

かゆいところに  
手が届く!

# はじめての組み込みシステムプログラミング

## SmartBB!!! 36077 学習キット

### チュートリアル

## はじめに

今日では組込みシステムは様々な機器に採用され、組込みシステムプログラマの需要が益々増えています。

しかし「組込みシステムプログラミングを勉強したいが、何から手を付けたらよいか分からない」と言う声がよく聞かれます。

本キットではそういった問題を解決するため、初歩からの学習を手助けする「かゆい所に手が届く」学習用キットを目標に開発されました。

このチュートリアルはマイコンのさまざまな応用を用意し、マイコンとはどんなものか、どんなことができるのかということを実際に動かしてみることで学習していきます。

このキットでの学習を通じて組込みシステムプログラミングに慣れて頂き、結果として日本の組込み技術者が増えてくれる事を願っております。

製作・著作 株式会社北斗電子

## 目 次

1. 組み込みシステムの概要.....	3
1.1. 組み込みシステムとは？.....	3
1.2. マイコンからの制御.....	4
2. 組み込みシステムの基礎.....	6
2.1. ....	6
2.1.1. 実際に電源を入れて LED 点灯デモプログラムを実行してみよう！.....	6
2.1.2. ハードウェアの仕様とプログラム.....	7
2.2. LED1 の点滅プログラムを作成してみよう！.....	8
2.2.1. ピンとビット(※1)が対応している！.....	9
2.2.2. LED1 点滅プログラミングの流れ.....	11
2.2.3. いよいよプログラミング！概要を見てみよう.....	12
2.2.4. 実際にプログラムを作成してみよう！.....	13
2.2.5. ビルドをしてみよう！.....	15
2.2.6. マイコン内の ROM に書き込み実行してみよう！(※1).....	16
2.3. ボタンの入力で点灯 LED を移動させよう！.....	18
2.3.1. 端子の状態を見る.....	19
2.3.2. 実際にプログラムを作成してみよう！.....	20
3. 組み込みシステム構築の実際と応用.....	22
3.1. 割り込み処理の概要.....	22
3.1.1. プログラムカウンタとスタック.....	23
3.1.2. 割り込み要因と割り込みベクタ.....	24
3.2. ボタンの入力で点灯 LED を移動させよう！ その 2.....	25
3.2.1. IRQ を使用するための設定.....	25
3.2.2. 実際にプログラムを作成してみよう！.....	27
3.2.3. さあビルド！.....	30
3.3. タイマ割り込み.....	32
3.3.1. タイマ割り込みとコンペアマッチ.....	34
3.3.2. タイマ Z0 を使う為の設定.....	35
3.3.3. 実際にプログラムを作成してみよう！.....	38
3.3.4. さあビルド！.....	40
3.3.5. 結果.....	40
4. マイコンによるモータの制御.....	41
4.1. モータ制御の概略.....	41
4.1.1. モータを回転させるには？.....	42
4.1.2. モータドライバとしてのデジタルパワーアンプ.....	42
4.2. タイマ Z0 を用いてモータ制御プログラミングを作成してみよう！.....	44
4.2.1. タイマ Z とデューティ比.....	46
4.2.2. 設定の手順.....	48
4.2.3. 実際にプログラムを作成してみよう！.....	49
4.2.4. ビルド&結果！.....	51
5. まとめ.....	52
6. TRY 回答例集.....	53

# 1. 組み込みシステムの概要

## 1.1. 組み込みシステムとは？

組み込みシステムとはマイコンを頭脳とするロボットのような物です。

ここで言うロボットとは必ずしも人型のロボットでは無く、ある目的の仕事をするために作られた機械等の物です。

例えば洗濯機も“洗濯ロボット”と言えます。形こそ人の形はしていませんが、人が洗濯物を洗濯槽に放り込んで開始ボタンを押して置くだけで、お昼寝をしている間に洗濯という仕事をこなしておいてくれます。

これは“洗濯ロボット”が“洗濯”→“脱水”→“濯ぎ”→“脱水”→“乾燥”→“お知らせ”までの仕事の要所を自分で判断して行う事が出来るからです。

筆者の私は 30 年前の洗濯機を記憶していますが、ある一定時間に洗濯槽の中のプロペラが一定方向に回り続ける物でした。

時間が来ると人の手で洗濯物を脱水槽に移し変えます。そこにはバネ仕掛けで時間を測るタイマはありましたが、要所所で洗濯機が自主的に判断する場面は無く、結局は人の手を充てにした道具でした。

