


複数のポリシ/メカニズムを搭載 した学習向け組込みOSの実装

拓殖大学院 茂木 高宏

目次

1. 背景と問題点
2. 目的
3. 方針
4. 全体構成
5. 複数のポリシとメカニズム
6. おわりに


背景

- 組込みOSの設計として、複数のポリシ/メカニズムが考えられている
 - 組込みOSアプリケーション技術者では、
実現したいアプリケーション特性に適応したOSのポリシ/メカニズムの選択が行なわれている
 - ▶ 複数のポリシ/メカニズムに対する知識が求められている
-  複数のポリシ/メカニズムの内部構造及び動作の理解

問題点

- 組込みOSでは、メカニズムからポリシの完全分離が困難

- ▶ ポリシに対する柔軟性の低下

-  複数のポリシの動作学習が難しい




複数のポリシにおける動作の差異の学習も困難

- メカニズムは、組込みOSの特徴に応じて変化

- ▶ 単一環境下から、OSの特徴を無視した複数のメカニズムの動作学習が難しい

- OS記述方式が複雑化し、内部構造の理解が困難

- ▶ コンフィギュレーションの存在

-  仕様が明確化されていない

目的

- 複数のポリシ/メカニズムを搭載した組込みOSの自作
 - ▶ 複数のポリシ/メカニズムの動作学習が可能
 - ▶ 複数のポリシ/メカニズムに対する動作差異の学習が容易に可能
 - ▶ 可読性を向上させたOSソースコード



対象：組込みアプリケーション技術者

~~1. 背景と問題点~~

~~2. 目的~~

3. 方針

4. 全体構成

5. 複数のポリシとメカニズム

6. おわりに

方針：OS設計方針

- 複数のメカニズム

- ▶システムコールの方式

- 複数のポリシ

- ▶タスクスケジューリング

- ▶優先度逆転防止プロトコル

- 動作の学習をするために

- サンプルタスクセットライブラリを提供

- OSの基本的な機能を提供

- ▶広く使用されている μ ITRON4.0仕様を参考

方針：OS記述方針

- C言語, アセンブラ, リンカスクリプトを使用
- OSソースコード可読性考慮として,
 - ▶ コンフィギュレーションを行なわない
 - ➡ すべてのシステムコールは μ ITRON4.0の動的APIを使用
 - ▶ MISRA C2004を参考にコーディング
 - ▶ コメント比率の増加とアセンブラ比率の減少

