

ワイヤレス充電ステーション制御システムの技術解析および試験復習レポート：平成26年度 春期 応用情報技術者試験 午後 問7

1. はじめに：本システムの概要と学習の戦略的意義

近年、スマートフォンをはじめとする携帯機器において、利便性の向上を目的としたワイヤレス充電技術が広く普及しています。本レポートで扱うG社のワイヤレス充電ステーションは、送電・受電コイル間の電磁誘導を利用し、近距離無線通信によって機器情報をやり取りする、現代の組込みシステムにおける「制御と安全設計」の粋を集めたモデルケースです。本レポートの目的は、単なる試験問題の解答確認に留まりません。組込みソフトウェアにおけるマルチタスクの優先度設計、状態遷移に基づく正確な処理フロー、そして物理的リスクを論理的に回避する安全設計の思考プロセスを深く掘り下げ、アーキテクトとしての視点を養うことにあります。まずは、システムの挙動を規定する動作状態と、制御の判断材料となるデータ構造について整理します。

2. ステーションの動作状態と携帯機器情報のデータ構造

組込みシステムの堅牢性は、システムの状態（State）と管理データ（Data Structure）の厳密な定義から生まれます。

動作状態の管理（表1）

ステーションは以下の4つの状態を遷移し、ユーザーに対して適切なフィードバックを行います。

- **待機状態**：機器が置かれるのを待つ状態。表示は **"STANDBY"**。
- **充電動作状態**：送電中の状態。表示は **残り充電時間**（hh:mm形式）。
- **充電停止状態**：満充電完了後の状態。表示は **"FULL"**。
- **異常検出状態**：異常を検知しシステムを安全に停止させた状態。表示は **"ERROR"**。

携帯機器情報のデータ構造（表2）

通信によって取得される各項目は、制御フローにおいて極めて重要な役割を果たします。特に「機器ID」は、認識できない場合に「ゼロ」が設定されるという仕様が、待機判定のトリガーとなります。

設計上の考察：残り充電時間の算出仮定

本システムでは「電池残量の増分が充電時間に比例する」という線形近似を採用しています。実務上のリチウムイオン電池の充電特性は、定電流（CC）から定電圧（CV）へと移行する非線形なものですが、リソースの限られた組込み機器のUI表示において、直近の推移から概算値を算出するこの手法は、計算負荷とユーザーの利便性を天秤にかけた合理的な設計判断といえます。

3. 組込みソフトウェアのアーキテクチャとタスク優先度設計

リアルタイム OS (RTOS) 環境において、システムの応答性と安全性を担保するのがタスク優先度の設計です。

優先度の論理的検証 (表3)

本システムのタスク優先度は「1 (高) ~4 (低)」の4段階で、各タスクは異なる優先度を持ちます。ソースコンテキストによると、すでに「異常検出状態」タスクには 1、「通信」タスクには 3 が割り当てられています。したがって、メインタスク (a) と安全監視タスク (b) に割り当て可能な値は {2, 4} のいずれかとなります。ここで優先度を決定する論理は以下の通りです。

- **安全監視タスク (b=2)** : 500ミリ秒ごとに起動される周期タスク。物理的な異常 (発熱等) を検知するため、確実に実行される必要があります。
- **メインタスク (a=4)** : ステーション全体の動作を制御し、基本的には常に稼働し続けるタスク。もしメインタスクの優先度を高く (数値を小さく) 設定してしまうと、メインタスクの実行中に安全監視タスクが割り込めず、異常検知に致命的な遅れが生じるリスクがあります。短時間で処理を終えて待機に入る周期タスク (安全監視) に高い優先度を与えるのは、組込み設計の鉄則です。

関数とリソースのマッピング (表4)

送電制御を行う各関数は、以下のタスクから実行されます。

- `Start_Power()` / `Stop_Power()` : 主に **メインタスク** から呼び出され、送電状態を制御します。
- `Get_Watt()` : **安全監視タスク** から呼び出され、送電効率の算出に用いられます。

