ハードウェア実験

組み込みシステム入門
第8回

2010年11月11日

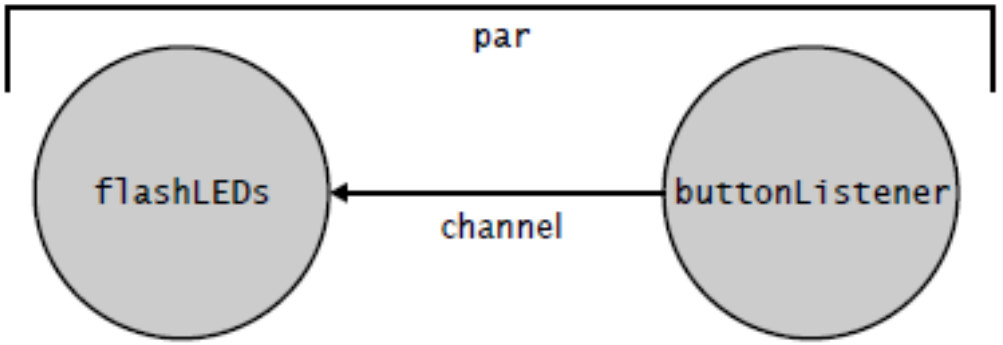
マルチスレッドと状態遷移の設計

前回予告した今回の内容

- ICトレーナ上のスイッチを読み込みます。
- 複数のスレッド間で、データの受け渡しを行うプログラムを課題とします。
  - 4番目のサンプルプログラム(Buttonプロジェクト)を改変します。
- 「状態機械」の動作をプログラムしてみます。

button projectの構造

- 並列に動作する二つのモジュールがある
  - flashLEDsと、buttonListener
- 二つのモジュールの間にchannelがある
  - buttonListener側で書き込み
    - c <: led;
  - flashLeds側で読み出し
    - case c :> ledVal :
    - caseがついている ⇒ Eventとして受けている。



button projectでのEvent

- caseで記述されている行
  - 43行目
    - タイマー・イベントを受け付けている。
    - case t when timerafter(time) :> void:
  - 56行目
    - 通信チャンネルからのLEDの値
    - case c :> ledVal :
  - 71行目
    - ボタンが押されたイベント
    - case button when pinseq(0) :> void:

何がEventか？

- システム側から見て、予期できない事象
  - ボタンが押された
    ⇒　人が操作している
  - 通信チャンネルからの値の受け取り
    ⇒　他のスレッドの動作は、関知しない
  - タイマーが予定時刻になった
    ⇒　時計をじっと見ている訳ではない。
- ※ タイマーは「わかる」のではないか？
  - じっと見ていればわかるが、それではタイマーの意味がない。（他の作業をしたいから、アラームや目覚まし時計を設定する！）
  - タイマーを一度セットしたら、タイマーからの通知は受け付けるものの、他の作業をするので「じっと監視」はしない！

