8-1-基 組み込みシステム開発に関する知識

1. 科目の概要

組み込みシステムの基本的な構造と活用方法、組み込みシステムを構成するハードウェア、ソフトウェア、OS など様々な要素の役割や特徴について解説する。

2. 習得ポイント

本科目の学習により習得することが期待されるポイントは以下の通り。

習得ポイント	説 明	シラバスの対応コマ
8-1-基-1.組み込みコンピュータシステム	組み込みコンピュータシステムとは何か、基本的な概念を説明する。組み込みコンピュータシステムにおけるハードウェア、ソフトウェア、ネットワークの全体的なアーキテクチャを解説する。	1
8-1-基-2.組み込みシステムとOSSの関係、活用事例	組み込みシステムにおける数多くのOSS活用事例を紹介し、組み込みシステムの構築にOSSの活用が有効であることを説明する。また組み込みシステムにおけるOSS活用時の留意点についても解説する。	2
8-1-基-3.組み込みコンピュータの要件	組み込みシステム開発におけるエミュレータの活用方法について紹介する。エミュレータを利用することで専用ハードウェアがない状況でも実行やデバッグができることを解説する。	3,4,5,6
8-1-基-4.組み込み向けソフトウェアの基本	組み込み向けソフトウェアの役割と特徴を説明する。組み込み向けソフトウェア処理の基本として、スレッドとスケジューリングを説明し、並行処理のアーキテクチャに関する基本的な方式とスレッド管理方法などに関する話題にも触れる。	7
8-1-基-5.コンテキストスイッチの仕組み	複数の処理（スレッド）を実行するコンテキストスイッチの仕組みを解説する。OSが処理の切り替えを管理するプリエンプション方式の実装方法を説明し、割り込みによるコンテキストスイッチにも触れる。	8
8-1-基-6.非同期処理と同期処理の実装パターンと特徴	非同期/同期の概念について触れ、その実装パターンと設計について内容と特徴を説明する。また実行モードの種類や割り込みの優先順位といった具体的な事項について説明する。	9
8-1-基-7.タスク優先順位制御とカーネルによる時間管理方法	スレッドの優先度とその制御仕様について、実装パターンと設計内容、特徴、手順を説明する。実際のスレッド制御を行うカーネルが時間制御方法としてどのような管理手法を用いているかについても触れる。	10
8-1-基-8.割り込みとDMA	組み込みシステムには欠かせない割り込みとDMA(Direct Memory Access)について解説する。割り込みの仕組みとそれによるOSの動作やDMAの仕組みやメリットを紹介する。	11
8-1-基-9.システムリソースの配分の仕組み(セマフォ、キュー、等)	組み込みシステムにおいては、システムリソースが限定されているため様々な工夫が必要になる。各アプリケーションが限られたシステム資源を効率的に共有する方法について、セマフォ、キュー、といった具体的な実装方法を踏まえて、その内容と特徴を説明する。	12,13
8-1-基-10.システムリソースの共有の仕組み(共有ファイル)	複数の組み込みアプリケーション同士でシステムリソースを共有する方法を、共有エリア・共有ファイル、カーネルによる各種のサービス、デッドロックの回避、割り込みディスパッチなどの具体的な実装を挙げて説明する。	14

1

3. IT 知識体系との対応関係

「8-1-基 組み込みシステム開発に関する知識」と IT 知識体系との対応関係は以下の通り。

科目名	基本レベル													
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
8-1-基 組み込みシステムに関する知識	組み込みコンピュータシステムとは何か	組み込みコンピュータのアーキテクチャ	組み込みシステムの基本構成	組み込みコンピュータハードウェアの基本	CPUアーキテクチャの基本	組み込みソフトウェアの概要	カーネル処理の基本	タスクとコンテキスト	非同期と同期の設計仕様	タスクの優先度とその制御仕様	割り込みとDMA	組み込みアプリケーション間の資源配分技術(セマフォ)	組み込みアプリケーション間の資源配分技術(キュー)	組み込みアプリケーション間のリソース共有技術

<IT 知識体系上の関連部分>

分野		科目名	1	2	3	4	5	6	7	8	9	10	11	12	13
組織関連事項と情報システム	1	IT-IAS 情報保証と情報セキュリティ	IT-IAS1.基礎的な問題	IT-IAS2.情報セキュリティの仕組み(対策)	IT-IAS3.運用上の問題	IT-IAS4.ポリシー	IT-IAS5.攻撃	IT-IAS6.情報セキュリティ分野	IT-IAS7.フォレンジック(情報証拠)	IT-IAS8.情報の状態	IT-IAS9.情報セキュリティサービス	IT-IAS10.脅威分析モデル	IT-IAS11.脆弱性		
	2	IT-SP 社会的な観点とプロフェッショナルとしての課題	IT-SP1.プロフェッショナルとしてのコミュニケーション	IT-SP2.コンピュータの歴史	IT-SP3.コンピュータを取り巻く社会環境	IT-SP4.チームワーク	IT-SP5.知的財産権	IT-SP6.コンピュータの法的問題	IT-SP7.組織の中のIT	IT-SP8.プロフェッショナルとしての倫理的な問題と責任	IT-SP9.プライバシーと個人の自由				
応用技術	3	IT-IM 情報管理	IT-IM1.情報管理の概念と基礎	IT-IM2.データベース問合わせ言語	IT-IM3.データアーキテクチャ	IT-IM4.データモデリングとデータベース設計	IT-IM5.データと情報の管理	IT-IM6.データベースの応用分野							
	4	IT-WS Webシステムとその技術	IT-WS1.Web技術 [1-1-7]	IT-WS2.情報アーキテクチャ [1-1-7]	IT-WS3.デジタルメディア	IT-WS4.Web開発	IT-WS5.脆弱性	IT-WS6.ソーシャルソフトウェア							
ソフトウェアの方法と技術	5	IT-PF プログラミング基礎	IT-PF1.基本データ構造	IT-PF2.プログラミングの基本的構成要素	IT-PF3.オブジェクト指向プログラミング	IT-PF4.アルゴリズムと問題解決	IT-PF5.イベント駆動プログラミング	IT-PF6.再帰							
	6	IT-IPT 技術を統合するためのプログラミング	IT-IPT1.システム間通信 [1-1-3]	IT-IPT2.データ割り当てと交換	IT-IPT3.統合的コーディング	IT-IPT4.スクリプティング手法	IT-IPT5.ソフトウェアセキュリティの実現	IT-IPT6.種々の問題	IT-IPT7.プログラミング言語の概要						
	7	CE-SWE ソフトウェア工学	CE-SWE0.歴史と概要	CE-SWE1.ソフトウェアプロセス	CE-SWE2.ソフトウェアの要求と仕様	CE-SWE3.ソフトウェアの設計	CE-SWE4.ソフトウェアのテストと検証	CE-SWE5.ソフトウェアの保守	CE-SWE6.ソフトウェア開発・保守ツールと環境 [1-1-4]	CE-SWE7.ソフトウェアプロジェクト管理	CE-SWE8.言語翻訳	CE-SWE9.ソフトウェアのフォールトトレランス	CE-SWE10.ソフトウェアの構成管理	CE-SWE11.ソフトェアの標準化 [1-1-6]	
	8	IT-SIA システムインテグレーションとアーキテクチャ	IT-SIA1.要求仕様	IT-SIA2.調達/手配	IT-SIA3.インテグレーション [1-1-4]	IT-SIA4.プロジェクト管理	IT-SIA5.テストと品質保証	IT-SIA6.組織の特性	IT-SIA7.アーキテクチャ						
システム基盤	9	IT-NET ネットワーク	IT-NET1.ネットワークの基礎	IT-NET2.ルーティングとスイッチング	IT-NET3.物理層	IT-NET4.セキュリティ	IT-NET5.アプリケーション分野 [1-1-5]	IT-NET6.ネットワーク管理							
	10	CE-NWK テレコミュニケーション	CE-NWK0.歴史と概要	CE-NWK1.通信ネットワークのアーキテクチャ	CE-NWK2.通信ネットワークのプロトコル	CE-NWK3.LANとWAN	CE-NWK4.クライアントサーバコンピューティング [1-1-3]	CE-NWK5.データのセキュリティと整合性	CE-NWK6.ワイヤレスコンピューティングとモバイルコンピューティング	CE-NWK7.データ通信	CE-NWK8.組込み機器向けネットワーク	CE-NWK9.通信技術とネットワーク概要	CE-NWK10.性能評価	CE-NWK11.ネットワーク管理	CE-NWK12.圧縮と伸張
			CE-NWK13.クラスタシステム	CE-NWK14.インターネットアプリケーション [1-1-5,7]	CE-NWK15.次世代インターネット	CE-NWK16.放送									
	11	IT-PT プラットフォーム技術	IT-PT1.オペレーティングシステム [1-1-3]	IT-PT2.アーキテクチャと機構	IT-PT3.コンピュータインフラストラクチャ	IT-PT4.デプロイメントソフトウェア [1-1-4]	IT-PT5.ファームウェア	IT-PT6.ハードウェア							
	12	CE-OPS オペレーティングシステム	CE-OPS0.歴史と概要	CE-OPS1.並行性	CE-OPS2.スケジューリングとディスパッチ	CE-OPS3.メモリ管理	CE-OPS4.セキュリティと保護	CE-OPS5.ファイル管理	CE-OPS6.リアルタイムOS	CE-OPS7.OSの概要	CE-OPS8.設計の原則	CE-OPS9.デバイス管理	CE-OPS10.システム性能評価		
コンピュータハードウェアとアーキテクチャ	13	CE-CAO コンピュータのアーキテクチャと構成	CE-CAO0.歴史と概要	CE-CAO1.コンピュータアーキテクチャの基礎	CE-CAO2.メモリシステムの構成とアーキテクチャ	CE-CAO3.インタフェースと通信	CE-CAO4.デバイスサブシステム	CE-CAO5.CPUアーキテクチャ	CE-CAO6.性能・コスト評価	CE-CAO7.分散・並列処理	CE-CAO8.コンピュータによる計算	CE-CAO9.性能向上			
複数領域にまたがるもの	14	IT-ITF IT基礎	IT-ITF1.ITの一般的なテーマ [1-1-4]	IT-ITF2.組織の問題	IT-ITF3.ITの歴史	IT-ITF4.IT分野(学科)とそれに関連のある分野(学科)	IT-ITF5.応用領域	IT-ITF6.IT分野における数学と統計学の活用							
	15	CE-ESY 組込みシステム	CE-ESY0.歴史と概要	CE-ESY1.低電力コンピューティング	CE-ESY2.高信頼性システムの設計	CE-ESY3.組込み用アーキテクチャ	CE-ESY4.開発環境	CE-ESY5.ライフサイクル	CE-ESY6.要件分析	CE-ESY7.仕様定義	CE-ESY8.構造設計	CE-ESY9.テスト	CE-ESY10.プロジェクト管理	CE-ESY11.並行設計(ハードウェア、ソフトウェア	CE-ESY12.実装
			CE-ESY13.リアルタイムシステム設計	CE-ESY14.組込みマイクロコントローラ	CE-ESY15.組込みプログラム	CE-ESY16.設計手法	CE-ESY17.ツールによるサポート	CE-ESY18.ネットワーク型組込みシステム	CE-ESY19.インタフェースシステムと混合信号システム	CE-ESY20.センサ技術	CE-ESY21.デバイスドライバ	CE-ESY22.メンテナンス	CE-ESY23.専門システム	CE-ESY24.信頼性とフォールトトレランス	