それでは便利な道具ではありますが、とても自主的に動いてくれるロボットとは言えません。

頭脳であるマイコンが洗濯機に組み込まれるようになり、そのシステムが洗濯ロボットになっているのです。

その様な組み込みシステムは現在では洗濯機だけではなく、テレビ、冷蔵庫、エアコン、炊飯器、携帯電話、自動車など我々の身の回りに採用されています。

## 1.2. マイコンからの制御

ではマイコンが頭脳となり洗濯ロボットを構成するにはどのような仕組みを取れば良いのでしょうか？

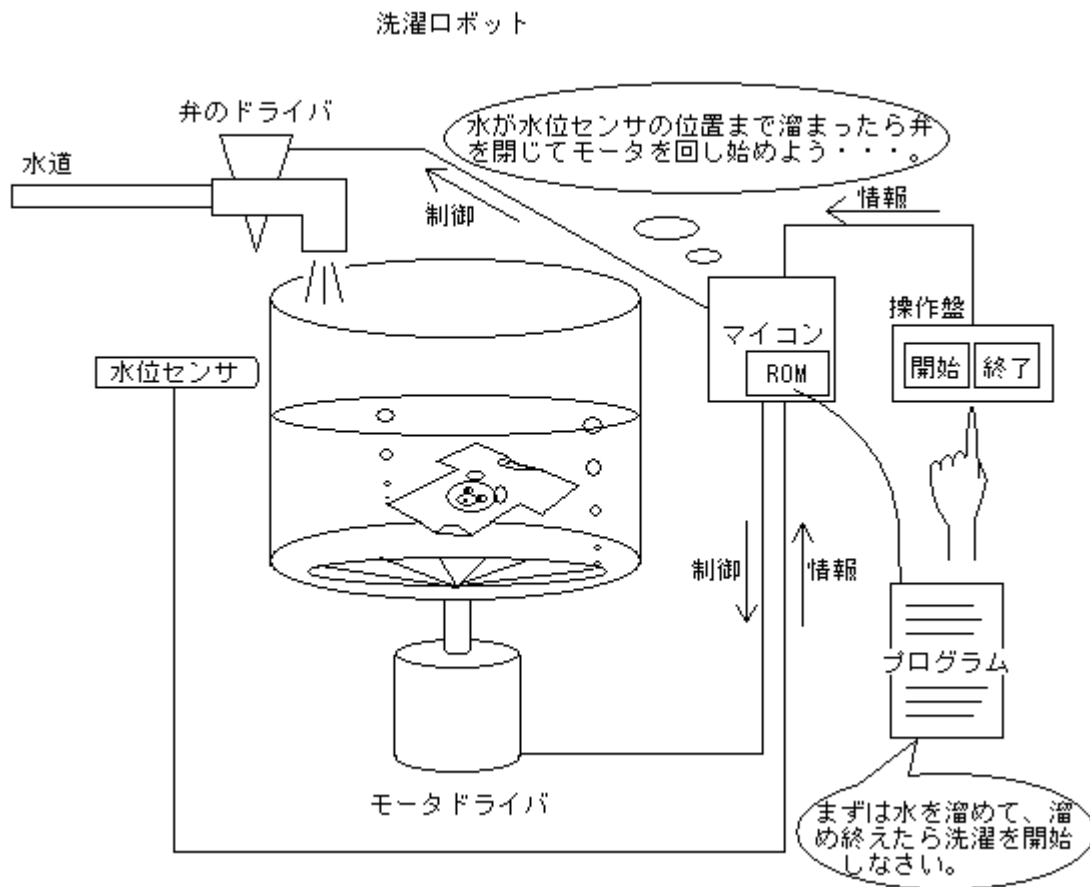


図1. 洗濯ロボットの様子

状況に応じてどう動くかを判断するのはマイコンです。ただ、マイコンが判断をするには判断材料が必要です。

またマイコンが自ら物理的な力を発揮して洗濯をするわけではなく、実際に作業をする部品も必要になります。

実際に洗濯ロボットはマイコンの他にセンサやモータなど様々な部品で構成されています。

センサからの電気的な情報をマイコンが受け取り、どう動くかを判断してモータドライバに指示をし、制御します。

今洗濯槽に水を溜めて行き、いっぱいになった時点でプロペラを回転させて衣類の汚れを取りたいとします。

まずはマイコンが洗濯槽に水を注ぐように弁ドライバを制御します。次に水位センサからの情報を受け取り水位を監視します。

水位センサから来る情報で水が必要十分溜まったと判断したら、弁ドライバに水を止めさせモータドライバにプロペラ回転を指示します。

さて、ここでさり気なく「必要十分」と言う文字を書きましたが、何を以って必要十分だと判断出来るのでしょうか？

そもそもマイコンが洗濯をしようとするのは何故でしょうか？

それはマイコンが洗濯をする為にはどう動き、どう判断すればよいか ROM という記憶装置の中にあらかじめプログラムとして書かれているからです。マイコンはただ直向にそのプログラムを実行しているだけなのです。そしてそのプログラムは我々人間が作成するものです。

本キットでは C 言語を用いてプログラムを開発する基礎を学んでいきます。

## 2. 組み込みシステムの基礎

### 2.1.

第1章の中で「マイコンって何だろう?」と言う部分を説明しました。いきなりですが本キットの中央に搭載されている黒い正方形の物体 H8/36077F がマイコンです。

