

徹底解剖：デジタル時計 の割込みハンドラ制御

2011年春期 応用情報技術者試験 午後問7（設問2）

— 思考のプロセスをなぞる

LED（6個）

LED 制御信号

クロックカウンタ

MPU

RTC

↑
クロック信号

初期設定データ

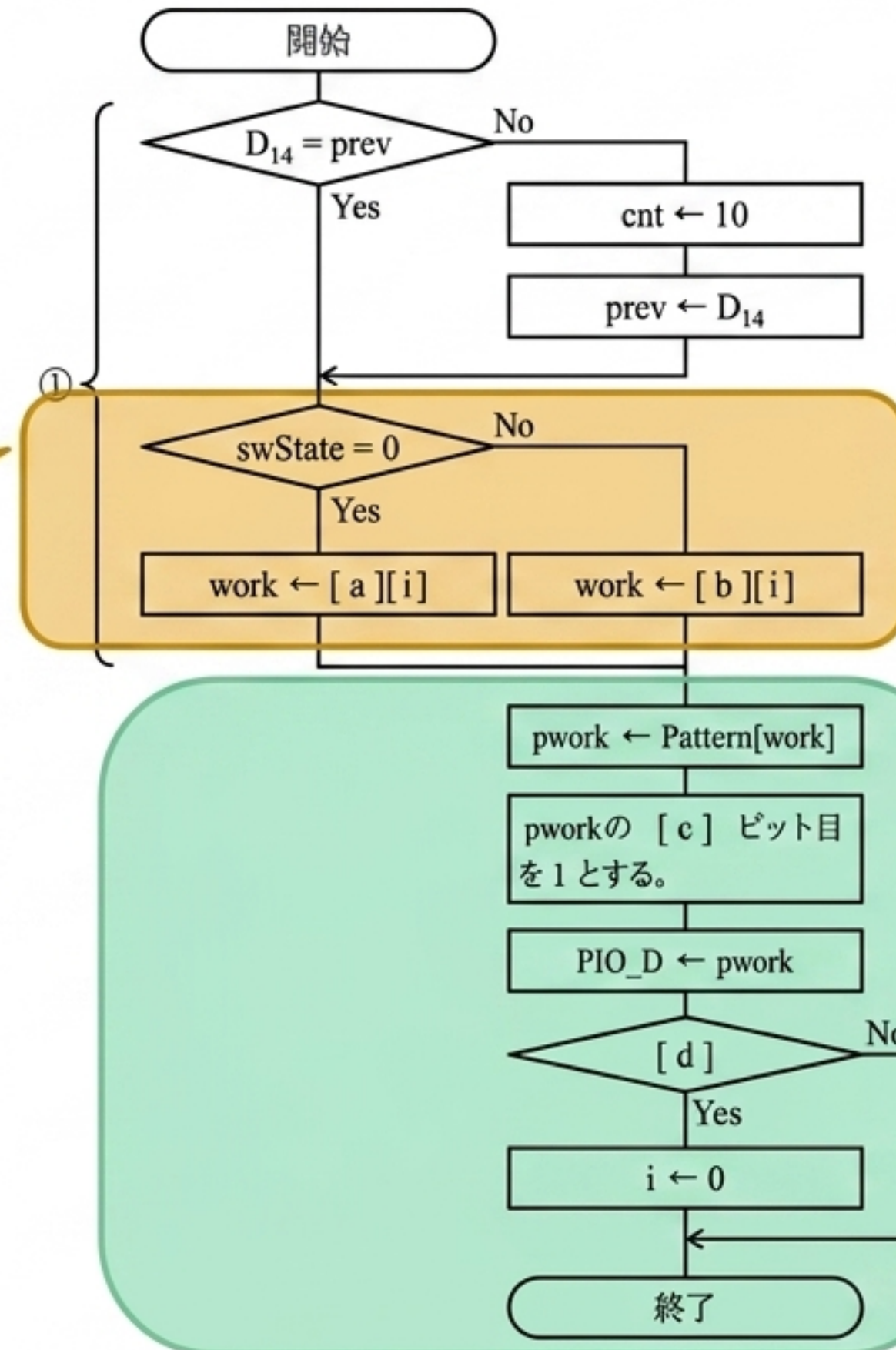
初期設定データ

The Mission: 答えではなく「解き方」を手に入れる

本資料の目的は、クロックカウンタ割込みハンドラの4つの空欄 [a], [b], [c], [d] の論理を読み解くこと。ただし、ここに解答そのものは記載しない。自力でロジックを導き出すための「思考の補助線」を提供する。