4. メインタスクの処理フローと条件判定の論理

図2のフローチャートは、システムの安定稼働を支える論理構成図です。

条件①の特定 : 機器識別

フローチャート冒頭の「条件①」は、待機状態を継続するか否かの分岐点です。表2において「充電テーブル上の携帯機器が認識できない場合は、ゼロが設定される」と定義されていることから、条件①は**「機器IDがゼロ」**となります。

10秒間のウェイトと異常検出の安定性

フロー内で `Start_Power()` 実行直後に「10秒間待つ」ステップが挿入されています。これは送電開始直後の過渡的な電力変動を避け、受電電力および送電電力 (`Get_Watt` で取得) の値が安定するのを待つための設計です。この待機時間を設けることで、後の「②異常検出」ステップにおける送電効率の計算精度を高め、異物混入等の誤検知や検知漏れを防いでいます。

5. 定量的解析 : 残り充電時間と送電効率の算出

精緻なモニタリングは、ユーザー体験の向上とシステムの安全確保を両立させます。

残り充電時間の計算演習（設問1）

1. **増加率**：10分間で 43.0% から 49.0% へ変化（+6.0%）。1分あたり **0.6%** の増加。
2. **必要量**：満充電（100%）まで残り **51.0%**（100 - 49.0）。
3. **計算**： $51.0 \div 0.6 = 85$ 分。
4. **表示形式**：85分は1時間25分であるため、表示は **"01:25"**。

送電効率のモニタリング（設問4）

異常検出の指標となる送電効率は、以下の計算式で算出されます。

$$\text{送電効率 (\%)} = \frac{\text{受電電力}}{\text{送電電力}} \times 100$$

ここで重要なのは、データの取得元が異なる点です。

- **受電電力**：通信タスクを介して携帯機器から取得した「携帯機器情報」内の値。
- **送電電力**：ステーション自身が計測する値であり、ソフトウェア処理として **Get_Watt** 関数を呼び出すことで取得します。ステーション側で自らの出力を把握し、相手側の受電量と比較することが効率監視の本質です。

6. 安全設計：異常発熱および金属異物の検出メカニズム

組込みシステムにおける物理的リスクの回避は、ソフトウェアアーキテクトの最重要責務の一つです。

多層的な異常検知ロジック

本システムは、性質の異なる2つの監視を組み合わせた多層防御を構築しています。

1. **直接的監視（c: 温度センサ）**：安全監視タスクが充電テーブルの **「温度」** を直接読み込み、異常発熱を検知します。
2. **論理的・間接的監視（送電効率）**：金属異物がセンサーに直接触れていない場合、温度上昇の検知が遅れる可能性があります。そこで、供給エネルギーと受け取りエネルギーの差異（効率低下）を論理的に監視し、異物への不要な給電を早期に発見します。

異常発生時の挙動（d: Stop_Power）

異常を検知した場合、システムは直ちに **Stop_Power** 関数を呼び出して送電を遮断します。その後、**異常検出状態** タスクが起動し、表示部に **"ERROR"** を出力した上で「無限ループ」へと遷移します。このフェイルセーフ設計により、ハードウェア的なリセット（主電源の再投入）がない限り動作を再開させず、二次被害を防止します。

7. まとめ：本問題から学ぶ組込みシステム設計の教訓

本問題の解析を通じて、実務に直結する3つの設計思想が浮き彫りになりました。

1. **優先度の必然性**：「周期性」と「緊急性」を持つタスクを、「定常的」なタスクより上位に配置する論理的根拠。

2. **近似とUI設計**：物理特性（電池の充電曲線）を理解した上での、合理的な簡略化手法の選択。
3. **多角的な安全網**：物理センサー（温度）と論理推論（効率）を組み合わせた、死角のない異常検知システムの構築。応用情報技術者試験の仕様記述を読み解く力は、そのまま実務における「曖昧さのない仕様書」を解読し、堅牢なアーキテクチャを構築する力へと繋がります。本レポートで整理した思考プロセスは、あらゆる組み込みシステム開発の現場で通用する普遍的な知見と言えるでしょう。