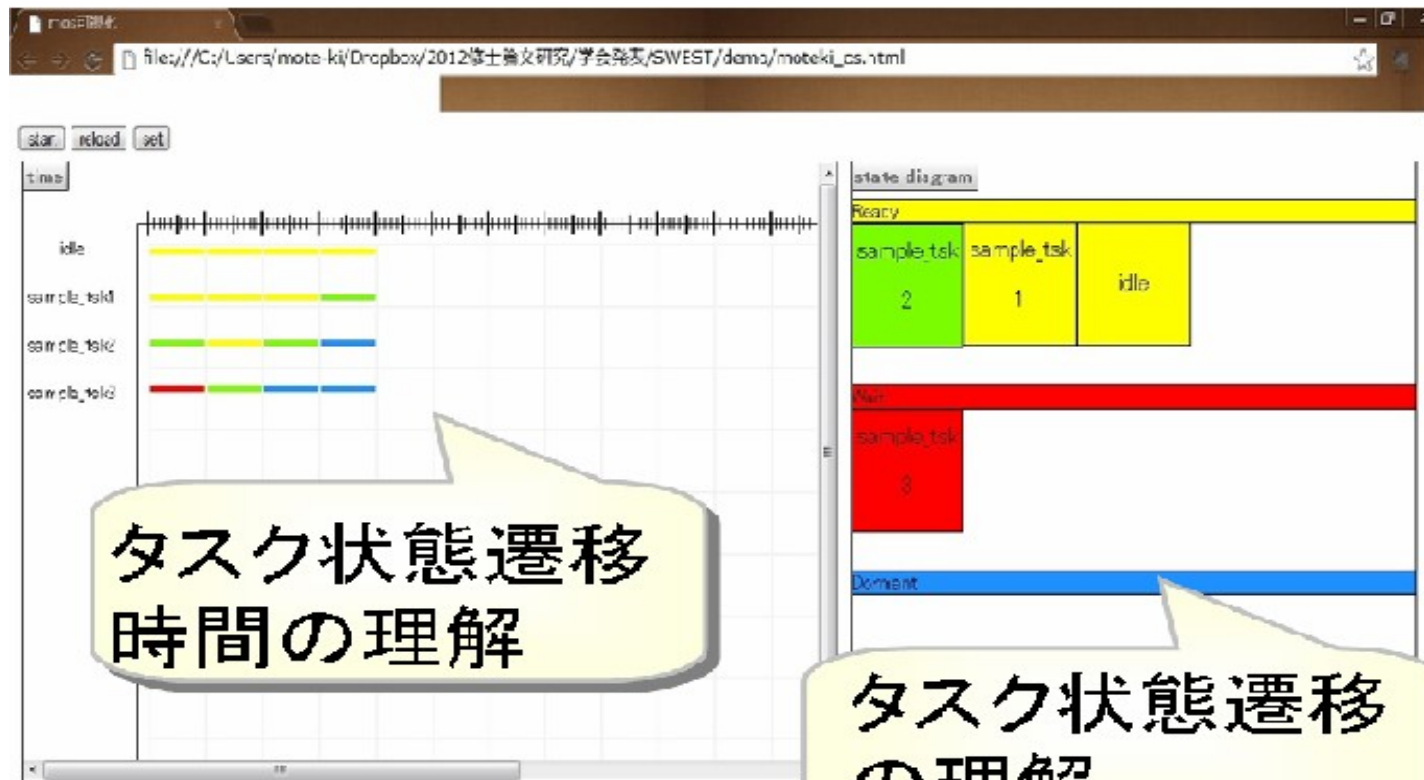
方針：学習方針

- OSの動作は実機上
- 複数のポリシ/メカニズムの動作の学習方法
 - ▶ 直感な動作学習
 - ➡ 同研究室で開発された可視化ツールを使用
 - ▶ 詳細な動作学習
 - ➡ シリアルコンソールに表示されるシリアルメッセージ
 - ➡ サンプルタスクセットライブラリ