このマイコンはプログラムを記憶できる ROM を内蔵しておりますので、今後学習で作成するプログラムは一旦そこに書き込み、実行する事になります。

洗濯ロボットではマイコンが制御する相手はモータドライバ<sup>(※)</sup>だったり弁のドライバだったりしました。更に情報をもろう相手としてセンサや操作盤がありました。

本キットではマイコンの他それらドライバに相当するものがあらかじめ一枚の基板の上に搭載されていますので、容易に組込み制御の学習が出来ます。

ではまずマイコンが弁のドライバを制御する様子を見てみましょう。とは言ってもボード上には水道弁のドライバはありませんので代わりにまずは単純な代替品として搭載されている8つのLEDを使います。

(※)本書で言う「ドライバ」とは、微弱電流の信号で指示を受けてそれに応じて大電流の部品を動かす「ユニット」を意味します。

#### 2.1.1. 実際に電源を入れて LED 点灯デモプログラムを実行してみよう!

このマイコンには出荷時にあらかじめ LED 点灯を制御するデモプログラムがマイコン上に書き込まれています。(※)

では実際に電源を J7 に挿入して下さい。LED が点灯し暫くすると LED1 以外が消灯する事が確認出来ましたでしょうか? このマイコンボードには電源投入時にリセットが掛かり、プログラムが先頭から実行される仕組みがあるため電源を投入するだけでデモプログラムを実行出来ました。

「え!?ただこれだけ?」と思われる事でしょう。そう、ただこれだけです。でもこれはまずは LED を点灯させ、ある一定時間が来たと判断したら消灯するという LED の制御をマイコンがプログラムに従って動作した結果です。

今回は目に見える部分が単純な LED だったので心に訴えるような結果ではなかったと思いますが、これが LED では無く水道に繋がっている弁のドライバだったらどうでしょう?

一定時間水が放水されてから自動的にキュッと止まる、こういう動作になります。これなら洗濯ロボットの動作の一部をこのデモプログラムで実現した事になります。

(※)出荷時に予め書き込まれているデモプログラムは他のプログラムを上書すると消えてしまいます。また、添付 CD には収録されておりませんので最初の動作確認用です。