4. OSS モデルカリキュラム固有の知識

OSS モデルカリキュラム固有の知識は、OSS を用いた組み込みシステム開発に関する知識である。Linux を活用した OS の基本動作などを扱う。

科目名	第1回	第2回	第3回	第4回	第5回	第6回	第7回
8-1-基 組み込みシステムに関する知識	(1) 組み込みコンピュータシステムとは (2) システム (3) 組み込みコンピュータシステムとOSSの関係	(1) 組み込みコンピュータハードウェアの特徴 (2) 組み込みコンピュータアーキテクチャの必要要件 (3) OSSを活用した組み込みコンピュータアーキテクチャの事例	(1) 組み込みシステムのCPU (2) OSの要件 (3) 組み込みシステムのネットワーク形態 (4) 低消費電力	(1) 組み込みハードウェアアーキテクチャの概要	(1) MPU内部のアーキテクチャ (2) 命令セットアーキテクチャ (3) メモリとMPUの接続 (4) 組み込みプロセッサアーキテクチャ	(1) ソフトウェア処理の基本 (2) 並行処理のアーキテクチャ (2) 並行処理OSとは	(1) 組み込みカーネルとは (2) カーネル処理の概要

第8回	第9回	第10回	第11回	第12回	第13回	第14回
(1) コンピュータアーキテクチャとは (2) カーネルコンテキスト	(1) 非同期と同期の設計 (2) タスクの構成資源 (3) 実行環境	(1) タスクの優先順位 (2) 資源の管理 (3) カーネルによる時間管理	(1) 割り込み (2) DMA	(1) 資源配分技術 (2) カーネルの提供する手段を使う方法	(1) キューの概要 (2) キューによるタスクの同期	(1) 共有エリア・共有ファイル (2) カーネルによるサービス (3) デッドロック (4) 割り込みスケジューリング（ディスパッチング）

（網掛け部分は IT 知識体系で学習できる知識を示し、それ以外は OSS モデルカリキュラム固有の知識を示している）

スキル区分	OSS モデルカリキュラムの科目	レベル
開発体系分野	8-1-基　組み込みシステム開発に関する知識	基本
習得ポイント	8-1-基-1. 組み込みコンピュータシステム	
対応する コースウェア	第1回 組み込みコンピュータシステムとは何か	

8-1-基-1. 組み込みコンピュータシステム

組み込みコンピュータシステムとは何か、基本的な概念を説明する。組み込みコンピュータシステムにおけるハードウェア、ソフトウェア、ネットワークの全体的なアーキテクチャを解説する。

【学習の要点】

* 組み込みコンピュータシステムは、コンピュータに処理を行わせることで、対象としている機器の利便性を高めるものである。
* 組み込みコンピュータシステムの例として、飛行機に搭載される各種機器、携帯電話、オーディオ機器が挙げられる。今後、組み込みコンピュータシステムの活用される領域はさらに広がることが予想される。
* 組み込みコンピュータシステムは、ボード（基盤）上に CPU、入出力デバイスなどのコンピュータ部品を配置しているハードウェア部分と、コンピュータシステムを制御するソフトウェア部分からなる。

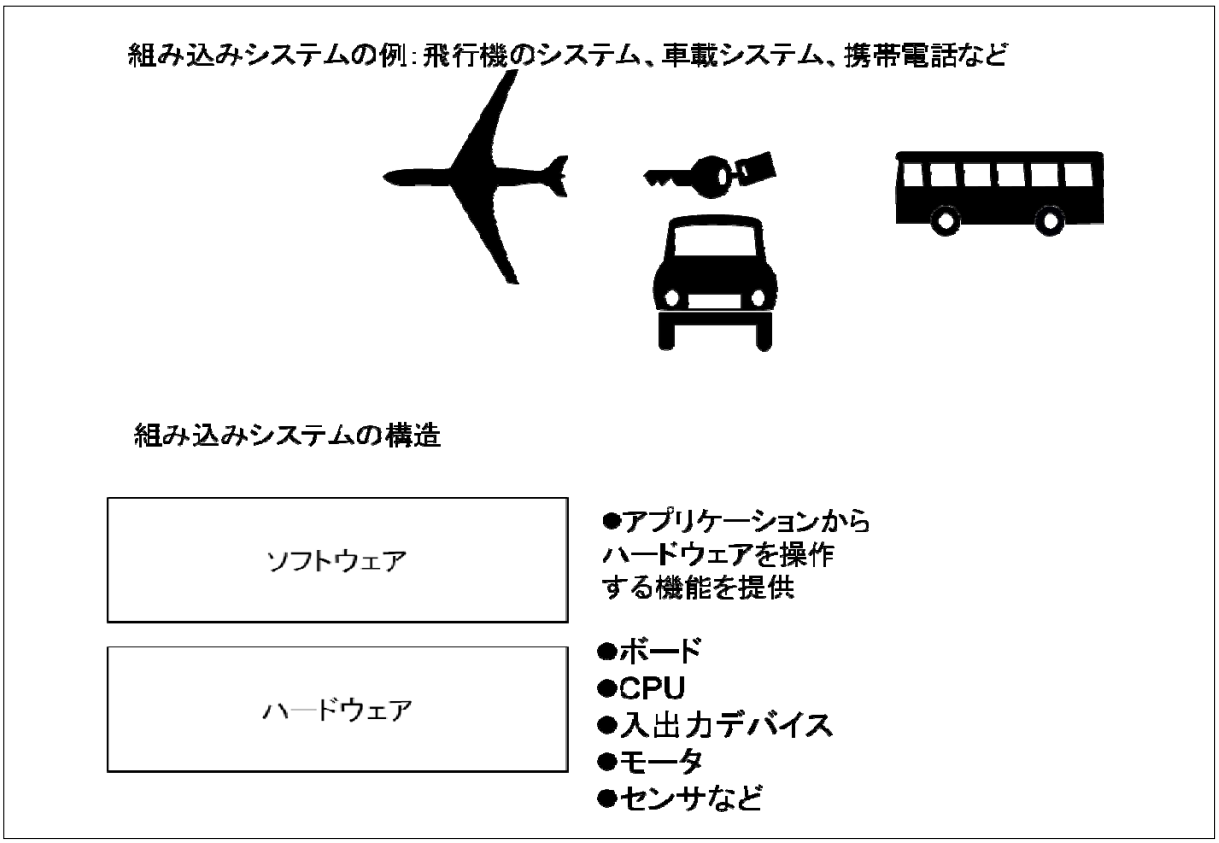

図 8-1-基- 1 組み込みシステムの例と構造

【解説】

1）組み込みコンピュータシステムとは

＊ 組み込みコンピュータシステムは、コンピュータに処理を行わせることで、対象としている機器の利便性を高めるものである。

＊ 組み込みコンピュータシステムの活用例
人工衛星、飛行機、自動車のような航空・運輸機器、携帯電話、テレビ、オーディオ機器のような家電機器に組み込まれるシステムなど、幅広い適用領域がある。