方針：直観な動作学習

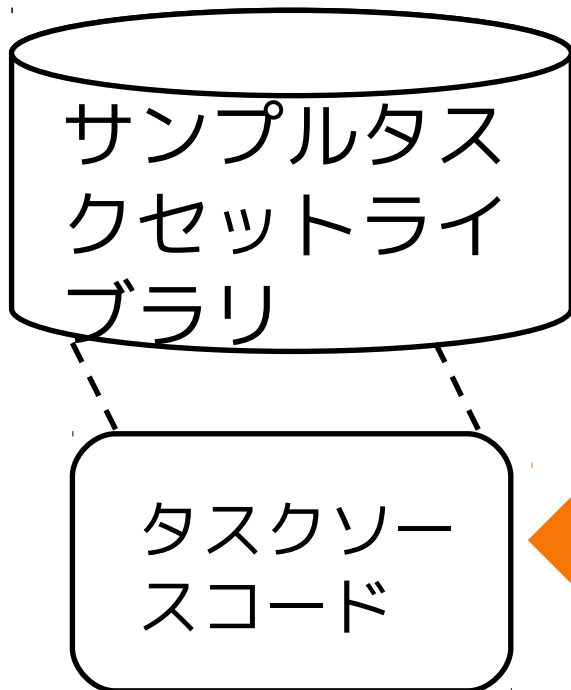
● 可視化ツールの使用

- ▶ 複数のポリシ(タスクスケジューリング)がタスクに与える動作の学習が可能



方針：詳細な動作学習

- シリアルコンソールを使用
 - ▶ 複数のポリシにおけるタスク切替えの事象を学習
 - ▶ カーネルオブジェクトの学習



ファイル(E) 編集(E) 表示(V) 端末(T) ヘルプ(H)

```
command unknown.  
> run tsk_set1  
> sample_tsk1 started.  
sample_tsk1 create tsk(sample_tsk2).  
sample_tsk1 create running in (sample_tsk2).  
sample_tsk2 started.  
sample_tsk2 create task(sample_tsk3).  
sample_tsk2 create running in (sample_tsk3).  
sample_tsk3 started.  
sample_tsk3 DORMANT.  
sample_tsk2 create running out (sample_tsk3).  
sample_tsk2 delete task(sample_tsk3).  
sample_tsk2 EXIT.  
sample_tsk1 create running out (sample_tsk2).  
sample_tsk1 DORMANT.
```

```
command unknown.  
> □
```

- 実機上のOSは、シリアルコンソールからコマンドで制御

~~1. 背景と問題点~~

~~2. 目的~~

~~3. 方針~~

4. 全体構成

5. 複数のポリシとメカニズム

6. おわりに

組み込みシステム全体構成

■ :自作部 □ :既存部

直観な動作
学習部

PC上

ブラウザ

可視化

ログ加工部

詳細な動作
学習部

ログ

シリアルコンソール

ターゲット上

~OS~

カーネル

サンプルタスク
セットライ
ブラリ

コマンド制御

~実装独自のコマンドー部~

run	...	タスクセットの起動
set	...	ポリシーの選択
sendlog	...	転送プロトコルでログを送信



実装したカーネルの機能

～カーネル～

同期機能

排他機能

mutex

セマフォ

メールボックス

タスク間通信

優先度逆転防止プロトコル

割込み機能

割込み6種類

多重割込み

メモリ管理機能

タスク機能

マルチタスク機

スケジューリング

ディスパッチャ

拡張機能

ログ管理機構

転送プロトコル

時間管理機能

アラームハンドラ 周期ハンドラ

タイママルチ管理

シリアルドライバ

タイマドライバ

：複数のポリシーを用意

OS基本機能を提供

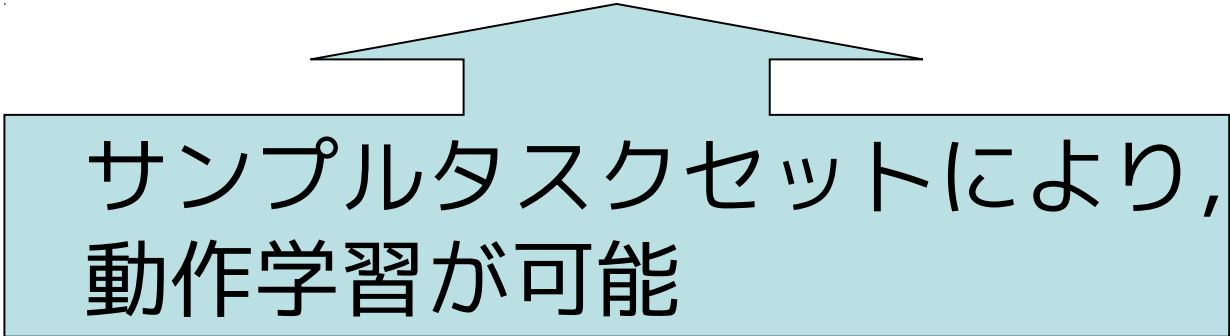
サンプルタスクセットで学習可能

60程度のシステムコールを用意

- ~~1. 背景と問題点~~
- ~~2. 目的~~
- ~~3. 方針~~
- ~~4. 全体構成~~
5. 複数のポリシとメカニズム
6. おわりに

複数のポリシ：タスクスケジューリング詳細(i)

- 13種のタスクスケジューリングを実装
 - ▶ 広く採用されているタスクスケジューリングを対象
 - ▶ 比較対象用として、汎用OS等で使用されるスケジューリングも対象



サンプルタスクセットにより、
動作学習が可能

- ▶ スケジューリングによって、レディー管理データ構造が変化
 - ▶ レディー管理データ構造を5種実装

複数のポリシ：タスクスケジューリング詳細

(ii) ~RTOS等で使用される~ ~汎用OS等で使用される~

- ・ Rate Monotonic
- ・ Deadline Monotonic
- ・ Earliest Deadline First
- ・ Least Laxity First