ターゲットの俯瞰と分割 (Divide & Conquer)



【前半戦】 $[a][b]$:
表示データの選択
(SW状態と配列の紐付け)

【後半戦】 $[c][d]$:
ダイナミック点灯の制御
(LED選択とループ処理)

手がかり1: スイッチ(SW)の仕様マトリクス

問題文に埋もれた仕様テキストを、構造化されたマトリクスに変換する。

	SWを押している (ON)	SWを押していない (OFF)
端子(D14)の値		
表示させる内容	年月日	時分秒

※ D14端子の値は、変数 swState に連動する

手がかり2: データの格納先（配列）の整理

RTCから取得されたデータは、MPUのメモリ空間（配列）にどのように格納されているか？

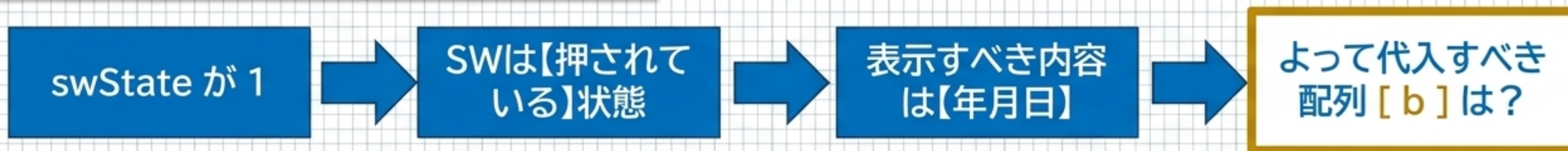
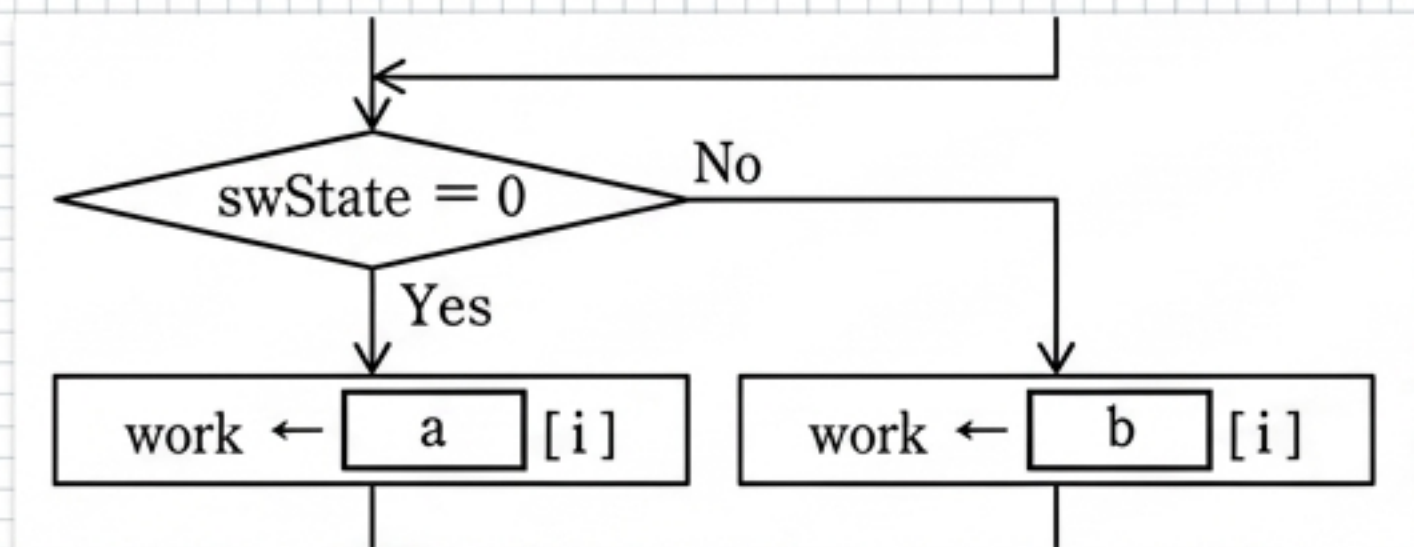
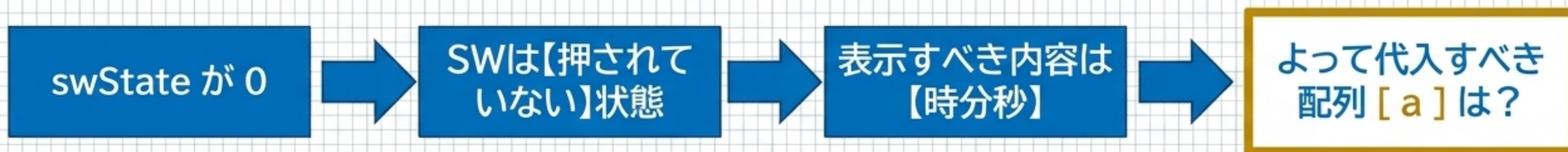
年月日データ