＊ 組み込みコンピュータシステムのアーキテクチャ

- 組み込みコンピュータシステムは、ハードウェア部分とソフトウェア部分からなる。
- 組み込みコンピュータシステムにおけるハードウェア部分は、組み込みシステムを制御する機能を持ち、ボード（基盤）上に、CPU、入出力デバイスなどのコンピュータ部品を配置している。
- 組み込みコンピュータシステムにおけるソフトウェア部分は、主にコンピュータシステムを制御し、アプリケーションからハードウェアを制御するための仕組みを提供する。

スキル区分	OSS モデルカリキュラムの科目	レベル
開発体系分野	8-1-基　組み込みシステム開発に関する知識	基本
習得ポイント	8-1-基-2. 組み込みシステムと OSS の関係、活用事例	
対応する コースウェア	第2回 組み込みコンピュータのアーキテクチャ	

8-1-基-2. 組み込みシステムと OSS の関係、活用事例

組み込みシステムにおける数多くの OSS 活用事例を紹介し、組み込みシステムの構築に OSS の活用が有効であることを説明する。また組み込みシステムにおける OSS 活用時の留意点についても解説する。

【学習の要点】

＊ 組み込みシステムにおいて、数多くの OSS が活用されている事例が存在する。

＊ オープンソース組み込み OS として、Linux、T-Kernel、eCos、TOPPERS/OSEK が著名であり、携帯電話や薄型テレビなどでの実績を多く持つ。また組み込み DB として Oracle Berkeley DB、SQLite などが利用できる。

＊ OSS を活用した組み込みコンピュータアーキテクチャの事例として、ユビキタスセンサシステム、ネットワークコントロールシステムなどが挙げられる。

＊ 組み込みシステムにおける OSS 活用時には、サポートや、再配布・改変の際のライセンス形態に留意する必要がある。

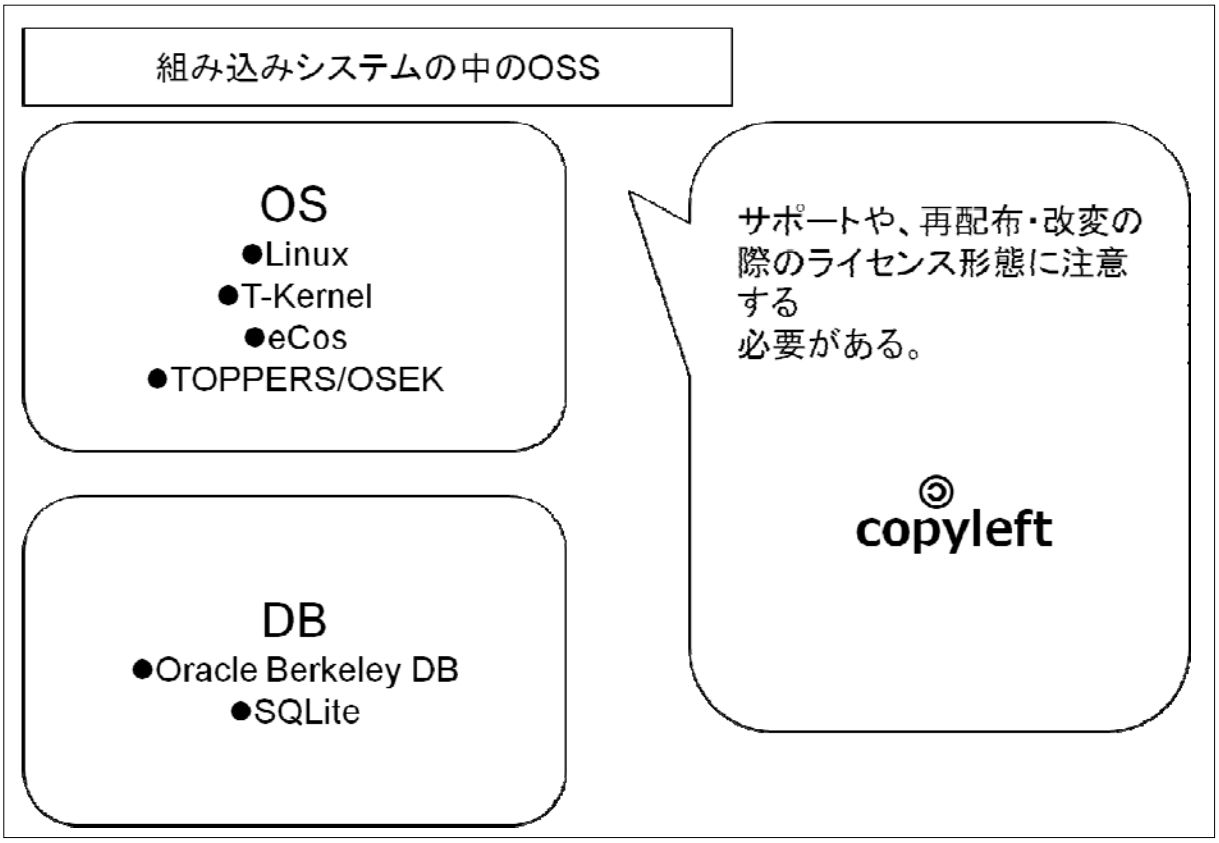

図 8-1-基- 2 組み込みシステムの中の OSS

【解説】

1）組み込みコンピュータシステムとOSS の関係

＊ OSS の必要性

- 従来、組み込みシステムにはメーカが独自に開発したOSを採用することが多かった。
- 現在は、CPU 性能の向上に伴い、Linux のような汎用 OS を修正して採用するケースが増えている。
- 開発・保守コスト、互換性、カスタマイズ性などの観点からOSSの組み込みOSの重要性が増している。

＊ OSS の OS・DBMS

- オープンソース組み込み OS として、Linux、T-Kernel、eCos、TOPPERS/OSEK が著名であり、携帯電話や薄型テレビなどでの実績を多く持つ。
- オープンソース組み込み DB として Oracle Berkeley DB、SQLite などが利用できる。

＊ OSS の活用事例

- ユビキタスセンサシステム、ネットワークコントロールシステムなどが挙げられる。

＊ OSS 活用時の留意点

- 組み込みシステムにおける OSS 活用時には、サポートや、再配布・改変の際のライセンス形態に留意する必要がある。

スキル区分	OSS モデルカリキュラムの科目	レベル
開発体系分野	8-1-基　組み込みシステム開発に関する知識	基本
習得ポイント	8-1-基-3. 組み込みコンピュータの要件	
対応する コースウェア	第3回 組み込みシステムの基本構成 第4回 組み込みコンピュータハードウェアの基本 第5回 CPU アーキテクチャの概要 第6回 組み込みソフトウェアの概要	

8-1-基-3. 組み込みコンピュータの要件

組み込みシステム開発におけるエミュレータの活用方法について紹介する。エミュレータを利用することで専用ハードウェアがない状況でも実行やデバッグができることを解説する。