- ・ First Come First Served
- ・ ラウンドロビン
- ・ ラウンドロビン×優先度スケジューリング
- ・ 優先度

- ・ Fairスケジューリング

- ・ Priority Fairスケジューリング

- ・ Multilevel Feedback Queue

- ・ O(1)スケジューリング

・ ラウンドロビンスケジューリング (ITRONの方式)



toppers/ASPカーネル

ラウンドロビン
スケジューリング

優先度
スケジューリング



構造を簡素化

Linuxカーネル

2.6.11

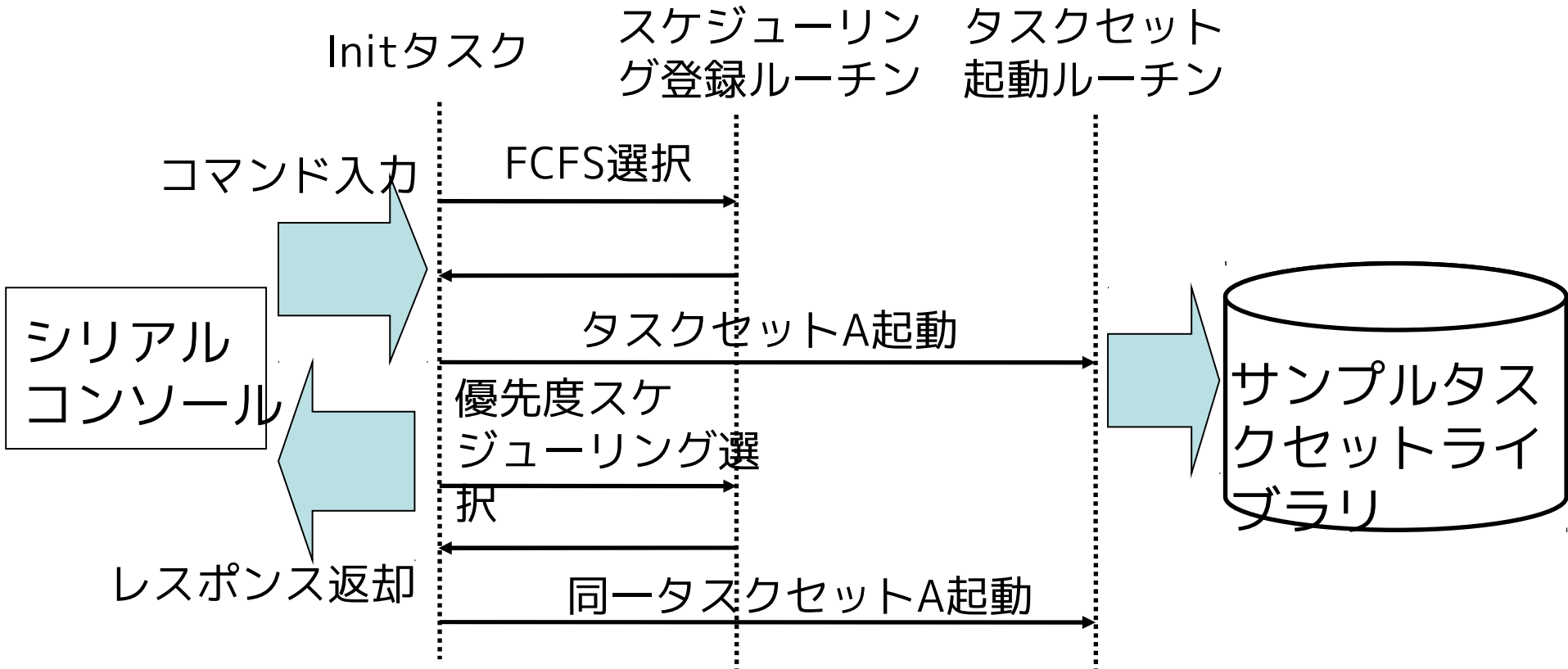
O(1)スケジューリングクラス

複数のポリシ：タスクスケジューリングの選択(i)

