3A-2

マルチコア対応組込みLinuxにおけるリソース制限方式の検討

勅使河原　佑美　　茂田井　寛隆　　山本　整　　落合　真一　　松本　利夫
三菱電機株式会社　情報技術総合研究所

1. はじめに

弊社では、マルチコアCPU対応の組込みLinux搭載機器へ新しい機能を追加するために、3rd Partyアプリの利用を検討している。

従来のシステムでは、自製アプリ(主機能)のみが動作していた。しかし、新しい構成では、主機能と3rd Partyアプリが動作し、3rd Partyアプリによりリソースが専有される可能性があり、主機能がリアルタイム性を維持できないといった問題がある。そのため、3rd Partyアプリが使用するリソースを制限する方式を検討している。

本稿では、リソースを管理する手段として、Linuxに搭載されているリソース管理機能である「cgroups[1]」を利用可能か評価・検討した。

2. 目的

主機能のリアルタイム性確保のために、cgroups管理下プロセスのリアルタイム性を評価する必要がある。その前段階として、まずはcgroupsが持つリソース管理機能の性能評価を行う。cgroupsは、CPU、メモリ、ディスク、ネットワークのリソース使用量・割合を特定のプロセスグループに割当てることができる。本稿では、3rd Partyアプリの動作制限に最も影響の大きいと考えられるCPUリソース管理機能、特にリアルタイム優先度に関するパラメータに関して評価・検討を行う。

3. 評価

3.1. 評価内容

本評価では、リアルタイム優先度のプロセスが動作する時間を確保できるか評価する。cgroupsのパラメータの中でリアルタイム優先度に関する以下の2つのパラメータ値が、CPU使用率へどのように影響するか測定する。

- cpu.rt_period_us
  各プロセスグループに対して設定するCPUリソース制御周期時間(μ秒)
- cpu.rt_runtime_us
  各プロセスグループに設定された周期時間におけるCPU利用可能時間(μ秒)

A Study of resource management for Embedded SMP Linux
Yumi TESHIGAWARA, Hirotaka MOTAI, Hitoshi YAMAMOTO, Shinichi OCHIAI, Toshio MATSUMOTO
Information Technology R&D Center, Mitsubishi Electric Corporation

3.2. 評価環境

評価機器として、組込み向けマルチコアCPUのTI社製OMAP4430を搭載したPandaBoardを用いた。OSはUbuntu10.10を用いた。

評価に用いたプログラムは、以下の2つ。これらは各プロセスのCPU使用率を計測するためのプログラムである。

- CoreMark
  CPUコアのベンチマーク。1秒間のベンチマークの繰り返し回数を計測する。
- loop
  無限ループ処理の書かれたプログラム。各グループのCPU利用率を最大まで上げる。

本評価は図1の構成で行う。cgroupsはプロセスのように階層的に構成されており、子は親の属性を受け継ぐ。親のグループ「cg」の下に子のグループ「cg0」「cg1」「cgA」を作成した。マルチコア構成上で、cg0をコア0に、cg1とcgAをコア1に割り当てた。これは、cg0とcg1にて主機能、cgAにて3rd Partyアプリのリソース管理を行うことを想定している。cgAと同じコア上で動作するプログラムがcgA上で動作する3rd Partyアプリの影響を受ける可能性がある。そのため、3rd Partyアプリの影響を受けさせたくない主機能はcg0、影響を受けても大丈夫な主機能はcg1に割り当てている。

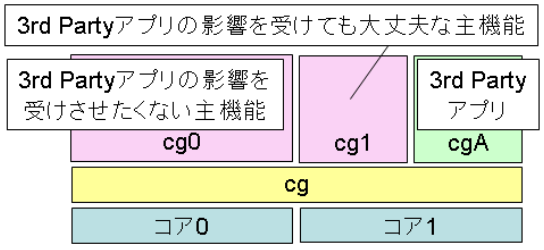

図 1：cgroupsの構成

以下の2点を計測する。

ア. cg0とcg1上でリアルタイム優先度を設定したloopを実行しながら、cgA上でリアルタイム優先度を設定したCoreMarkを実行し、1秒間のベンチマークの繰り返し回数を取得する。

