CE Linux Forum

igel

Linux リアルタイム機能の検討
- ユーザレベルでデバイスドライバは実現可能か？ -

松原 克弥
株式会社イーゲル

funded by 株式会社ルネサスソリューションズ

igel

自己紹介

- 松原　克弥(まつばら　かつや)
  - 株式会社イーゲル
    - 組み込み分野におけるLinuxソリューションのコンサルティング
    - 組み込みOS(Linux, NetBSD, T-kernel, Windows)上のデバイスドライバからミドルウェアまでの設計・開発
    - http://www.igel.co.jp/
  - IPAセキュリティセンター CODE Blogプロジェクト
    - オープンソース・ソフトウェアを使って、ソースコードを読む技を調査・解析するプロジェクトです。
    - https://www.codeblog.org/
  - 慶應義塾大学 環境情報学部
    - オープンソース・ソフトウェアを利用して、コンピュータリテラシとプログラミング基礎を教えています。

igel

背景

- オープンソースの魅力
  - コードが読める＝疑問に思えば解析できる、ほしいと思えば改造できる。
  - 開発者と直接コミュニケーションできる。
  - コードに関連した情報も交換・共有（インターネットの発展も寄与）
- 組み込み系開発でも、疑問に思っていること、あったらいいなと思っていることを共有・解決したい‥
- CELFは組み込みLinux技術者の情報交換・共有の場
  - 「Linuxって、応答性はどの程度保証できるの？」
  - 「カーネルプログラミングって難しいよね。」
- 第5回テクニカルジャンボリーで飛び入り発表、以降、ほぼ毎回進捗報告を行っている。
  - 多くの技術者で同様の疑問を持っていることを認識
  - 「ARMではこうしているよ」「ここの関連情報があったよ」「USBやWLANをのドライバが参考になるんじゃないかな」などのさまざまな情報が得られる。

igel

目的とissues

- ユーザレベルでデバイスドライバを実現したい
  - 開発が容易
  - ドライババグによるシステムダウンを軽減
  - より密接なアプリとの連携
- いくつかの問題
  - I/Oメモリ、物理メモリへのアクセス
  - 割り込み要求(IRQ)の受信
  - 応答速度
  - ..
- カーネル2.6の新機能
  - NPTL(Native POSIX Thread Library)
  - スケジューラの改善(O(1)スケジューラ等)
  - カーネルプリエンプション