配列 Date	[0]	[1]	[2]	[3]	[4]	[5]
	西暦年	西暦年	月	月	日	日

時分秒データ

配列 Time	[0]	[1]	[2]	[3]	[4]	[5]
	時	時	分	分	秒	秒

推理ボード: [a] と [b] の導出ロジック

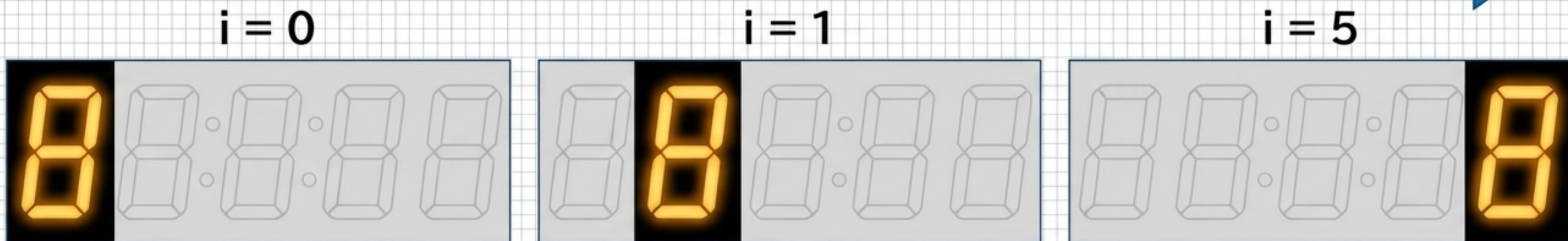


💡 手がかり1と2を繋ぎ合わせれば、答えは自ずと浮かび上がる。

ダイナミック点灯方式とは？（ハードウェアの制約）

すべてのLEDが同時に光っているように見えるが、実は「変数 i 」を使って1桁ずつ順番に高速で点灯させている。

時間の経過(高速切り替え)