## 2.1.2. ハードウェアの仕様とプログラム

では先程の実験でマイコンはどのようにLEDを点灯/消灯させたのでしょうか？  
マイコンは電気信号で相手を制御します。図2を見て下さい。

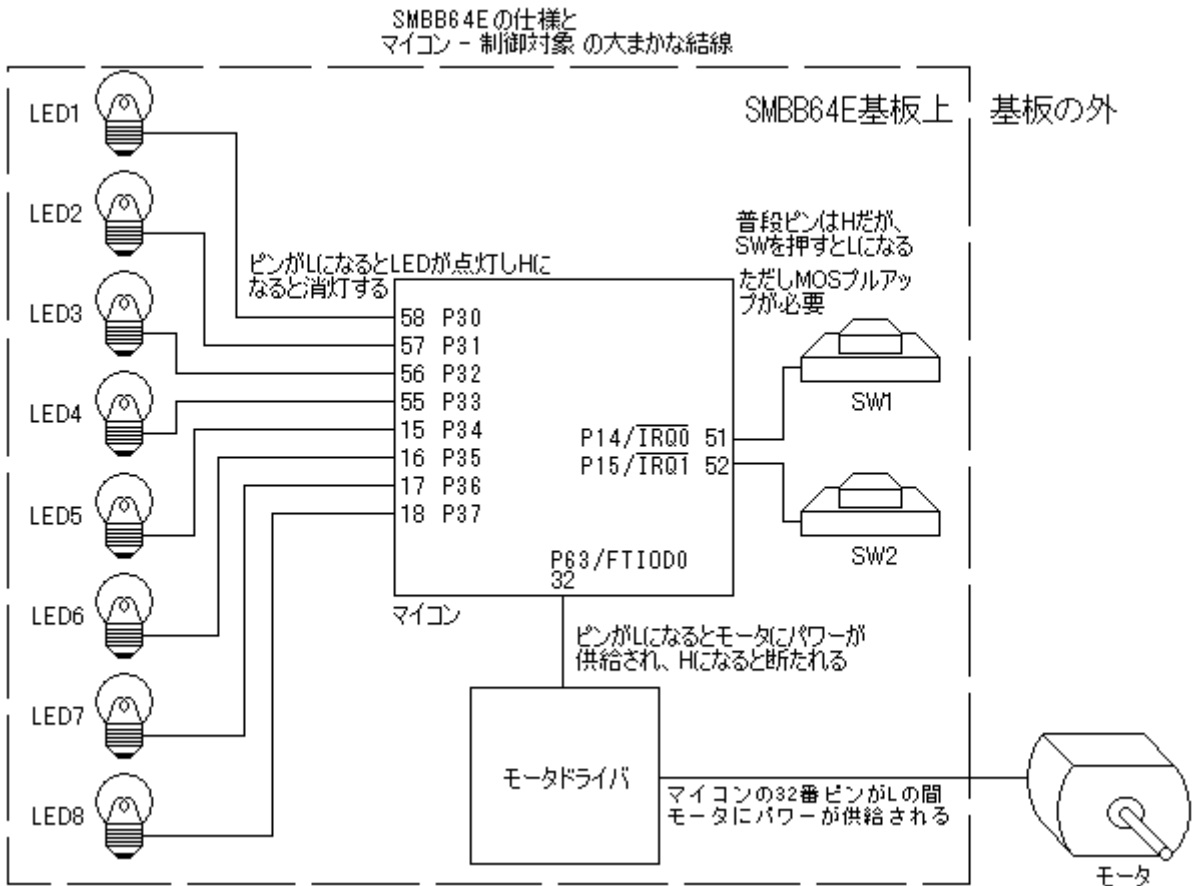


図2. SMBB64Eの仕様と、マイコン - 制御対象 の大まかな結線

ここではマイコンの「どこ」と「制御対象」が繋がっているか、「どういう」信号をやり取りするかと言う SMBB64E のハードウェア仕様が示されています。本キット付属の SMBB64E の仕様から本書で取り上げる部分を抜粋した物です。(※1) 組み込みシステムをプログラミングするにはそのハードウェアの仕様を知らなくてはなりません。

図2の説明をします。まずマイコンの上に記されている「ピン番号」とはマイコンから出ている端子の番号です。基板上に半田付けされている金属部分が端子です。それぞれの端子に番号が割り振られています。次に「H」と「L」という表現が出てきますが、これは電氣的な状態を表す表現です。「H」と「L」の2つしかありません。(※2)

先程の「点灯していたLEDが消灯した」という動作はLEDと接続されているマイコンのピンがプログラムによってL→Hに制御された結果です。

大まかに言うと、組み込みシステムでの「プログラムによる制御」とはハードウェアの仕様に従ってマイコンのどこのピンをどういう状態にするか？と言う事をプログラムする事と言えます。(※3)

