

RT-Mach における Java Virtual Machine の実装と評価

1 F-5

大谷三岳 武田正之

東京理科大学大学院 理工学研究科 情報科学専攻

1 はじめに

Java Virtual Machine においては、スレッドの実現方法について特に規定しておらず、様々な実装が考えられる。また、RT-Mach は PC/AT 互換機という手軽で安価なプラットフォーム上で動作する上に、OS レベルでスレッドをサポートしており、これを利用した実装も可能である。

本研究では、このスレッドの実行効率化のためのステップとして、Java Virtual Machine を RT-Mach の提供する RT-Thread モデルを利用して実装し、その評価を行う。

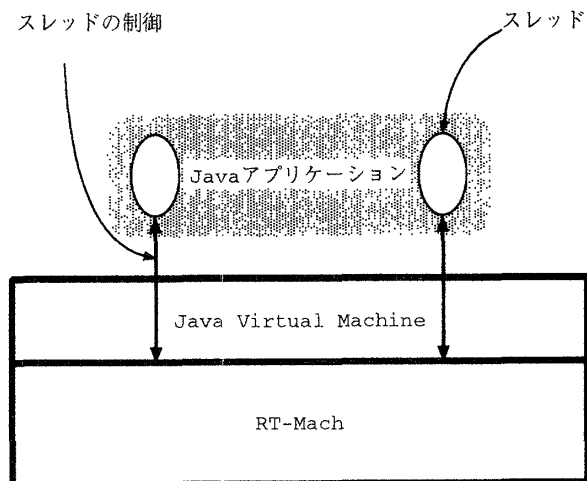


図 1: RT-Mach における Java Virtual Machine

2 動作環境

Java Virtual Machine の実装は以下の環境で行う。

ハードウェアプラットフォーム

PC/AT 互換機 (Pentium100MHz CPU)

オペレーティングシステム

RT-Mach + 4.4BSD-Lites エミュレータ

使用言語

C 言語

3 Java Virtual Machine におけるスレッド

各スレッドは thread 構造体によってその状態を保持される。スレッドと thread 構造体は 1 対 1 に対応し、各 thread 構造体は線形リストで結ばれる。thread 構造体には優先順位、動作状態 (SUSPEND, RUNNING, DEAD) などの情報を記憶しておく。

またスレッドを動作させるためには、次のような関数が必要である。

- スレッドを生成する関数、終了させる関数
(startThread, killThread)
- 動作状態を変化させる関数
(suspendThread, resumeThread)
- スケジューリングを行う関数
(scheduleThread)

これら上記の関数を RT-Thread モデルを利用して実装する。

4 RT-Thread モデルの利用

RT-Thread モデルにおいては、以下のようなライブラリ関数が提供されており、これらの関数を使用することで、スレッドの動作状態を制御するのは OS である RT-Mach となり、「動作状態を変化させる関数 (suspendThread, resumeThread)」を作成する必要がなくなる。

kern_return_t

```
rt_simple_thread_fork(func, arg, pri)
void (*func)();
int *arg;
int pri;
```

rt_simple_thread_fork はスレッドを生成し実行する。引数として func にはスレッドの実行開始点である関数へのポインタ、arg はその関数に与える引数 (1つだけ)、そしてスレッドのプライオリティ pri を与える。

kern_return_t

```
thread_terminate(target_thread)
mach_port_t target_thread;
thread_terminate は target_thread で与えられたスレッドを強制的に終了させる。
```

kern_return_t

```
rt_set_scheduling_policy(policy)
int policy;
rt_set_scheduling_policy はスケジューリング・ポリシーを policy に設定する。スケジューリング・ポリシーについては、以下のような値を設定でき、それぞれの場合について評価を行う。
```

- SCHED_POLICY_MKTIMESHARE
(mach timesharing)
- SCHED_POLICY_FIXEDPRI_RR
(round-robin fixed priority)
- SCHED_POLICY_FIXEDPRI_FIFO
(FIFO fixed priority)
- SCHED_POLICY_RATE_MONOTONIC
(rate monotonic)

- SCHED_POLICY_DEADLINE_MONOTONIC
(deadline monotonic)

- SCHED_POLICY_EARLIEST_DEADLINE_FIRST
(earliest deadline first)

5 まとめ

Java Virtual Machine のスレッドの効率化を目指して、RT-Mach の提供する RT-Thread モデルを利用した実装と評価を試みてきた。

今の段階では、評価データの分析が不十分で具体的な結果を示せないが、今後は評価データを基に、より効率的なスレッドの実装方法を検討していく予定である。

参考文献

- [1] Real-Time Mach 3.0 User Reference Manual, School of Computer Science Carnegie Mellon University, 1995
- [2] The Java Virtual Machine Specification, Sun Microsystems Computer Corporation, 1995
- [3] OS の基礎と応用, A.S.Tanenbaum=著/引地信之, 引地美恵子=訳, トッパン, 1995