【学習の要点】

＊ 組み込みシステムにおいてソフトウェアの実行環境は専用ハードウェア上のみであるが、CPU エミュレータを利用することで模擬することができる。

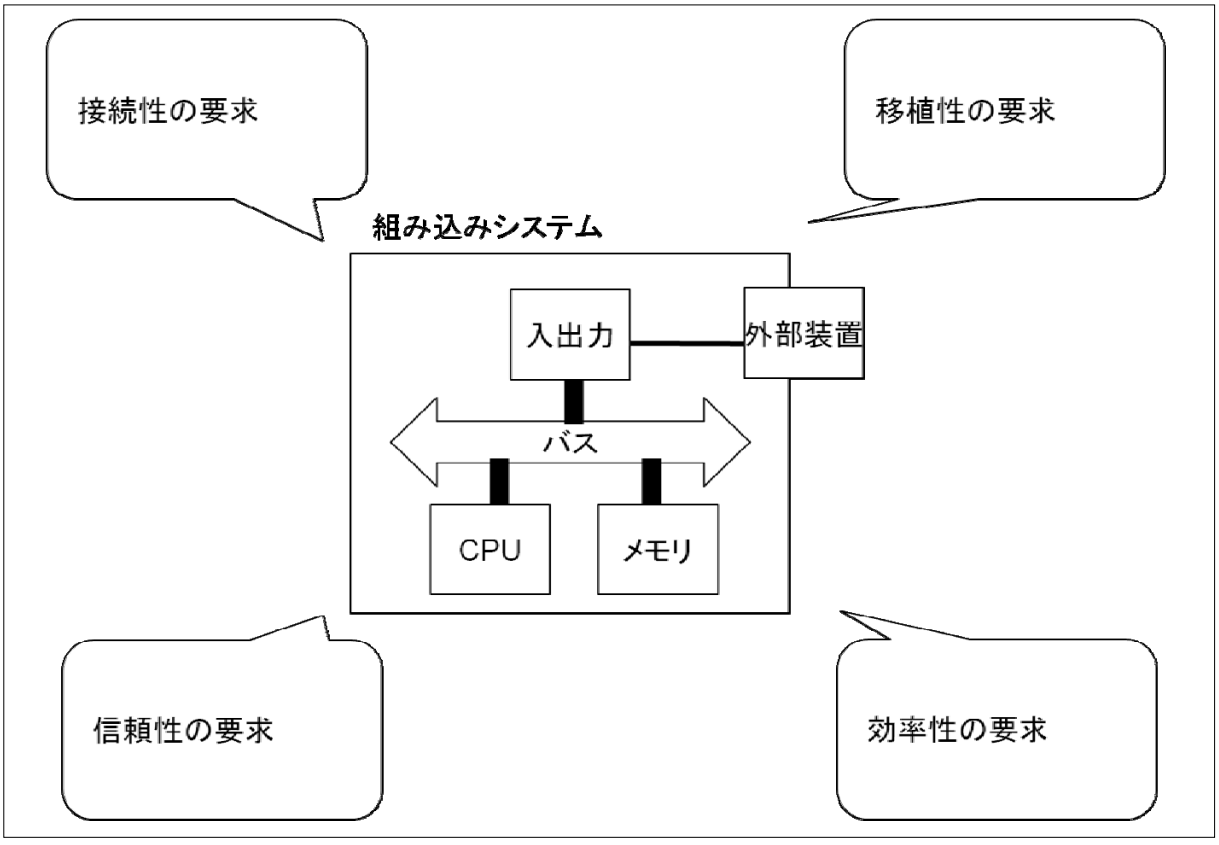

図 8-1-基- 3 組み込みコンピュータの要件

【解説】

1) 組み込みコンピュータハードウェアの特徴

＊ 組み込みコンピュータの基本構成は、CPU、メモリ、操作ボタン・表示器などとの入出力インターフェイスであり、これらの間ではシステムバスを使ってやり取りをする。

＊ ハードウェアの組み合わせは目的に応じて多岐にわたる。たとえば、入出力機器として、ロボットなどを動かすモーターや、温度・照度・音声・磁力などを読み取るためのセンサなどが接続される。

＊ 情報家電のネットワーク化などが進み、ネットワーク接続を要求される組み込みシステムが増えている。DLNA（Digital Living Network Alliance）や ZigBee(IEEE 802.15.4)などの進展も影響していくと考えられる。。

2) 組み込みコンピュータアーキテクチャの必要要件

組み込みコンピュータアーキテクチャには、接続性、信頼性、効率性、そして移植性という４つの要件がある。

＊ 接続性

- ネットワークを介したメンテナンス、情報更新、機能更新が組み込みシステムの情報機器化に伴い必要となる。

＊ 信頼性

- 点検コストや問題による自主回収のコストを避けるために、メンテナンスを極力排する必要がある。

＊ 効率性

- リアルタイム処理により応答時間を一定範囲内にする必要がある。

＊ 移植性

- 短期間で開発や製品化が行われることに対応するため、ハードウェアやソフトウェアの再利用が必要となる。

スキル区分	OSS モデルカリキュラムの科目	レベル
開発体系分野	8-1-基　組み込みシステム開発に関する知識	基本
習得ポイント	8-1-基-4. 組み込み向けソフトウェアの基本	
対応する コースウェア	第7回 カーネルの基本	

8-1-基-4. 組み込み向けソフトウェアの基本

組み込み向けソフトウェアの役割と特徴を説明する。組み込み向けソフトウェア処理の基本として、スレッドとスケジューリングを説明し、並行処理のアーキテクチャに関する基本的な方式とスレッド管理方法などに関する話題にも触れる。

【学習の要点】

* 処理内容を実現するプログラムに対して、その実行の単位をスレッドと呼ぶ。
* どのスレッドを実行するか決定することをスレッドスケジューリングと呼ぶ。スレッドスケジューリングはリアルタイム OS では特に重要な機能である。
* 優先度に従ったスレッドのスケジューリングでは、システムコールをトリガとしてスレッドの実行状態を実行待ちに変化させ、順に処理されるようにする。
* スケジューラが必要な際に実行中のスレッドを強制的に中断し、別のスレッドを実行に移すことをプリエンプションと呼ぶ。

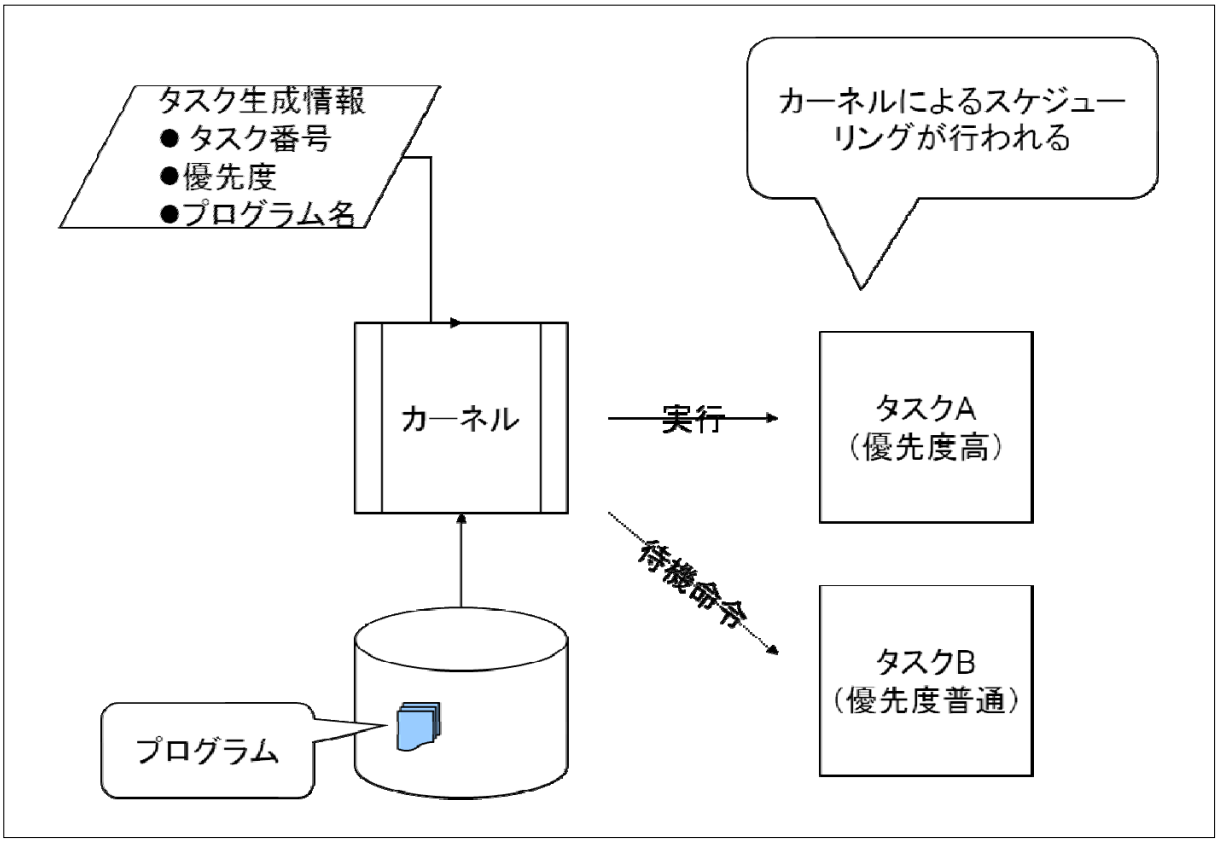

図 8-1-基- 4 カーネルによるスレッド管理

【解説】

1）ソフトウェア処理の基本

＊ タスク

- 処理内容を実現するプログラムに対して、そのスケジュール可能な実行の単位をタスクと呼ぶ。リアルタイム OS においてはスレッドと同等の意味合いで用いられる。
- カーネルによっては、プロセスと呼ばれる別のスケジュール可能な実行の単位を提供している。プロセスには固有のアドレス空間やメモリなどの実行環境が与えられる。

＊ タスクスケジューリング

- どのタスクを実行するか決定することをタスクスケジューリングと呼ぶ。
- 優先度に従ったタスクのスケジューリングでは、システムコールをトリガとしてタスクの実行状態を実行待ちに変化させ、順に処理されるようにする。
- スケジューラが必要な際に実行中のタスクを強制的に中断し、別のタスクを実行に移すことをプリエンプションと呼ぶ。

2）並行処理のアーキテクチャ

＊ 並行処理の処理概要

- 並行処理は複数の相互に影響を及ぼしあう処理を同時並行に実行することを指す。
- タイムクォンタム方式では、各タスクにタイムクォンタムという一定の CPU 時間を順番に割り当てて処理を進めていく。
- タスクコントロールブロック（TCB：Task Control Block)は、カーネルがタスク固有の情報を保持するデータ構造であり、タスクの実行中に動的に切り替わるコンテキスト情報を保持する。