(※1) 注目すべき点に絞って簡素化してあります。

(※2) この様にHとLで表す信号をデジタル信号と呼んでいます。

(※3) マイコンのピンの状態を制御するだけでなく、ピンの状態を読む事も出来ます。

## 2.2. LED1 の点滅プログラムを作成してみよう！

ではここで課題です。

### Q. SMBB64E の LED1 を無限に点滅させるシステムを作成しなさい。

方法を考えてみましょう。まず 2.1.2 の図 2 で SMBB64E のハードウェアの仕様を見ます。そこから電氣的にマイコンの 58 番ピンを H→L→H→…に制御すれば良い事がわかります。

ここからソフトウェアの出番です！ どう制御するかを知るために、ここで使用マイコンのハードウェアマニュアルを参照します。添付 CD に収録されているマイコンのハードウェアマニュアルを開いて下さい。(※1) まず見る場所は「概要」の「ピン配置図」です。マイコンのハードウェアマニュアルのピン配置図には 58 番ピンの所に P30 と書かれています。この頭の P は「ポート」を意味します。ポートとはマイコンのピンの状態を管理する単位で、その後続く「3」はポート番号 3、その次の「0」は 0 番目を意味します。よく見ると LED2 に対応する 57 番ピンには P31、LED3 に対応するピンには P32、LED4 に対応するピンは…と LED1～8 にはそれぞれ P30～P37 が対応している事がわかります。

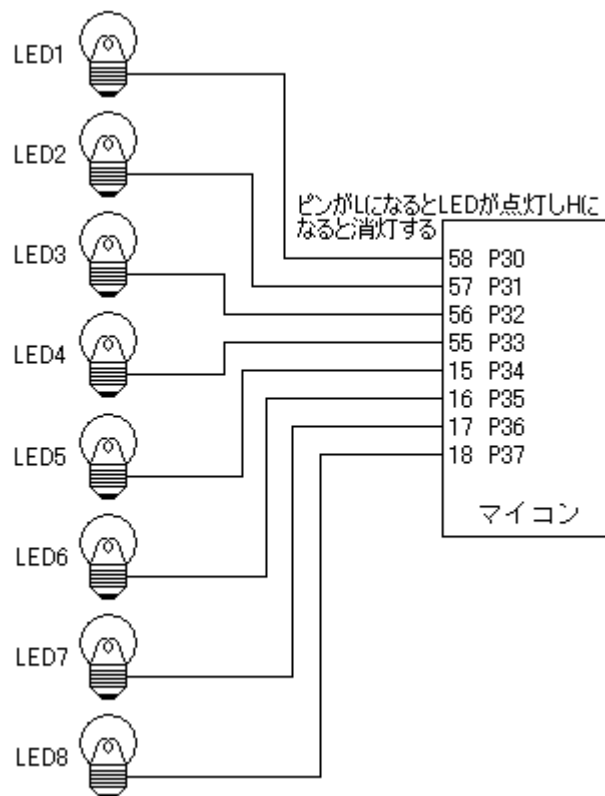


図3. “LED”と”ピン”と”ポート”の関係

さて、ここで話を戻して改めて 58 番ピンの制御を考えます。58 番ピンを制御するにはポート 3 の 0 番目を変更し、制御します。

どう変更するか…。ここからが多少複雑になりますが大事な部分ですので理解できない場合は読み飛ばさずに反復するようにして下さい。

まず、変更はレジスタ単位で行います。レジスタとはマイコンの中に設けられている設定スイッチの様な物で、プログラムによって設定できます。(※2) H8/36077F マイコンのハードウェアマニュアルでポ

ート3について調べると(※3)、ポート3にはポートコントロールレジスタ3(以降PCR3)とポートデータレジスタ3(以降PDR3)と言う2つのレジスタが存在し、それらをプログラムから設定することで対応するピンを制御できる事が判ります。

まずマイコンのハードウェアマニュアルによるとポート3は汎用入出力ポートである事が判ります。汎用入出力って何でしょう？ 入出力ポートはI/O(INPUT/OUTPUT)ポートと呼ばれ、マイコンからピンの状態を読み取るINPUT(入力)、マイコンからピン状態を制御するOUTPUT(出力)から成り立っています。今回はLED1の状態つまり58番ピン(P30)をマイコンから制御するため、出力ポートとしなくてはなりません。(※4)

ここでPCR3の登場です。PCR3を設定する事でポート3の入出力を設定出来ます。次にPDR3が登場します。ポート3がPCR3によって出力端子となっているので、PDR3に設定した内容がポート3の状態になります。どう設定したら良いのかを次節で説明していきます。