なぜ、buttonListenerだけ別のスレッドにした？

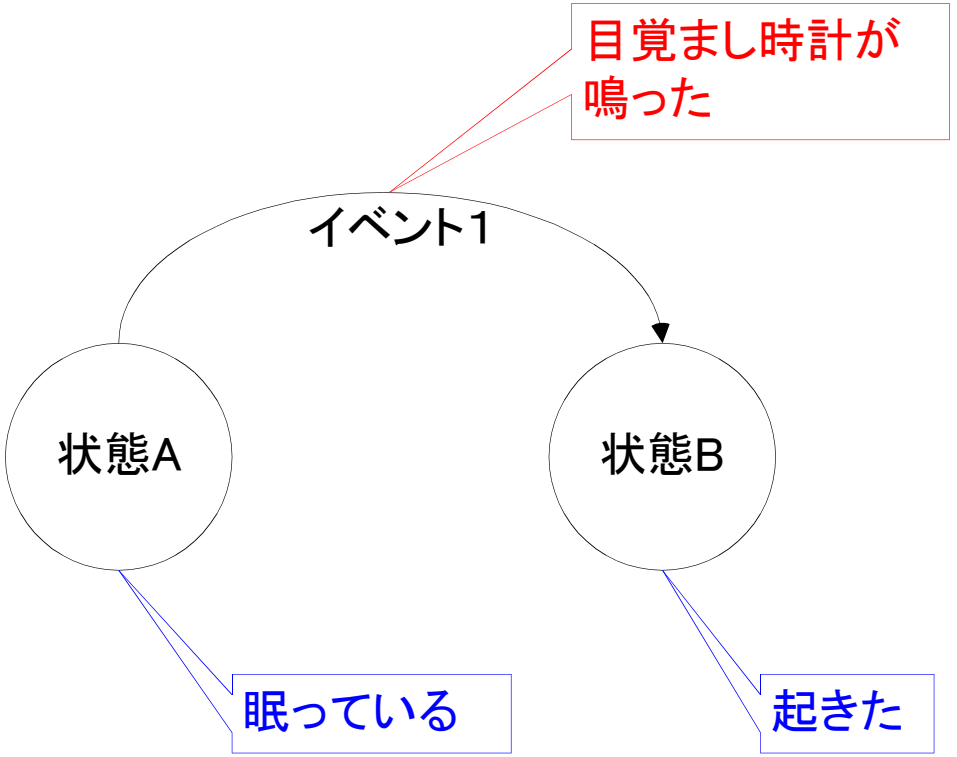
- buttonListenerの①のSelectでは、ボタンが押されるまでsleepする。
- このcase文は、他のEventと同じselectに書くことができる。
- ところが、一度ボタンが押されると②の部分でsleepし、ボタンが離されるまで待つ。
  - ここで待ってしまうと、Selectの他の選択肢が処理できない。

```
64 void buttonListener(in port button, chanend c)
65 {
66   int led = 1;
67   while (1)
68   {
69     select                                    ①
70     {
71       case button when pinseq(0) :> void:
72       {
73         c <: led;
74         led = (led + 1) & 0xF;
75
76         button when pinseq(1) :> void;        ②
77
78         break;
79       }
80     }
81   }
82 }
83
```

このために、button処理のpinseq は、タイマーなどからスレッドを分けて、独立したスレッドで走らせている。

Eventと状態遷移図

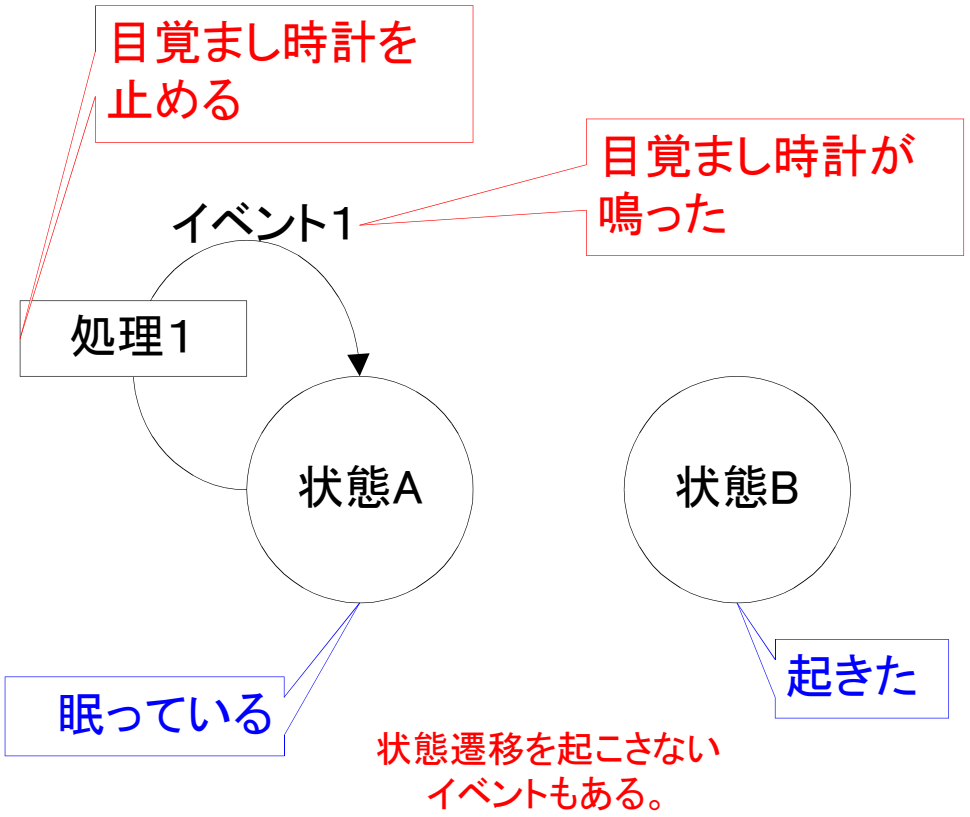
- Event駆動型のシステムでは、システムの「状態」を定義する。
- それぞれの状態で、Eventが発生した際に、「状態の遷移」が起きる。
  - 起きると定義する
- 状態が遷移する際に特定の処理を行う場合がある。