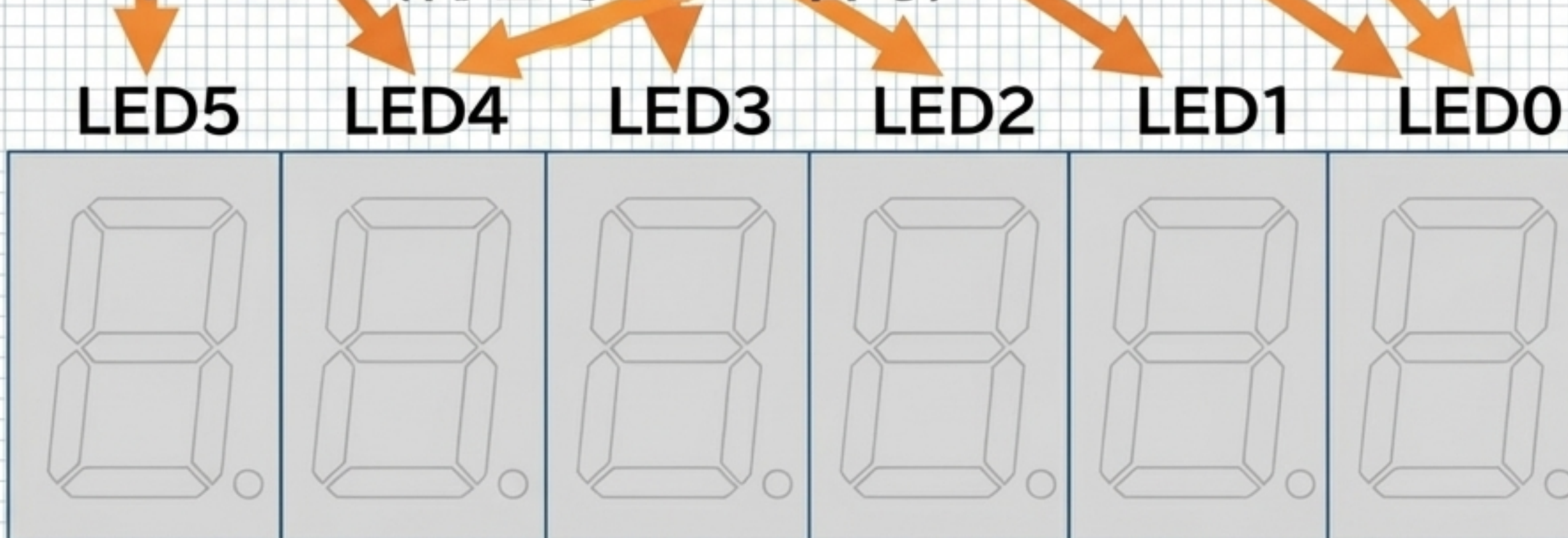
変数 i は 0 からスタートするカウンタ。この値が変化することで、点灯させるLEDの物理的な位置がスライドしていく。

手がかり3: 変数 i とLED番号の対応

ソフトウェア (配列の添字 i)



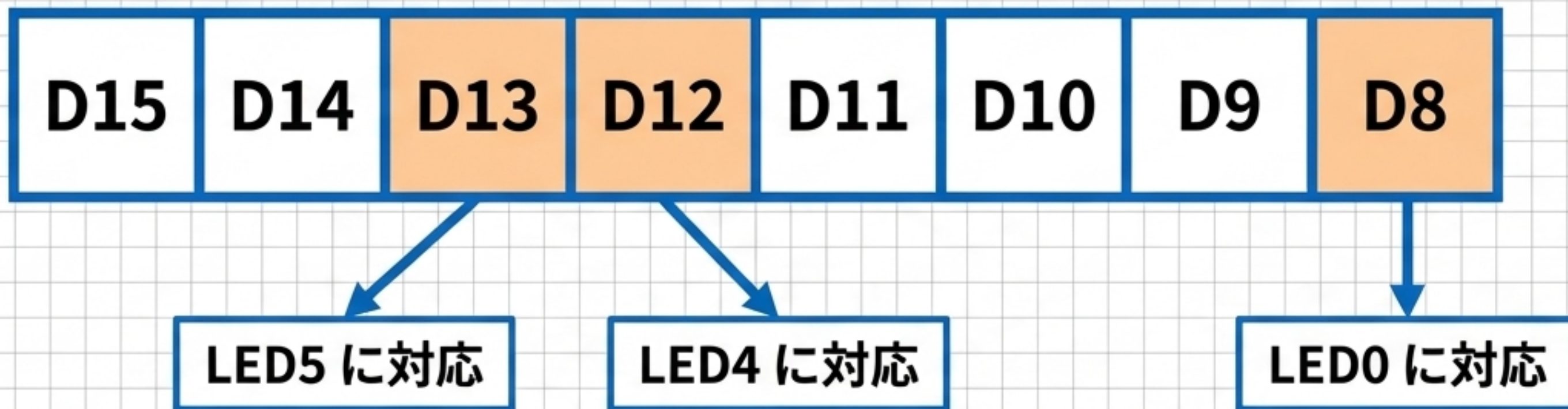
ハードウェア (物理的なLED番号)