(※1)マイコンのハードウェアマニュアルの内容はリビジョンにより場所や書き方が異なるかもしれません。巻末に注意事項があるので必ずお読み下さい。

(※2)全てのレジスタが制御できるわけではありません。

(※3)素早いマイコンのハードウェアマニュアル参照には「慣れ」が必要です。何がどこに書いてあるか？は慣れれば大丈夫です！

(※4)逆にピンの状態PDR3に読み込むには入力ポートとします。

## 2.2.1. ピンとビット(※1)が対応している！

SMBB64Eではポート3のPCR3とPDR3を変更するとLED1~8までを制御出来る事は解りました。では、LED1、LED3など、個別に制御するにはどうしたらよいでしょうか？

ここで2進数が登場します。2進数とは0と1で表現される数字で、「ある」か「無し」しか無く中間がありません。LとHを0と1に置き換えると両者の関係が非常に似ている事に気が付きましたか？

実はここではPCR3とPDR3のP30~P37それぞれの個別設定を2進数の0と1で行います。

まずPCR3でP30~P37全てを出力ピンにしたいと思います。

PCR3の各ビットは1,1,1,1,1,1,1,1(2進数)←どのビットも全て1

となります。

ただC言語では2進数を表現できないのでプログラミングでは16進数に直して代入します。(※2)

なので

PCR3=0xff(16進数)

となります。

次にPDR3を設定します。ポート3は今行ったPCR3の設定によって出力ポートとなったのでPDR3に書かれた内容をそのまま出力します。

PDR3でP30~P37それぞれの個別設定をするのにも2進数を用います。

0はL, 1はHを表す事が出来ます。

P30~P37のピンをLやHに設定するにはPDR3のそれぞれに0や1を設定します。

例えば全てのピンをLにするのなら

PDR3の各ビットは0,0,0,0,0,0,0,0(2進数) ←どのビットも全て0

PDR3 = 0x00(16進数)

または全てのピンをHにするのなら

PDR3 の各ビットは 1,1,1,1,1,1,1,1 (2進数) ←どのビットも全て 1

PDR3 = 0xff(16進数)