＊ タスクの管理方法

- 生成されたタスクの情報をリングバッファなどでバッファリングすることで、タスクのスケジューリングを行う。

スキル区分	OSS モデルカリキュラムの科目	レベル
開発体系分野	8-1-基　組み込みシステム開発に関する知識	基本
習得ポイント	8-1-基-5. コンテキストスイッチの仕組み	
対応する コースウェア	第8回 タスクとコンテキスト	

8-1-基-5. コンテキストスイッチの仕組み

複数の処理(スレッド)を実行するコンテキストスイッチの仕組みを解説する。OS が処理の切り替えを管理するプリエンプション方式の実装方法を説明し、割り込みによるコンテキストスイッチにも触れる。

【学習の要点】

- ＊ 各スレッドはそれぞれ固有のコンテキストを持つ。コンテキストとは、実行のスケジューリングごとに必要となる CPU のレジスタ状態であり、プログラムの連続した流れとして見ることができる。
- ＊ コンテキストスイッチは、スケジューラがあるスレッドから別のスレッドへ実行を切り替える際に、レジスタの値を退避・復元することにより発生する。
- ＊ OS のスケジューラが必要に応じて実行中のスレッドを強制的に中断し、別のスレッドを実行する方式をプリエンプション方式と呼ぶ。
- ＊ スレッドのスケジューリングは優先度によって制御される。

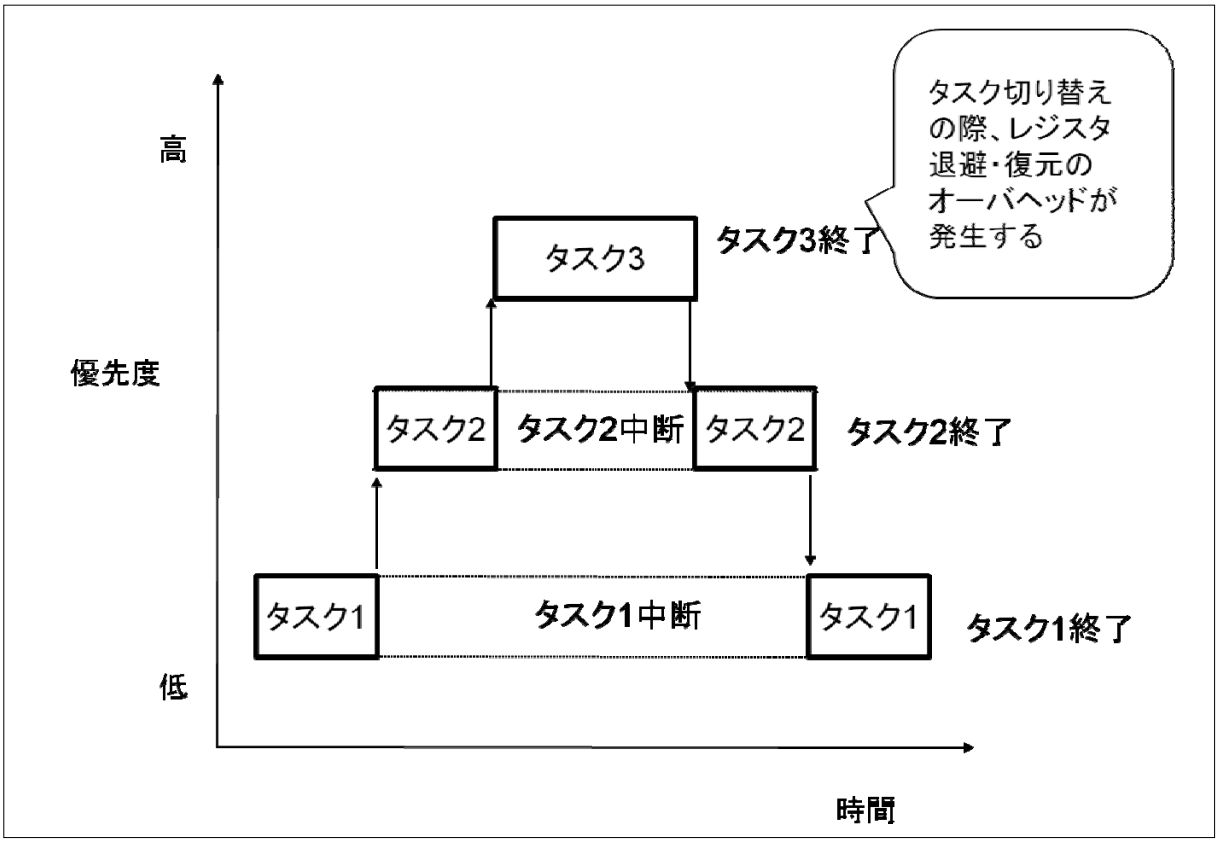

図 8-1-基- 5 コンテキストスイッチの発生する状況

【解説】

1) コンテキストスイッチング

＊ コンテキストスイッチングとは

- コンテキストとは、実行のスケジューリングごとに必要となる CPU のレジスタ状態であり、プログラムの連続した流れと見られる。
- コンテキストスイッチは、スケジューラがあるタスクから別のタスクへ実行を切り替える際に行うレジスタの値の退避・復元により発生する。
- OS のスケジューラが必要に応じて実行中のタスクを強制的に中断し、別のタスクを実行する方式をプリエンプション方式と呼ぶ。

＊ コンテキストスイッチングの実装仕様

カーネルのスケジューラがタスク 1 からタスク 2 に実行を切り替える時の動作は以下の通り。

- カーネルはタスク 1 のコンテキストをレジスタに退避する
- カーネルはタスク 2 のコンテキストを読み込み、タスク 2 を実行中の状態にする。
- タスク 1 が再び実行状態に復帰する場合、タスク 1 はコンテキストスイッチ直前に実行していた実行状態から再開する。

＊ コンテキストスイッチ時間

- スケジューラがタスク切り替えに要する時間をコンテキストスイッチ時間という。
- 頻繁なコンテキストスイッチを発生させるような設計は、コンテキストスイッチ時間の累積によるパフォーマンスの低下を招く恐れがあり、避けるべきとされる。

2) カーネルコンテキスト

＊ システムコールなどは、通常のアプリケーションが実行されるユーザモードとは異なり、カーネルモード(I-26-2 参照)で実行されるため、カーネルモードの実行に必要なカーネルコンテキストを呼び出す必要がある。

＊ デバイスドライバなどがハードウェア割り込みを行う場合、カーネルの割り込みハンドラを登録する。

＊ 割り込み処理を行っている際のコンテキストを割り込みコンテキストという。

スキル区分	OSS モデルカリキュラムの科目	レベル
開発体系分野	8-1-基　組み込みシステム開発に関する知識	基本
習得ポイント	8-1-基-6. 非同期処理と同期処理の実装パターンと特徴	
対応する コースウェア	第9回 非同期と同期の設計仕様	

8-1-基-6. 非同期処理と同期処理の実装パターンと特徴

非同期/同期の概念について触れ、その実装パターンと設計について内容と特徴を説明する。また実行モードの種類や割り込みの優先順位といった具体的な事項について説明する。

【学習の要点】

* 非同期処理は、あるスレッドが実行をしている際に、他のスレッドが別の処理を実行できる方式である。同期処理では、あるスレッドが実行している間、他のスレッドの処理は中断される方式である。
* スレッドの情報を保持する管理テーブル(Task Control Block: TCB)から、プログラムへのポインタとコンテキストの情報を取得して、カーネルはスレッドを起動する。
* スレッドの実行モードにはカーネルモードとユーザモードがある。カーネルモードは全ての命令の実行が許可されるのに対し、ユーザモードでは実行できる命令に制限が加わる。