反比例の関係:
配列の添字 i が増えるほど、対象となるLED番号は減っていく。

手がかり4: LED番号と制御ビット(PIO_D)の対応

特定のLEDを光らせるためには、ハードウェアの制御レジスタ (PIO_D) の対応するビットを「1」にする必要がある。



つまり、対象とするLED番号に対して、操作すべき『ビット位置』が決まっている。

推理ボード: [c] の導出口ジック

規則性発見テーブル

変数 i	対象LED	制御ビット位置	隠れた規則性 (i + ビット位置)
0	LED5	13	$0 + 13 = 13$
1	LED4	12	$1 + 12 = 13$
...	...	...	...
5	LED0	8	$5 + 8 = 13$



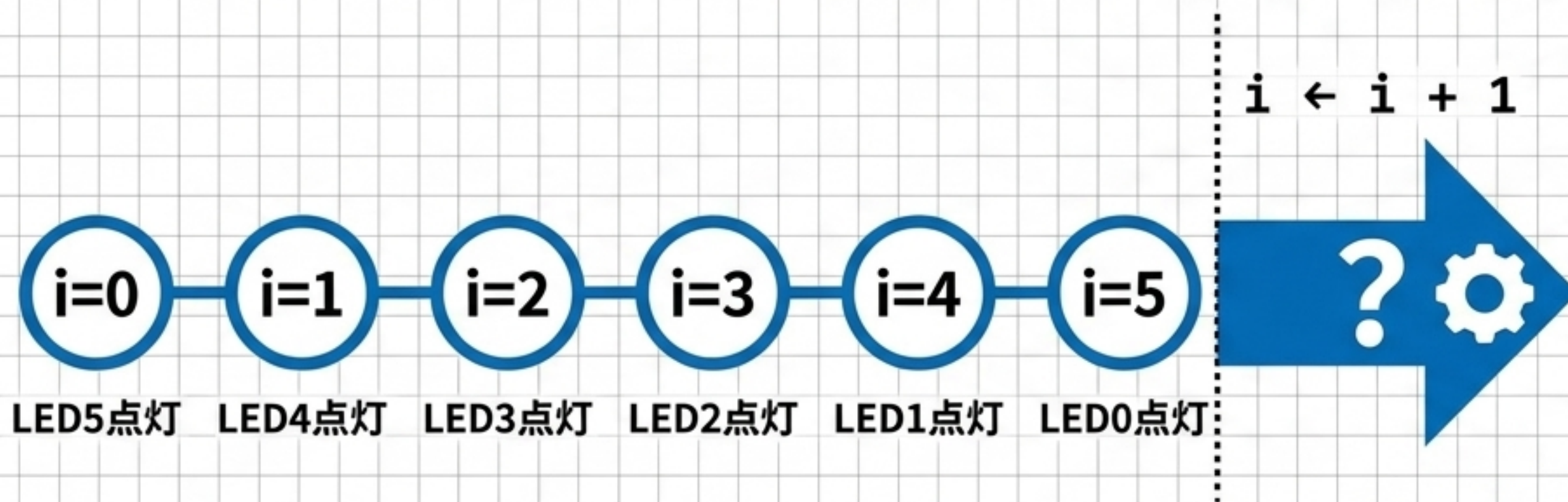
問いかけ：

変数 i を使って、目的の「制御ビット位置 (13~8)」
を導き出す数式はどうなるか？

※足して常に同じ数「13」
になる関係に注目。
数式は「13 と i」を使って
どう表現できる? ($= [c]$)