Eventの発生と処理と状態

- Event発生に伴い、特定の処理を行う。

  （行わない場合もある）
- 特定の処理だけ行い、状態の遷移を伴わないイベントもある。
- 状態遷移を伴わないイベントも、処理は明記する。



button projectの「状態」

- 「状態」は、スレッドごとに定義する。
- flashLedsでは、「状態」は二つ
  - LEDが点灯している (isOn が true)
  - LEDが消灯している (isOn が false)
- buttonListenerでは、「状態」は一つ
- それぞれの状態で、Eventにどう対処するか、表にまとめる。
- 表の列は「状態」、行は「イベント」で、ある「状態」の時にある「イベント」が発生したら、どんな処理を行い、どの状態に移行するかを表にまとめる。

	状態1 (LED点灯)	状態2 (LED消灯)
タイマー	LEDを消灯させる →状態2へ遷移	LEDを点灯させる →状態1へ遷移
チャンネル入力	処理：ledValへ代入	

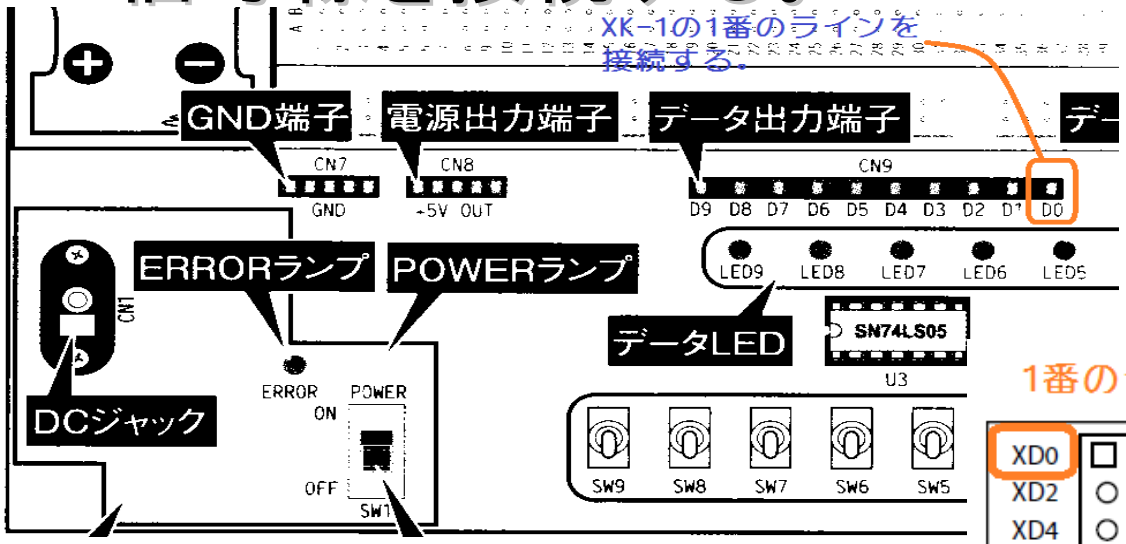
練習課題1

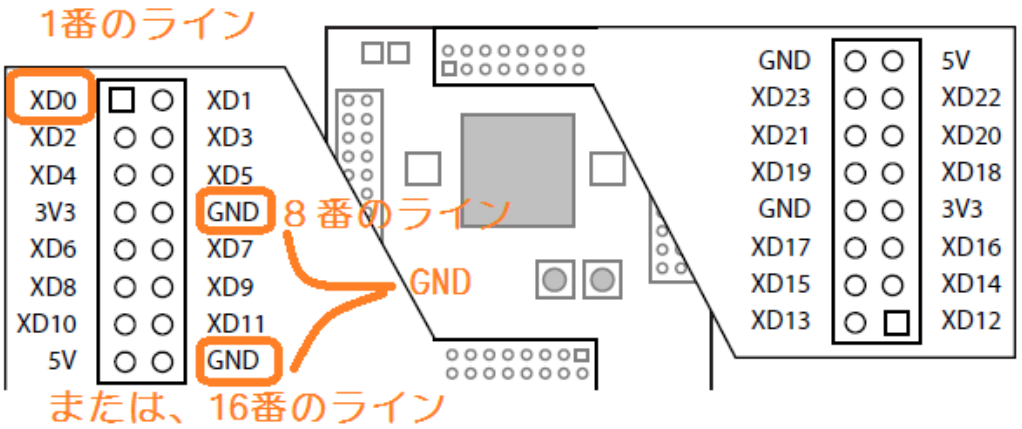
- ICトレーナ上のスイッチの読み込み
  - ICトレーナ上のスイッチを、一つのポートで読み込みます
  - ICトレーナ上のスイッチをONにすると、XK-1上のLEDが点灯し、OFFにすると1つが消灯するプログラムを書いて下さい。
- P1Aか、P1Bを使ってみよう。
  - XS1_PORT_1Aでアクセスするポートは、