- シリアルコンソールからタスクスケジューリング選択コマンドを使用して登録

→ 同一タスクセットに対して異なるスケジューリング
適応時のレスポンスが取得可能

シーケンス(同一タスクセットをFCFSと優先度スケジューリング)



同一タスクセットに異なるスケジューリング適応のレスポンス

Input : サンプルタスクセット

TaskA(優先度低)

```
{ puts("taskA start");  
  タスク生成と起動(TaskB);  
  puts("taskA end");  
}
```

TaskB(優先度中)

```
{ puts("taskB start");  
  タスク生成と起動(TaskC);  
  puts("taskB end");  
}
```

TaskC(優先度高)

```
{ puts("taskC start");  
  puts("taskC end");  
}
```

Output : レスポンス

```
ファイル(E) 編集(E) 表示(V) 端末(T) ヘルプ(H)  
mote-kiOS#>set FCFS  
FCFS scheduling set  
mote-kiOS#>run tsk_set3  
taskA start  
taskA end  
taskB start  
taskB end  
taskC start  
taskC end  
mote-kiOS#>set PRS  
PRS scheduling set  
mote-kiOS#>run tsk_set3  
taskA start  
taskB start  
taskC start  
taskC end  
taskB end  
taskA end  
mote-kiOS#>
```

FCFSスケジューリングをセット

優先度スケジューリングをセット

同一タスクセット

複数のポリシ：優先度逆転防止プロトコル(i)

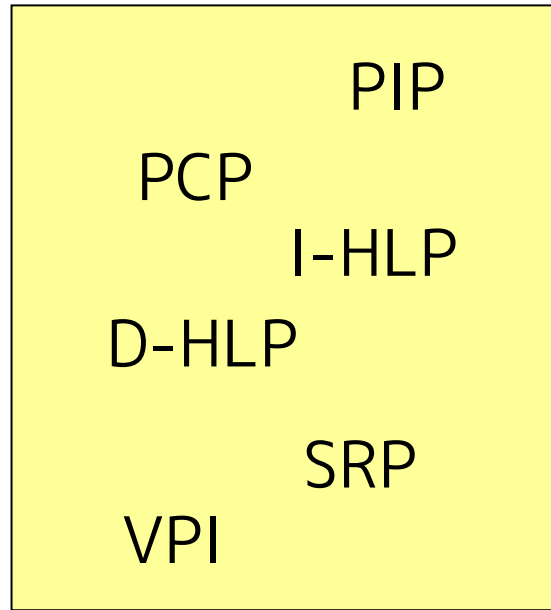
- 6種の優先度逆転防止プロトコルを実装

- ▶ Priority Inheritance Protocol(以下PIP)
- ▶ Priority Ceiling Protocol(以下PCP)
- ▶ Immediate Highest Locker Protocol(以下I-HLP)
- ▶ Delay Highest Locker Protocol(以下D-HLP)
- ▶ Stack Resource Policy(以下SRP)
- ▶ Virtual Priority Inheritance(以下VPI)

 mutexの属性として実装

複数のポリシー：優先度逆転防止プロトコル(ii)