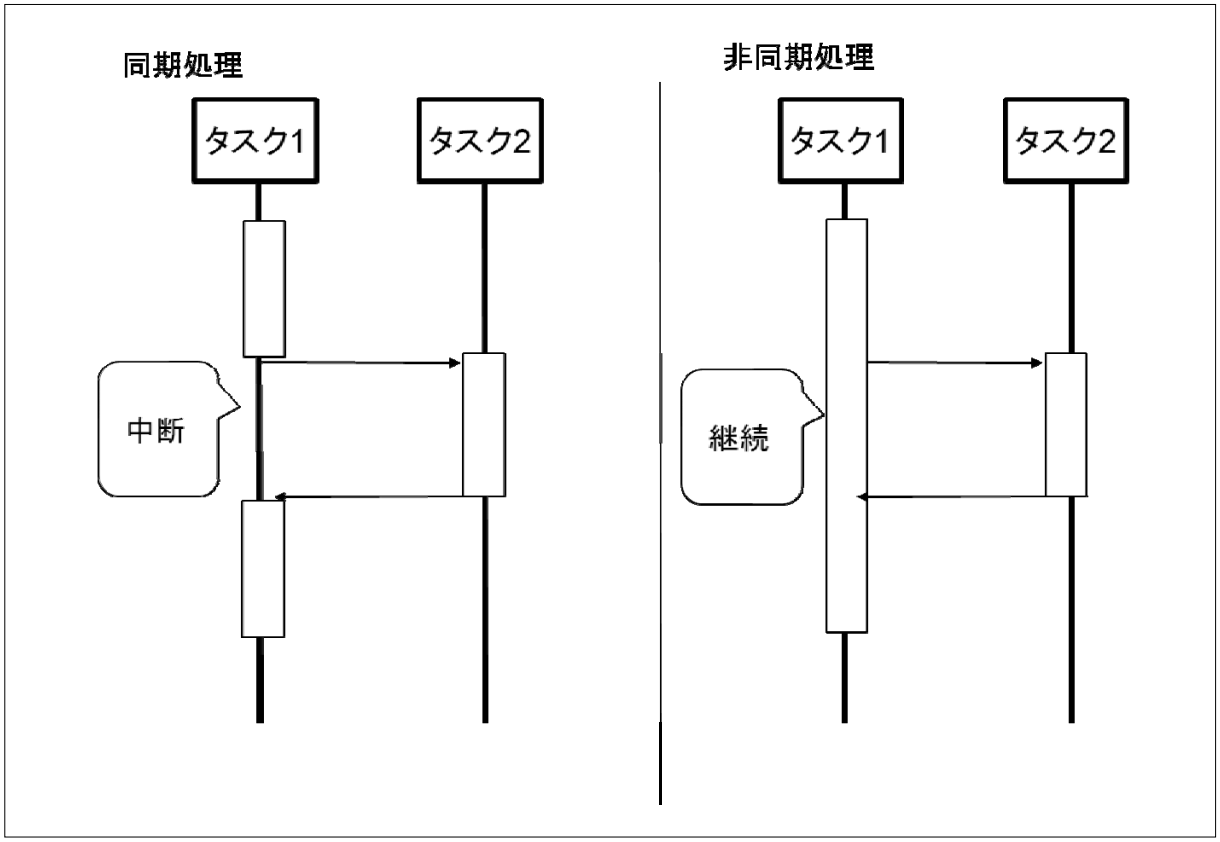

図 8-1-基- 6 同期処理と非同期処理

【解説】

1) 非同期と同期の設計

非同期処理は、あるタスクが実行をしている際に、他のタスクが別の処理を実行できる方式である。同期処理では、あるタスクが実行している間、他のタスクの処理は中断される方式である。

＊ システム全体のコンテキストの設計

- 非同期処理を入れ、多重にコンテキストを用意することにより処理速度の向上を目指す設計が行われる。

2) タスクの構成資源

＊ タスクの生成

- タスクは生成時に、プログラム部分の他、名前、GUID（一意の ID）、優先度、タスクの情報を保持する管理テーブル（Task Control Block: TCB)、スタックが割り当てられる。

＊ 実行モードと起動方法

- カーネルモード
  OS の実行モードであり、全ての命令の実行が許可される。あらゆるハードウェア資源にアクセスできる。

- ユーザモード
  通常のプログラムの実行モードであり、命令の実行に制限が加わる。ハードウェア資源に制限・監視付きでアクセスできる。

スキル区分	OSS モデルカリキュラムの科目	レベル
開発体系分野	8-1-基　組み込みシステム開発に関する知識	基本
習得ポイント	8-1-基-7. タスク優先順位制御とカーネルによる時間管理方法	
対応する コースウェア	第10回 タスクの優先度とその制御仕様	

8-1-基-7. タスク優先順位制御とカーネルによる時間管理方法

スレッドの優先度とその制御仕様について、実装パターンと設計内容、特徴、手順を説明する。実際のスレッド制御を行うカーネルが時間制御方法としてどのような管理手法を用いているかについても触れる。

【学習の要点】

＊ カーネルは優先度を設定してスレッドの優先度順実行を制御する。

＊ カーネルによる時間管理には、プロセス実行を時分割で切り替えるタイムスライス方式、割り込み等のイベントを受けてスレッドを切り替えるプリエンプティブ方式などがある。

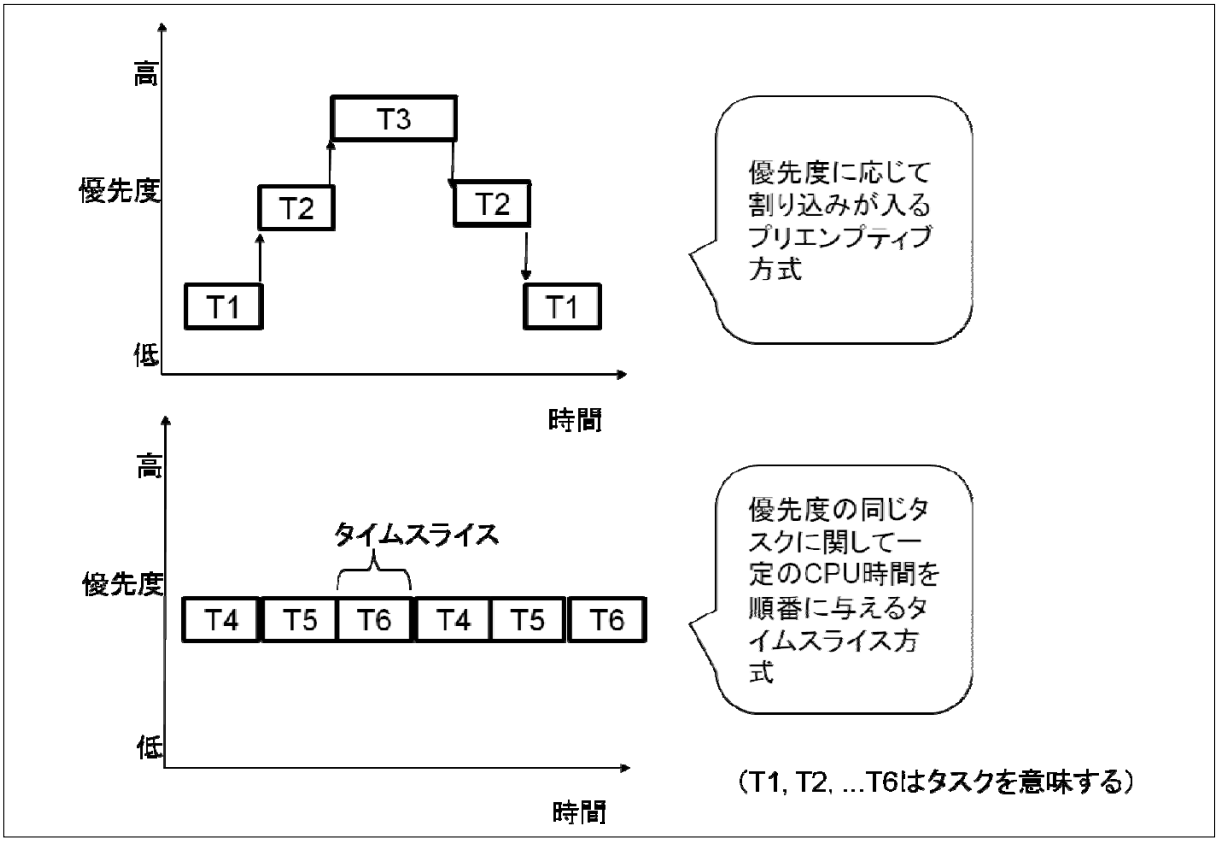

図 8-1-基- 7 プリエンプティブ方式とタイムスライス方式

【解説】

1) タスクの優先順位

カーネルは優先度を設定してタスクの優先度順実行を制御する。

＊ プログラムのリエントラント性と優先度

- リエントラント(再入可能)とは、プログラムに含まれるあるルーチンの実行中に同じルーチンが呼び出されても正しい結果を返せることを指す。
- C 言語の場合、スタティック変数の更新を行わない関数はリエントラント性が保証される。

＊ 同期／排他制御と優先度

- リエントラントでないルーチンを使う前後で排他制御を用いると、そのルーチンはリエントラントルーチン同様に、複数のプログラムから利用できる。
- 上述の使い方のできるライブラリを特に、スレッドセーフライブラリ(Thread Safe Library)と呼ぶ。

2) カーネルによる時間管理

＊ カーネルによる時間管理には、タスクの実行を時分割で切り替えるタイムスライス方式、割り込み等のイベントを受けてタスクを切り替えるプリエンプティブ方式などがある。

- タイムスライス方式
  複数のタスクを平等に実行させるための方式。タイムスライスという単位時間でタスクを切り替えて実行する方式である。

- プリエンプティブ方式
  外部からの割り込みが発生すると、スケジューリングを行い、優先度に応じて処理を切り替える方式である。

- フィードバック待ち行列
  優先度のある待ち行列にタスクを並べ、順に処理を行う。その際、一定時間内に処理が終わらない場合、優先度を落とす方式である。

スキル区分	OSS モデルカリキュラムの科目	レベル
開発体系分野	8-1-基　組み込みシステム開発に関する知識	基本
習得ポイント	8-1-基-8. 割り込みと DMA	
対応する コースウェア	第１１回 割り込みと DMA	