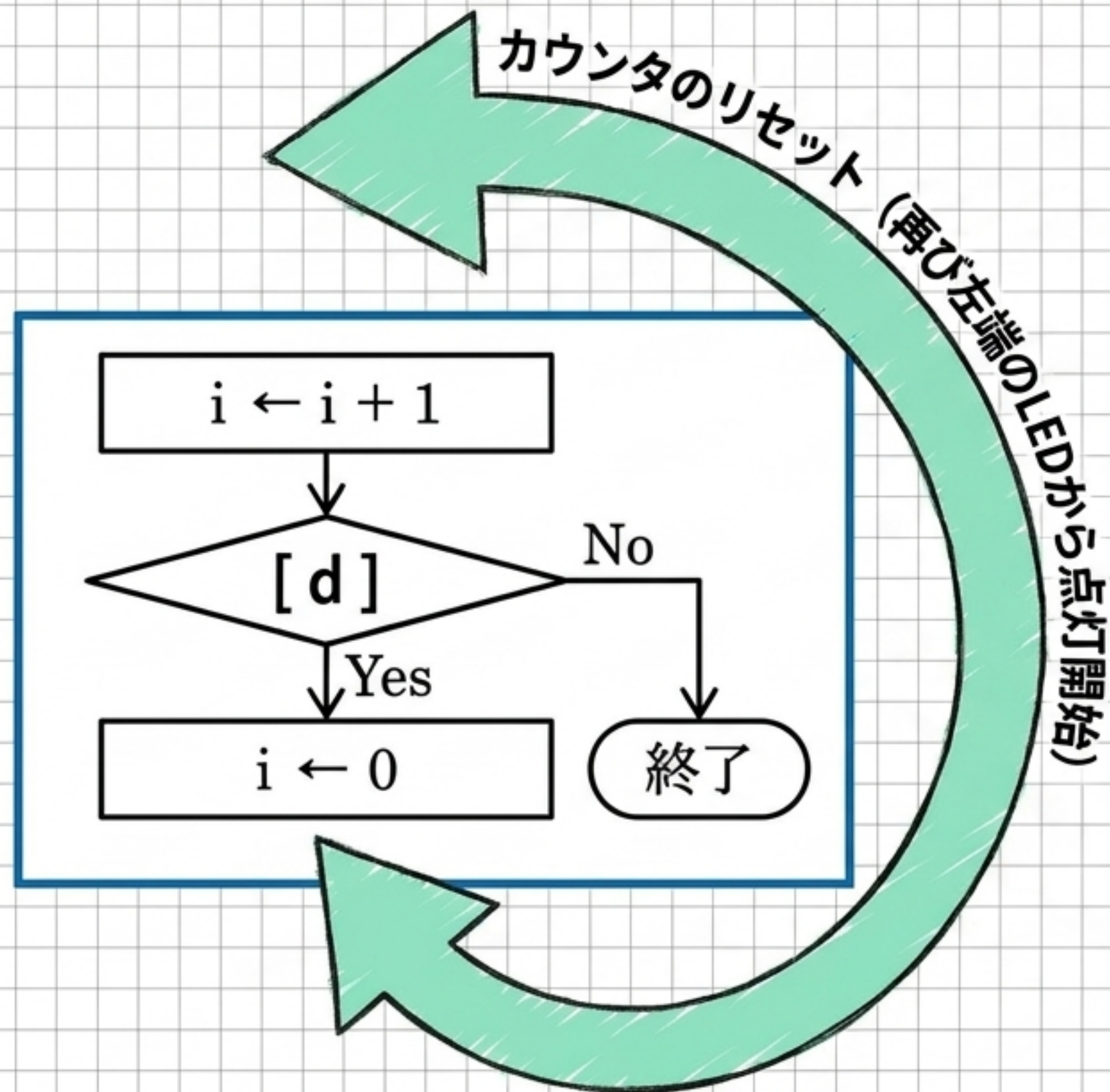
手がかり5: カウンタ i のライフサイクル

ダイナミック点灯は途切れることなく永遠にループしなければならない。
すべてのLEDを表示し終えた直後に何が起きるか？



全てのLED (6個) を
光らせ終わり、
最後の $i=5$ の処理が
完了した直後、 フ
ローチャートにより
 i に $+1$ が加算される。
この瞬間の i の値は
【いくつ】 になって
いるか？

推理ボード: [d] の導出口ジック



ループを永遠に繰り返すための条件

- 6個のLEDを表示し終えた瞬間に、カウンタを 0 に戻す必要がある。
- 『 $i \leftarrow i + 1$ 』の直後、リセット ($i \leftarrow 0$) を発動させるべき変数 i の値は何か？

つまり、 $[d]$ に入るべき判定条件式は？

統合とまとめ (Synthesis: Blueprint Completed)