イ. cgAとcg0とcg1上でリアルタイム優先度を設定したloopを実行し、topコマンドで各プロセスのCPU使用率を測定する。

3.3. 評価結果

表 1は各グループの cpu.rt_period_us、cpu.rt_runtime_us の設定値、表 2の CPU 使用率 C は、表 1の設定値で、3.2節のアを実行して得た値である。表 2の CPU 使用率 T は、表 1の設定値で、3.2節のイを実行して得た値である。No.1～3 は、cgA/cg0/cg1 の CPU の割当てを指定することが可能かを評価するために、cgA/cg0/cg1 の cpu.rt_runtime_us 値の比率を変更して測定した。No.4～8 は、cpu.rt_period_us を評価するために、cgA/cg0/cg1 への設定割合は同じで周期が違う設定で測定した。

表 1：評価の設定値

No.	cpu.rt_period_us	cpu.rt_runtime_us		
		cg0	cg1	cgA
1	1,000,000	425,000	425,000	100,000
2	1,000,000	300,000	300,000	350,000
3	1,000,000	225,000	225,000	500,000
4	1,000,000	25,000	25,000	900,000
5	100,000	2,500	2,500	90,000
6	10,000	250	250	9,000
7	2,000	50	50	1,800
8	1,000	25	25	900

表 2：評価結果

No.	CPU 使用率 C(%)	CPU 使用率 T(%)		
	cgA	cg0	cg1	cgA
1	15.2	84	82	19
2	58.3	60	43	57
3	81.7	45	19	81
4	100.0	5	1	100
5	100.0	5	0	100
6	99.8	5	0	100
7	99.4	5	0	100
8	98.8	5	0	100

4. 考察

〈CPU 使用率〉

表 2の CPU 使用率 T の cg0 の値から、nをコアの数とすると、CPU 使用率は、

$$\text{CPU使用率} \approx \frac{cpu.rt\_runtime\_us}{cpu.rt\_period\_us} \times 100 \times n \ldots \text{式①}$$

という関係にあることが分かった。ただし、cg1 と cgA の値から、同じコア上にある cg1 と cgA の値を使った式①の結果をそれぞれCPU_1、CPU_Aとした場合、コア 1の使用率の総和は、

$$CPU_1 + CPU_A > 100$$

となり、それぞれの CPU 使用率 T は、CPU_1、CPU_Aと異なることが分かる。例えば、No.3 の場合は、$CPU_1 = 45$、$CPU_A = 100$となり、$CPU_1 + CPU_A = 145 > 100$である。測定した CPU 使用率 T はそれぞれ 19%、81%となっており、CPU_1、CPU_Aと異なる。また、No.1～3 より、$CPU_1 + CPU_A - 100$の値が大きくなるほど、CPU 使用率 T と式①から求めた値の差が大きくなる。よって、期待通りの CPU 使用率に制限するには、$CPU_1 + CPU_A \leq 100$となるように値を設定する必要がある。

〈スケジュール周期〉

表 2の CPU 使用率 C の No.4～8 の値では、cg1 にはほとんど CPU 時間が割り当てられていないにも関わらず、cpu.rt_period_us、cpu.rt_runtime_us の値が小さくなるほど、CPU 使用率 C が下がっていることが分かる。これは、cpu.rt_period_us が CPU サブシステムのスケジューリング間隔(μ秒)であるため、値が小さいほどスケジューリング頻度が増加し、その分 CPU を使用するためである。よって、本評価の環境では、cpu.rt_period_us の値を 100,000 以上(CPU 使用率 C が 100%)にしておくと設定通りの CPU 使用率に制限できることが分かった。

5. まとめ

本評価により、以下の 2 点が分かった。

1. コア数、cgroups のパラメータ値と CPU 使用率は、式①の関係にある。ただし、同じ CPU 上にあるプロセスグループにおいて上記の式により求めた CPU 使用率の合計が 100 を超えないように設定する必要がある。
2. cpu.rt_period_us、cpu.rt_runtime_us の値が小さくなるほど、CPU 使用率が下がる。本評価の環境では、cpu.rt_period_us の値を 100,000 以上に設定すると CPU 使用率に影響を与えないことが分かった。他の環境では値が変わると考えられ、環境ごとに調べる必要がある。

以上の点を考慮すれば、cgroups により、リアルタイム優先度が設定された主機能の CPU 時間を確保することができることが分かった。

今後の課題として、他の管理性能(特にリアルタイム性に関するパラメータ)を評価した後、cgroups 管理下におけるプロセスのリアルタイム性を詳細評価していく予定である。

参考文献

[1] Paul Menage: “Linux Kernel Documentation/cgroups/cgroups.txt” http://www.kernel.org/doc/Documentation/cgroups/cgroups.txt(2011).