8-1-基-8. 割り込みと DMA

組み込みシステムには欠かせない割り込みと DMA(Direct Memory Access)について解説する。割り込みの仕組みとそれによる OS の動作や DMA の仕組みやメリットを紹介する。

【学習の要点】

＊ 割り込みとは、外部からの信号などによって CPU に優先的に処理をさせる方法である。

＊ 割り込み処理中は他のスレッドが動作できなくなるので、割り込み処理は必要最小限にとどめること。

＊ DMA を利用することで、CPU を利用することなくメモリ転送をおこなうことができるため、CPU 資源を効率よく使用することができる。

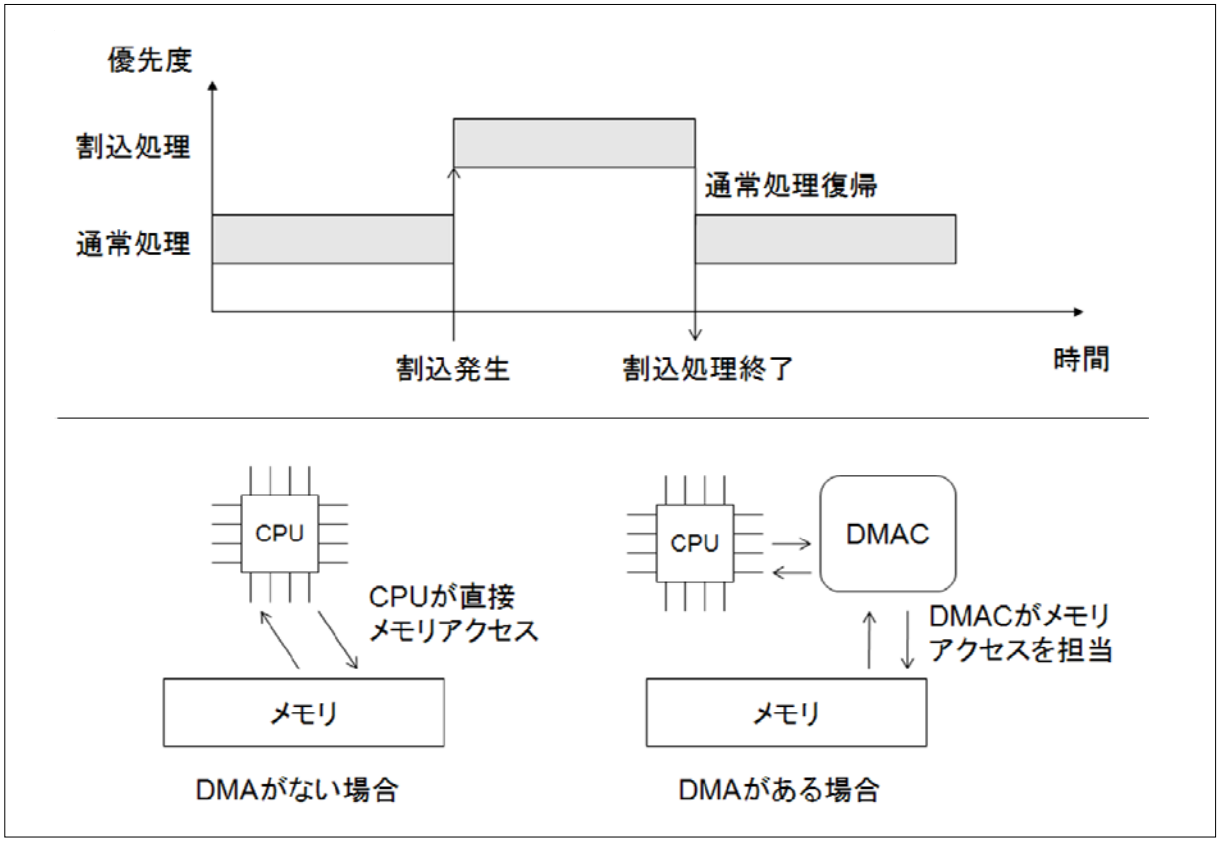

図 8-1-基- 8 割り込みと DMA

【解説】

1) 割り込み

割り込みとは CPU がハードウェアまたはソフトウェアから受け取る要求で、割り込みを受け取った CPU はすべての処理を中断し、割り込みのための処理を行う。
割り込みを利用することで、要求が来るまで別の処理を実行することができるため、CPU 時間を有効に利用することができる。

2) 割り込みの種類

割り込みには大きく分けて外部信号として起動されるハードウェア割り込みとソフトウェアから起動されるソフトウェア割り込みの 2 種類がある。

* ハードウェア割り込み
  - マスカブル割り込み
    禁止できる割り込み
  - ノンマスカブル割り込み
    禁止できない割り込み
  - リセット割り込み
    電源投入と同時に入る割り込み。リセット割り込みからソフトウェアの動作が開始される CPU もある。
* ソフトウェア割り込み
  ソフトウェアから発生する割り込み。トラップと呼ばれることもありソフトウェアから意図的に発生することも可能。

3) 割り込み処理の注意点

割り込み処理の最中は他の処理(プロセス/スレッド)は完全に停止してしまうため、割り込み処理からプロセス/スレッドに通知を出すなど、必要最低限の処理時間にする必要がある。

4) DMA

DMA とは CPU を使用せず、DMAC(DMA Contoller)と呼ばれるハードウェアがメモリアクセスを専用に受け持つ。このため、メモリ転送のための CPU 時間を空けることができるので、効率的に CPU を利用することができる。

スキル区分	OSS モデルカリキュラムの科目	レベル
開発体系分野	8-1-基　組み込みシステム開発に関する知識	基本
習得ポイント	8-1-基-9. システムリソースの配分の仕組み(セマフォ、キュー、等)	
対応する コースウェア	第12回 組み込みアプリケーション間の資源配分技術(セマフォ) 第13回 組み込みアプリケーション間の資源配分技術(キュー)	

8-1-基-9. システムリソースの配分の仕組み(セマフォ、キュー、等)

組み込みシステムにおいては、システムリソースが限定されているため様々な工夫が必要になる。各アプリケーションが限られたシステム資源を効率的に共有する方法について、セマフォ、キュー、といった具体的な実装方法を踏まえて、その内容と特徴を説明する。

【学習の要点】

* 組み込みシステムにおいては、CPU リソース、メモリリソース、ネットワークリソース、制御時間といったリソースの配分が重要である。
* 複数のスレッドが共有資源にアクセスしようとした際に、整合性を保つために同時アクセスを禁止することを排他制御と呼ぶ。排他制御にはロックやデッカーのアルゴリズム、セマフォなどの方式が使われる。
* カーネルによって制御されているスレッド間の排他制御手段としてセマフォがある。セマフォはスレッドが共有資源にアクセスするために必要な鍵の役割を果たす。スレッドは、セマフォを取得することで共有資源にアクセス可能になる。
* スレッド間通信にキューを用いてメッセージを交換することで、同期をする方式もある。