1

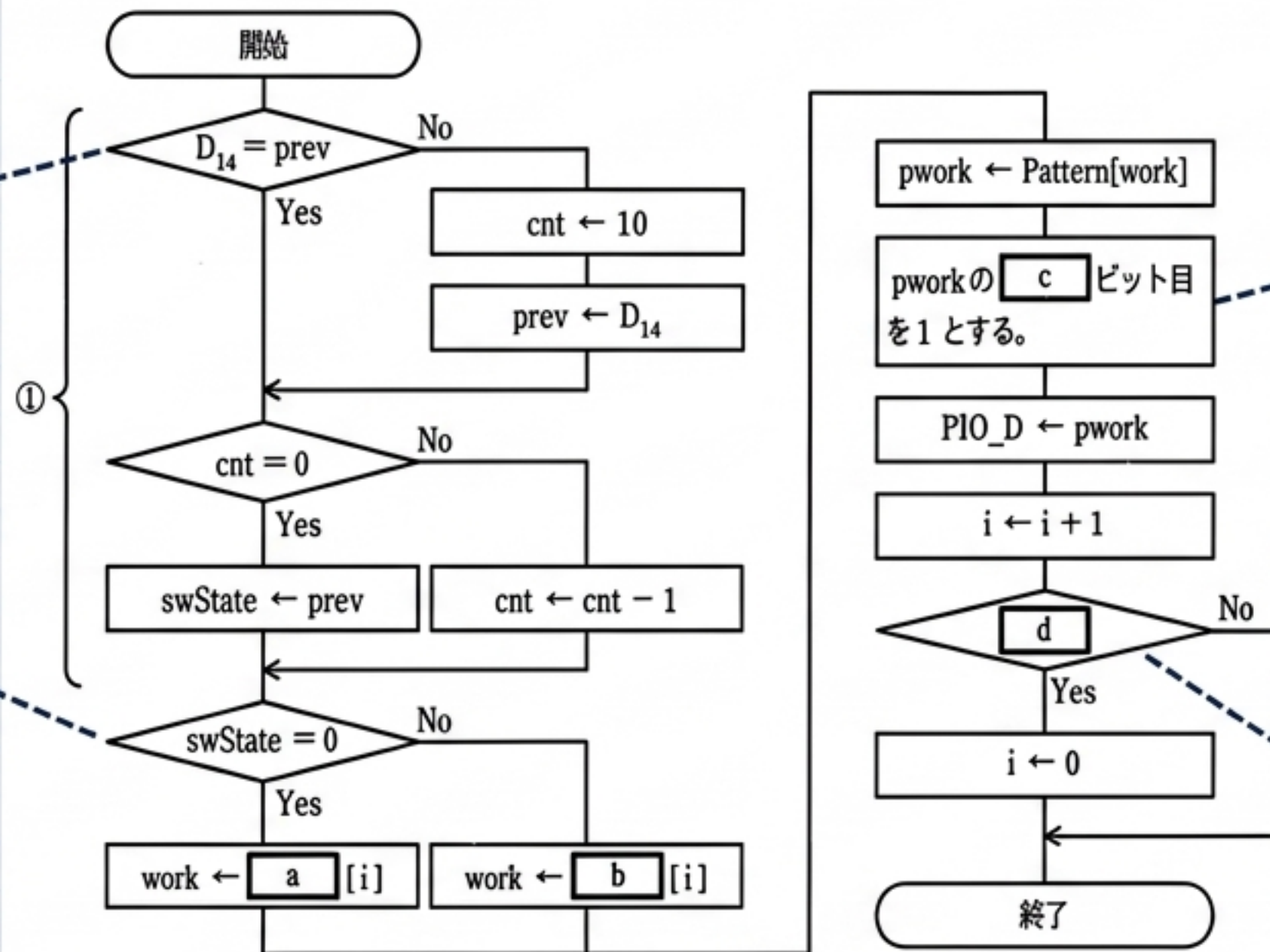
物理現象の吸収

チャタリング（物理接点のノイズ）をソフトウェアの待機ループで排除し、安定状態を担保。

2

ハードとデータの結節点

SWの物理的状态（ON/OFF）を判定し、メモリ上の対象配列（Date/Time）を切り替える。[a, b]



3

論理から物理番地への変換

ループ変数 i の規則性を利用して、レジスタの特定ビット位置を数式でピンポイント操作。[c]

4

無限のハードウェア駆動

カウンタのライフサイクルを監視し、表示が一巡した瞬間にリセットをかけ、ダイナミック点灯を永遠に持続させる。[d]

図5 クロックカウンタ割込みハンドラの流れ図

答えを暗記するのではなく、仕様書に記された「物理的な制約」をプログラムの「変数と条件式」へと翻訳する力。それこそが、組込みシステム設計の本質である。