    - P1A0 → XD0 → 1列目上 → 1番線

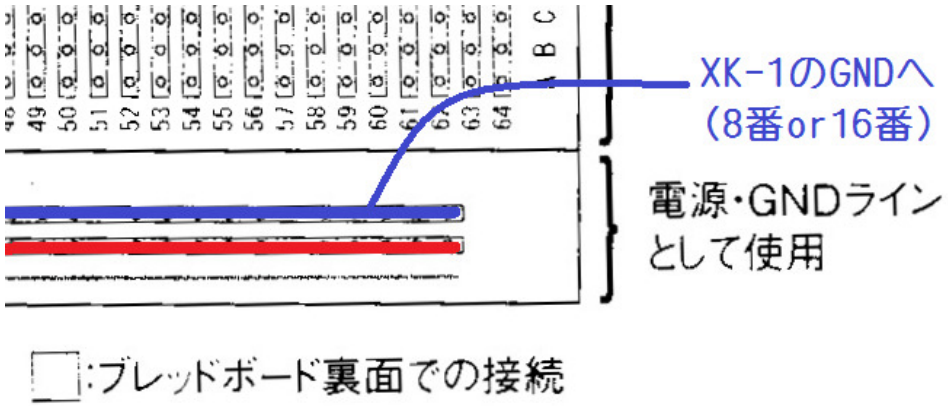
Pin	Port			Processor
	1b	4b	8b	
XD0	P1A0			
XD1	P1B0			

XK-1とICトレーナの接続

- GNDラインは必ず接続する。
- 信号線を接続する。







□:ブレッドボード裏面での接続

報告課題

- 以下のプログラムを書いて下さい。
- 「点滅中」と、「待機中」の二つの状態を作る。
  - 「待機中」は、XK-1上の4つのうち特定のLEDが点灯。
  - 「点滅中」では、さらに「点灯中」と「消灯中」の二つの状態があり、交互に点灯と消灯を繰り返す。
- ICトレーナ上のスイッチ（任意の一つ）がONになったら、「点滅中」になり、OFFになったら「待機中」になるようにする。
  - ICトレーナ上の一つのスイッチで、LEDを切り替える。
- 発展課題A
  - XK-1上のスイッチ1を押すとLEDがカウントアップするようにする。
  - 1回押すごとに、LED表示の4ビット値がカウントアップする。
- 発展課題B
  - XK-1上のスイッチ2を押すと、点滅の周期が変化するようにする。
  - 点滅の周期が一定以上短くなったら、最初の状態に戻す。

スレッドの追加

- 「プッシュ」ボタンで、「押された状態だけに反応」させ、OFF→ONをイベントとして、ON→OFFの変化はsleepで処理する場合（button projectの例）、ボタンごとにスレッドを追加する必要がある。
- ボタンプロジェクトと同様の手法で、

