム同期	McASP0_FSX 入力に設定して使用
McASPx_AHCLKX	送信マスタ・クロック	内部クロックを使用するため使用しない
McASPx_ACLKR	受信ビット・クロック	McASP0_ACLKX，McASP0_FSX と同期して動作する設定で使用するため使用しない
McASPx_FSR	受信フレーム同期	
McASPx_AHCLKR	受信マスタ・クロック	内部クロックを使用するため使用しない

第5章

50MHz，16チャンネル，Androidを使って波形表示

BeagleBone Black＋FPGAボードで作るロジック・アナライザ

岩田 利王

Androidで動くロジック・アナライザを作ろう！

「組み込みAndroid」と聞くと何やらとっつきにくいイメージがあるかもしれませんが，BeagleBone Blackの登場によりそのハードルが一気に下がりました．

今回はBeagleBone BlackとFPGA（Field Programmable Gate Array）を組み合わせることにより，直感的で分かりやすいGUI（Graphical User Interface）と，高速な信号処理機能とを兼ね備えたシステムを実現します．

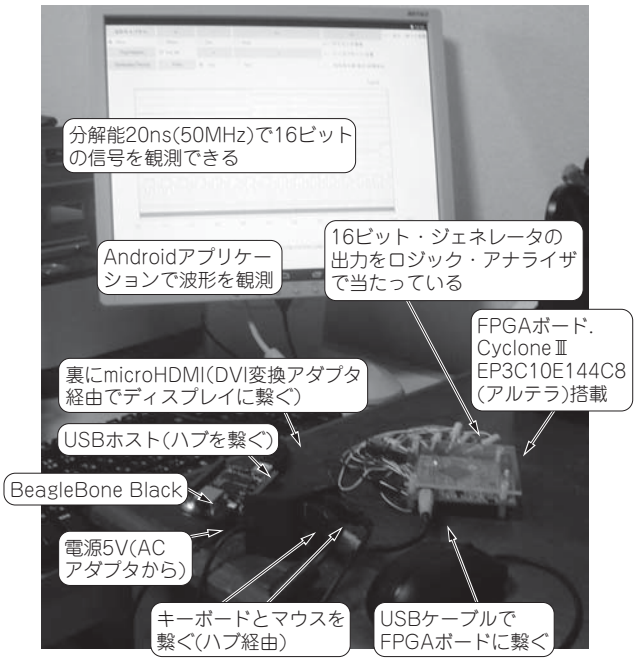


写真1　何を作るか――こんなロジック・アナライザです

- これが今回製作する「BBBロジック・アナライザ」！
- BeagleBone BlackのUSBホスト・ポートは1個しかないのでハブ（3ポート以上）を介してマウス，キーボード，FPGAボードに繋ぐ
- FPGAボードはBeagleBone BlackからUSB経由で給電される
  このようすは以下のサイトで見ることができる
  http://www.youtube.com/watch?v=xMH5YJ3hof8

BeagleBone BlackとFPGAボードで作る理由と方法

ロジック・アナライザのような測定器を実現するには，①信号をいかに高速に取り込むか，②GUIをいかに構築するか，が主なキーになります．

今回は①を実現するのにFPGAボード，②を実現するのにBeagleBone Blackを使用します．本節ではこれら二つのボードを採用した理由，さらに双方のインターフェースをとるのに効率的な方法は何かを説明します．

写真2　BeagleBone Black上で走るAndroidアプリケーションで波形描画

- これがロジック・アナライザ用Androidアプリケーション－BBB APP！
- BeagleBone Blackに搭載のmicroHDMIコネクタからHDMIケーブル→DVI変換アダプタを介してディスプレイに繋ぐ
- アプリケーションの[波形キャプチャ]ボタンを押すたびにロジック・アナライザの波形が更新される

インターフェース SPECIAL
Linuxガジェット BeagleBone Blackで I/O
CQ出版社 〒170-8461 東京都豊島区巣鴨 1-14-2 TEL.03-5395-2141(営業部)
定価：本体2,800円＋税