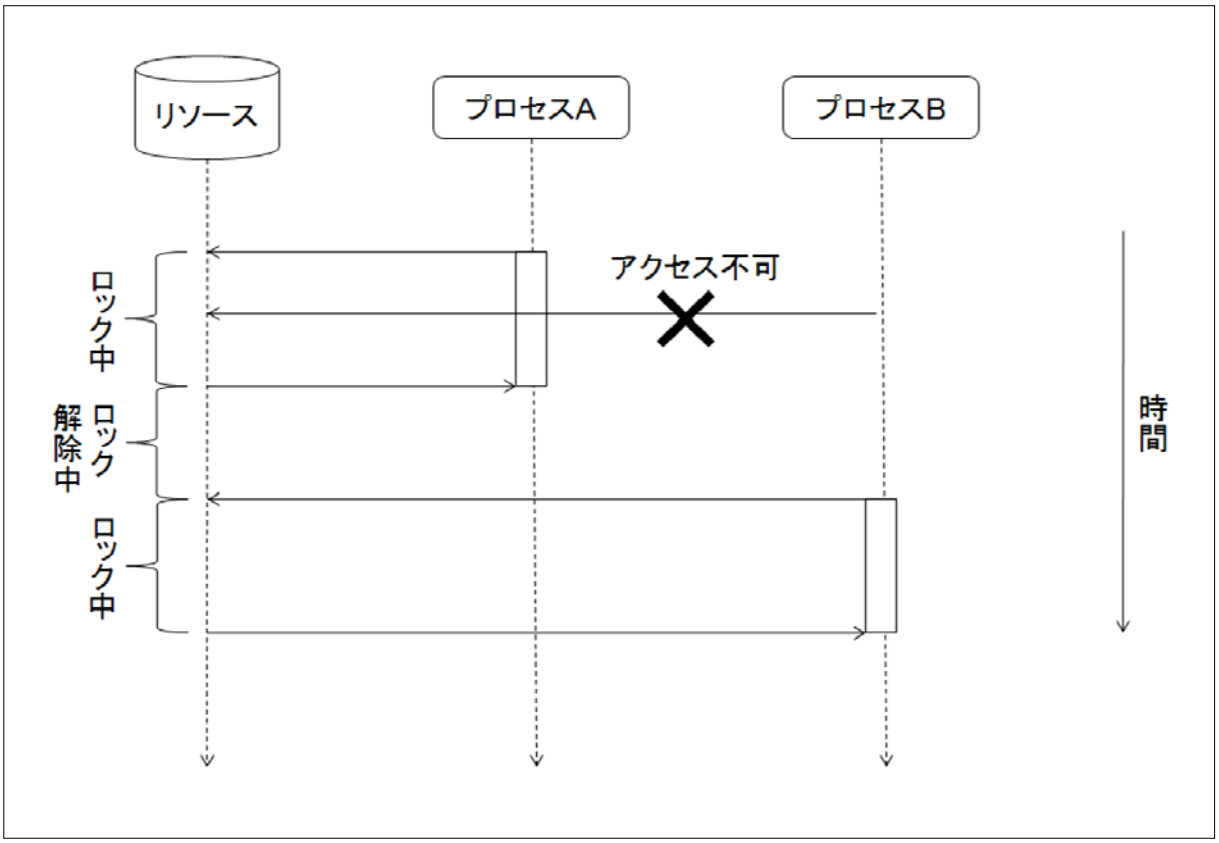

図 8-1-基- 9 アクティビティ同期の例(バリア同期)

【解説】

1) 資源配分技術

＊ 配分すべき資源には、CPU リソース、メモリリソース、ネットワークリソース、制御時間がある。

＊ クリティカルセクション

- 共有リソースにアクセスするコードの区間であり、プリエンプションが発生することによってデータの整合性などに問題が生じる可能性のある部分を指す。

＊ 同期制御と排他制御

- リソース同期
  共有リソースの整合性を保つために複数タスク間での同期をすること。

- アクティビティ同期
  タスクがアクティビティを他のタスクと同期すること。バリア同期がその例として挙げられる。バリア同期では、複数のタスクがある時間をバリアとして「待ち合わせ」をする。

- スピンロックによる排他制御
  あるタスクがクリティカルセクションに入る直前にフラグをチェックすることで、他のタスクがクリティカルセクションに入っていないことを確認してからクリティカルセクションに入る。他のタスクがクリティカルセクションに入っている場合は、フラグをチェックし続ける。

＊ ハードウェアによる方法

- Test and Set 命令は、スピンロックにおけるフラグのチェックをアトミックに実現する機械語命令である。

＊ ソフトウェアによる方法

- デッカーのアルゴリズム(I-24-9 参照)、ランポートのパン屋のアルゴリズムといった相互排除のアルゴリズムが考案されている。

2) カーネルの提供する手段を使う方法

＊ セマフォ
セマフォはタスクが共有資源にアクセスするために必要な鍵の役割を果たす。セマフォを取得することで共有資源にアクセス可能になる。

- バイナリセマフォ
  セマフォは 0 か 1 の値を持つ。0 ならば共有資源は利用不可、1 ならば利用可能である。

- ジェネラルセマフォ(カウンティングセマフォ)
  カウンタにより、共有資源へのアクセス権の複数回の取得や解放を管理できる。

＊ キューイング

- タスク間通信にキューを用いてメッセージを交換することで、同期をする方式もある。

スキル区分	OSS モデルカリキュラムの科目	レベル
開発体系分野	8-1-基　組み込みシステム開発に関する知識	基本
習得ポイント	8-1-基-10. システムリソースの共有の仕組み(共有ファイル)	
対応する コースウェア	第14回 組み込みアプリケーション間のリソース共有技術	

8-1-基-10. システムリソースの共有の仕組み(共有ファイル)

複数の組み込みアプリケーション同士でシステムリソースを共有する方法を、共有エリア・共有ファイル、カーネルによる各種のサービス、デッドロックの回避、割り込みディスパッチなどの具体的な実装を挙げて説明する。

【学習の要点】

＊ アプリケーションプロセス間で共有するデータの参照方式にはシンボル参照、ファイル参照などがある。

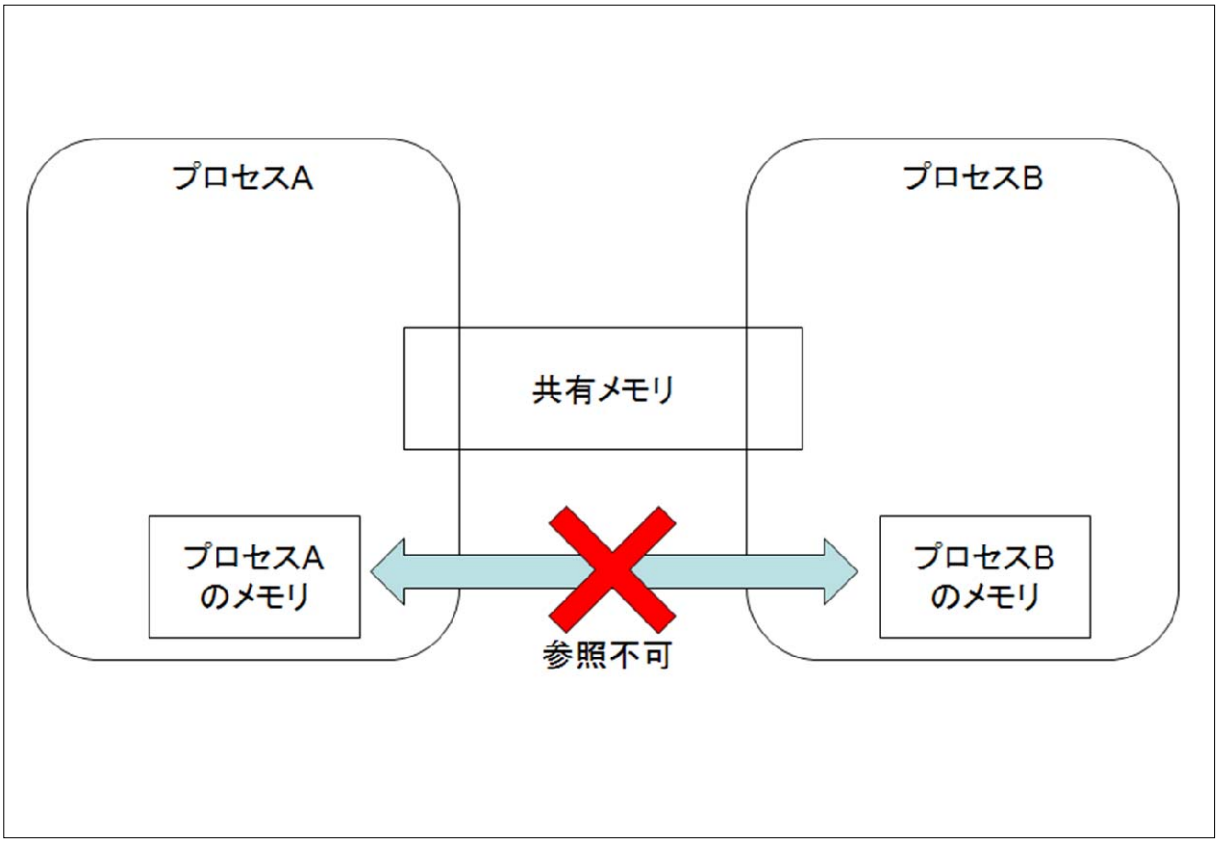

図 8-1-基- 10 共有メモリとプロセス間のメモリアクセス

【解説】

1) 共有データの参照

組み込みソフトウェアは、モジュール化が進んでいるため、タスク間でデータやライブラリを共有することが多い。

* シンボル参照
  - プロセス空間内の同一領域に、共有データ領域を予約しておく方式。
* ファイル参照
  - メモリ上に擬似的にファイルを構成することで、共有ファイルとしてデータの読み書きを行う方式。

2) カーネルによるサービス

* メッセージキュー
  - 2つのタスク間のデータ通信などに利用されるカーネルオブジェクト。
* パイプ
  - タスク間のデータ交換機構。データは片方のディスクリプタに書き込まれ、もう片方のディスクリプタから読み取られる。Linux などのシェルにおいてパイプは"|"記号で表現され、頻繁に利用される。

3) デッドロック

* デッドロックとは、複数のタスクがリソース要件を満たせずに処理が全く進まなくなってしまう状態である。
* デッドロックを処理する戦略として、デッドロック検出と復旧、デッドロック回避、デッドロック予防がある。これらはリアルタイム OS の機能として提供される。

4) 割り込みスケジューリング（ディスパッチング）

* ディスパッチャはスケジューラの一部であり、実際のコンテキストスイッチ処理を行う。
* システムコールを行う主体がタスクか割り込みサービスルーチン(Interrupt service routines: ISR)かによって処理が異なり、ISR が優先される。