6種の優先度逆転防止
プロトコル



待ちタスクのレディー
返却アルゴリズム

レディーデータ構造5種

～優先度逆転防止プロトコル
で発生する3つの主作用～

デッドロックの誘発現象
推移的優先度継承現象
連続ブロッキング現象

～優先度逆転防止プロトコル
で発生する1つの副作用～

デッドロックの予防

13種のスケジューリン
グ

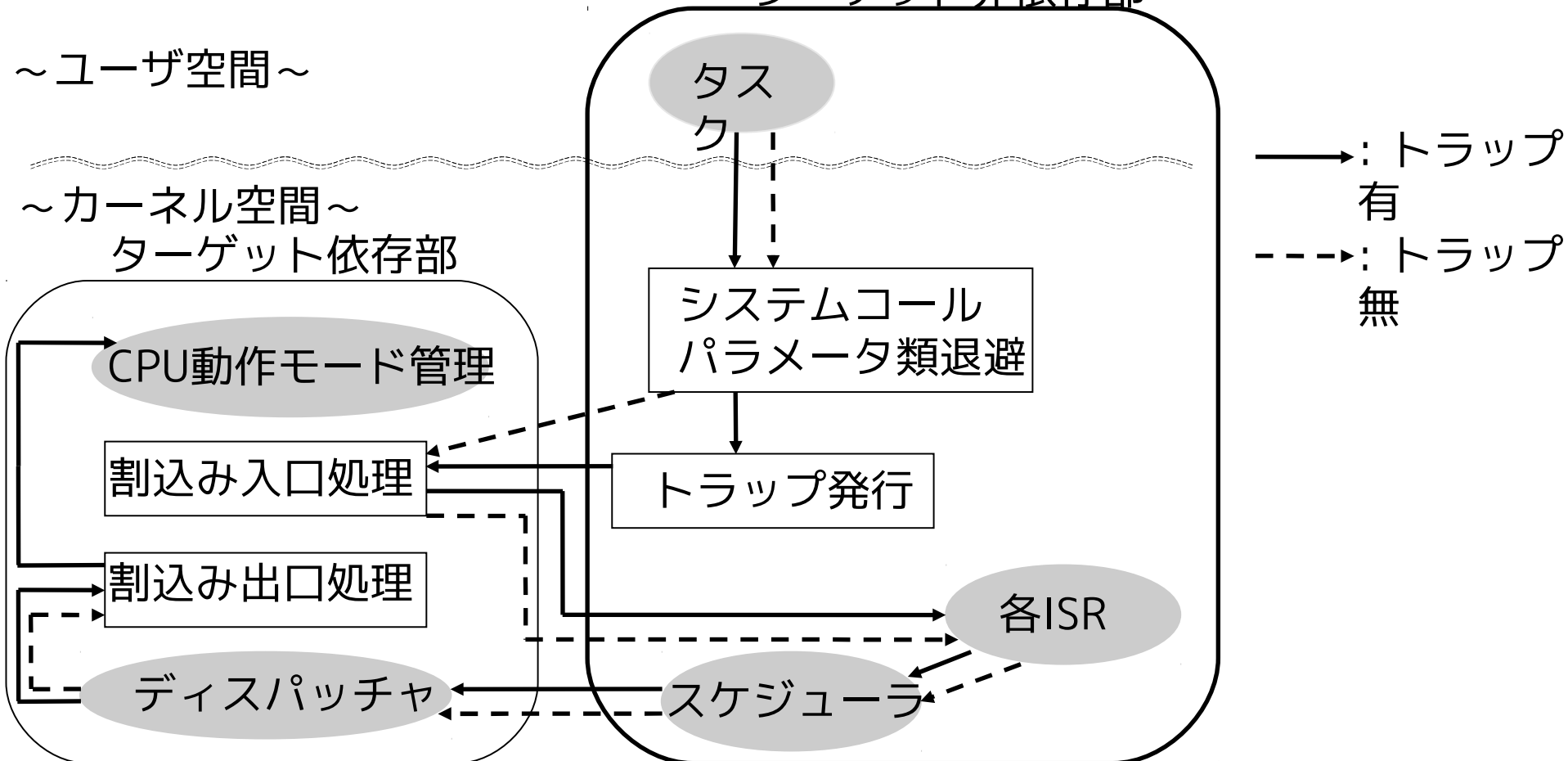


すべてのパターンをサンプルタスクセットライブラリ
を使用して動作学習が可能

複数のメカニズム：システムコールの方式

- トラップ有りのシステムコール
 - トラップ無しのシステムコール
- 2通り用意

→ トラップ無しではCPU動作モードを切替ない
ターゲット非依存部



- ~~1. 背景と問題点~~
- ~~2. 目的~~
- ~~3. 方針~~
- ~~4. 構成~~
- ~~5. 複数のポリシとメカニズム~~
6. おわりに

おわりに

成果

- 複数のポリシ/メカニズムを搭載した組み込みOSとサンプルタスクセットライブラリが実装できた

今後の課題

- 直観な学習と詳細な学習の完全対応を行う
- 複数のポリシ/メカニズムに対する動作差異の学習を可能にする