となります。

L,H,L,H,L,H,L,H と交互にするのなら

PDR3 の各ビットは 0,1,0,1,0,1,0,1

PDR3 = 0x55

となります。

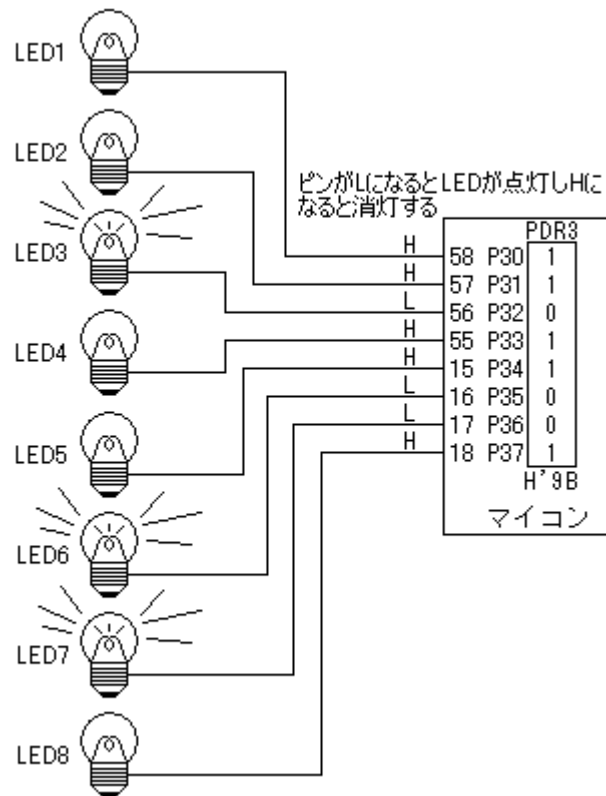


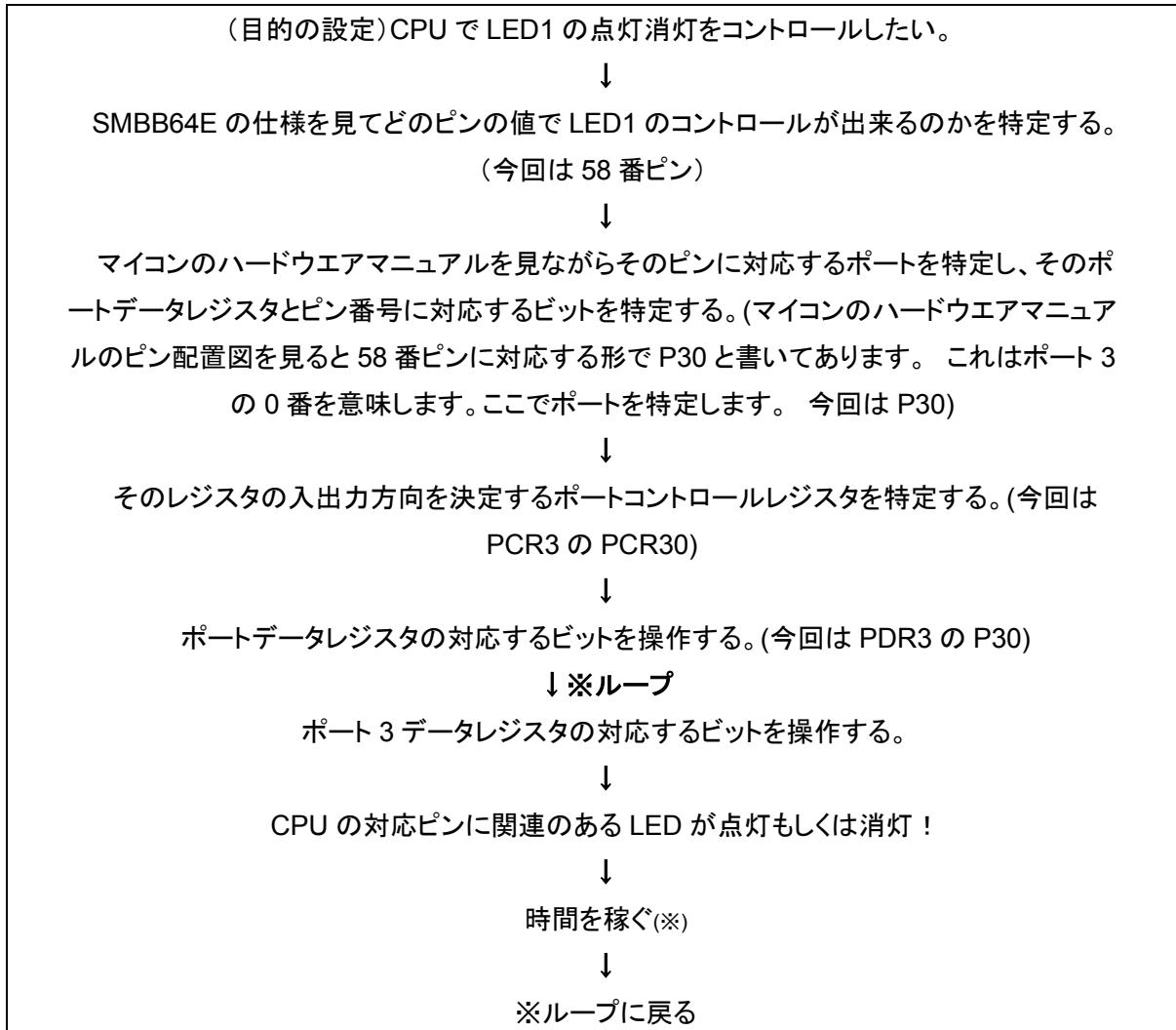
図4. “LEDの状態”と”ピンの状態”と”PDR3の状態”の関係

(※1) ビットとはデジタルコンピュータの扱う最小単位です。8ビットで1バイトを構成します。

(※2) 2進数から16進数の変換は関数電卓があると便利です。

## 2.2.2. LED1 点滅プログラミングの流れ

LED1 点滅プログラム作成における考え方の流れの例を示します。



(※) 時間稼ぎループは無くてもプログラムの点滅自体はするのですが、目にも止まらぬ速さで点滅を繰り返してしまうので実際に点滅しているかどうか判らなくなる為、設けてあります。

本編は製品版付属 CD 内に  
PDF 形式で収録されています。

LED 制御の基礎からモータ制  
御まで解りやすく解説！

続きは製品ご購入後！！