  button1Listener

  button2Listener

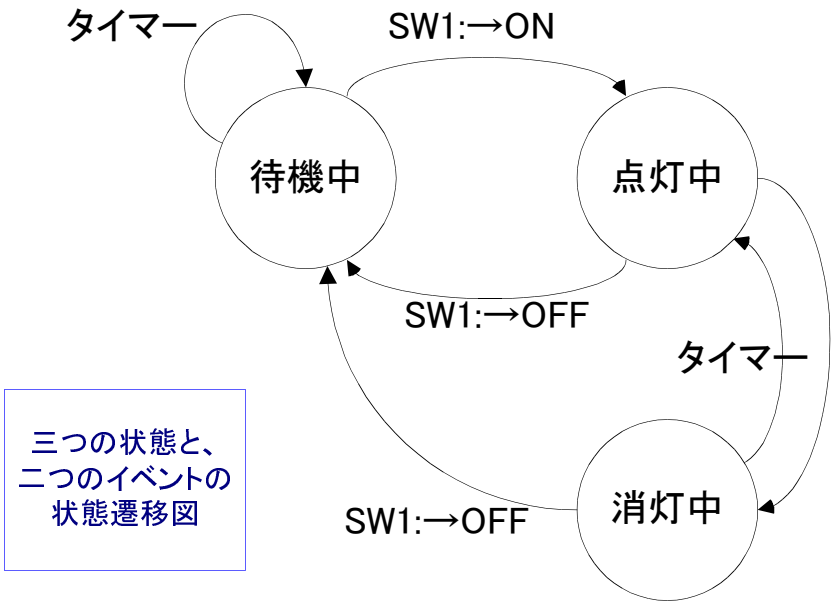
  の二つのスレッドを用意する。
- それぞれとの間に、チャンネルを用意する。
  - chan c1, c2;

マルチスレッドを使わない場合

- ボタンイベントで、「ON→OFF」に戻る部分の処理を、スレッドのsleepという形で実装しない場合には、「ON→OFF」も「状態の変化を起こす、イベント」として対応する。
- この場合には、シングルスレッドで全ての状態を管理して、利用するスイッチの「ON→OFF」、「OFF→ON」や、タイマーなどのイベントを検知して「状態遷移」をプログラムする。
  - 最初に、わかりやすい「状態遷移図」や、「状態遷移表」を設計することが大切。
- マルチスレッドを使えば、状態遷移は単純化されるが、シングルスレッドなら状態管理は若干複雑化する。（どちらを取るかは自由）

状態機械の定義

- 状態定義は、メインとなるスレッドで管理する。
  - flashLedsがmain?
- 複数のスレッドの「状態」の組み合わせを、全体としての「状態」として定義する考え方もできるが、今回の課題では、プッシュボタンのスレッドでは、「ボタンの処理」と、チャンネルを通じての通知だけを行う。



三つの状態と、二つのイベントの状態遷移図

マルチスレッドと状態定義

- 複数のスレッドの「状態」の組み合わせを、全体の「状態」として定義する考え方もできる。
  - [マルチスレッドにする場合]今回の課題では、プッシュボタンのスレッドでは、「ボタンの処理」と、チャンネルを通じての通知だけを行う。従って、スレッドの「状態」を定義する必要はない。
- なるべく、「単純」に設計する
  - KISS：Keep It Simple and Stupid
  - 設計をごちゃごちゃと「複雑化」させず、「わかりやすく」しておく。
  - 「定義」の仕方一つで、メンテナンスしやすさが変わってくる。

課題の報告について

- 最初に、「動作仕様」を明記して下さい。
  - どういう動作をさせようとしたか。
  - スライドP12のような、文章で記述する。
- ICトレーナとの接続
  - 何本の配線を行い、どのラインをどのピンに接続したか。
  - できれば図で、報告する。
- 状態遷移図 and/or 状態遷移表
- プログラムリスト
- 「動作についての報告」

報告課題の評価について（目安）

- D評価：プログラムリストがないもの
- C評価：プログラムリストがあるだけ、あるいは、リストに「写真」などが添付されているだけで、設計仕様、設計上の主眼（注意した点）の説明がないもの。
- B評価；設計仕様とリストが明記されていて、何をどう考えてそうした設計になったか、思考のプロセスが明確になっているもの。
- A評価：B評価の条件に加えて、動作の検証が緻密（単に「動きました」だけではない）なもの、もしくは、発展課題も行っているもの。
- S評価：自分で課題を発展させているもの、あるいは、報告が優れているもの

次回の予告

- ライントレースカーを組み上げて、接続のための準備を行います。
- XK-1とライントレースカーが接続できることを試します。
- （この「次回の予告」ページは、予告なしに改変する場合があります。）