igel

これまでの過程

1. スレッドの応答性調査
2. ULDDフレームワークの設計
3. ULDDの実装
4. Linuxリアルタイム対応の調査

igel

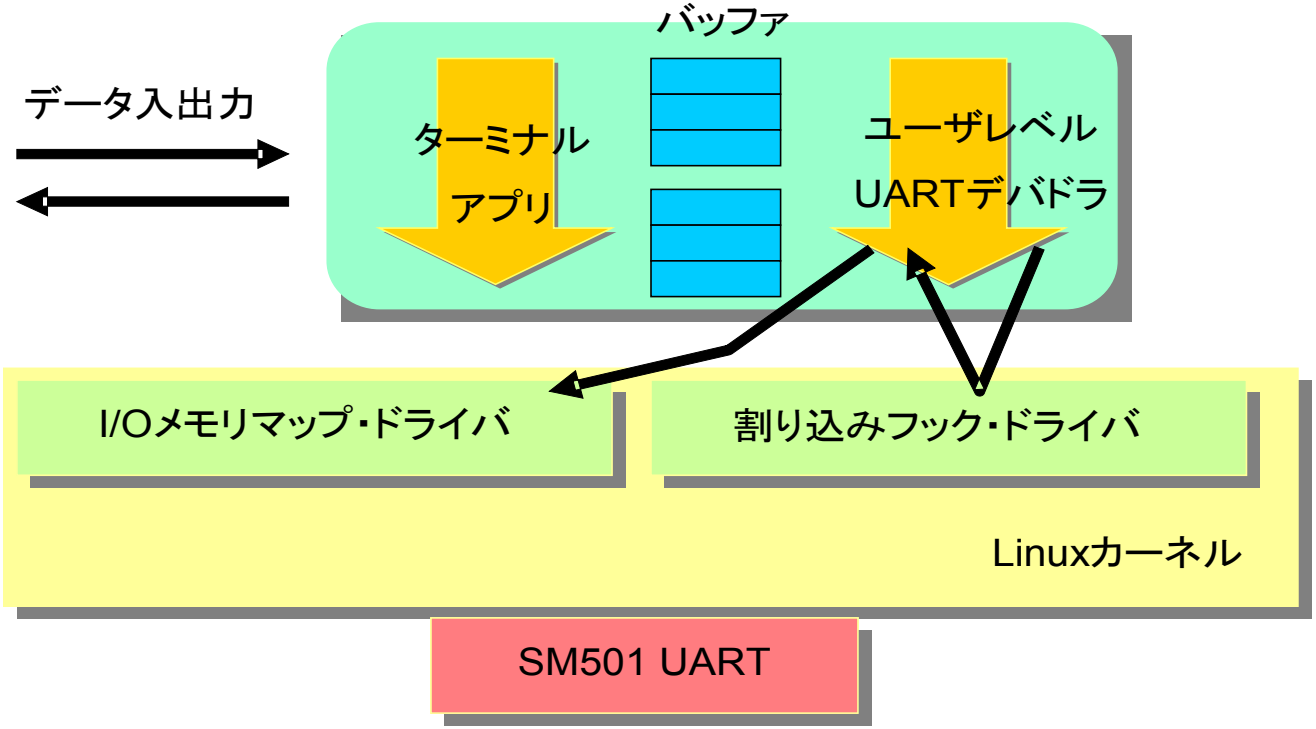
1. スレッド応答性

- スケジューラ遅延
- カーネル側でタスクをwakeupしても、すぐに実行が再開されるわけではない。→**スケジューラ**が次に再開するタスクを決定
- スケジュールポリシーは複雑
  - 通常（non-RT）タスクの動的優先度
    - スリープ時間が影響
    - タイムスライス
  - Rtタスクはの優先度は静的、かつ、常ににnon-RTより優先

igel

2. ULDDフレームワーク

- Peter Chubbの論文がベース
  - Peter Chubb, “Get more device drivers out of kernel,” OLS2004.

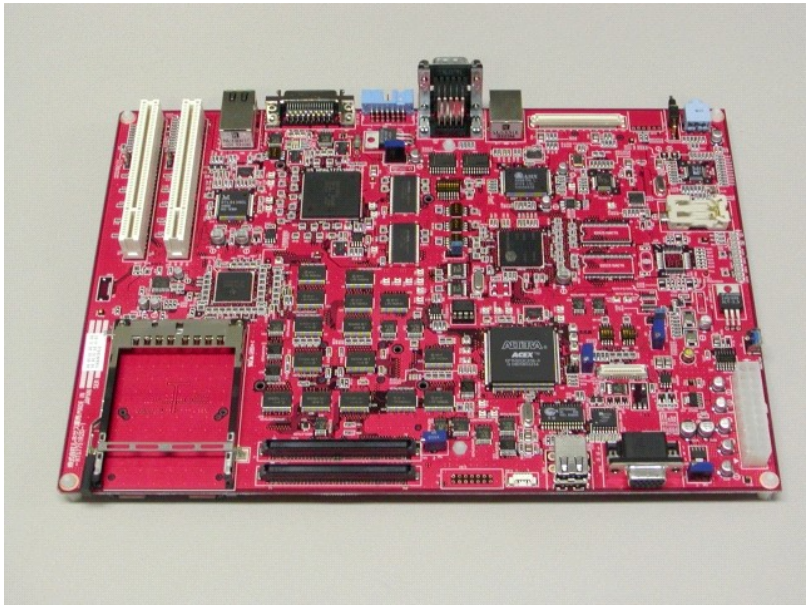


igel

3. ULDDの実装

- SM501 UARTデバイスドライバ
  - 8250コンパチブル
  - バイト転送とFIFOモードをサポート(本実装はバイト転送)

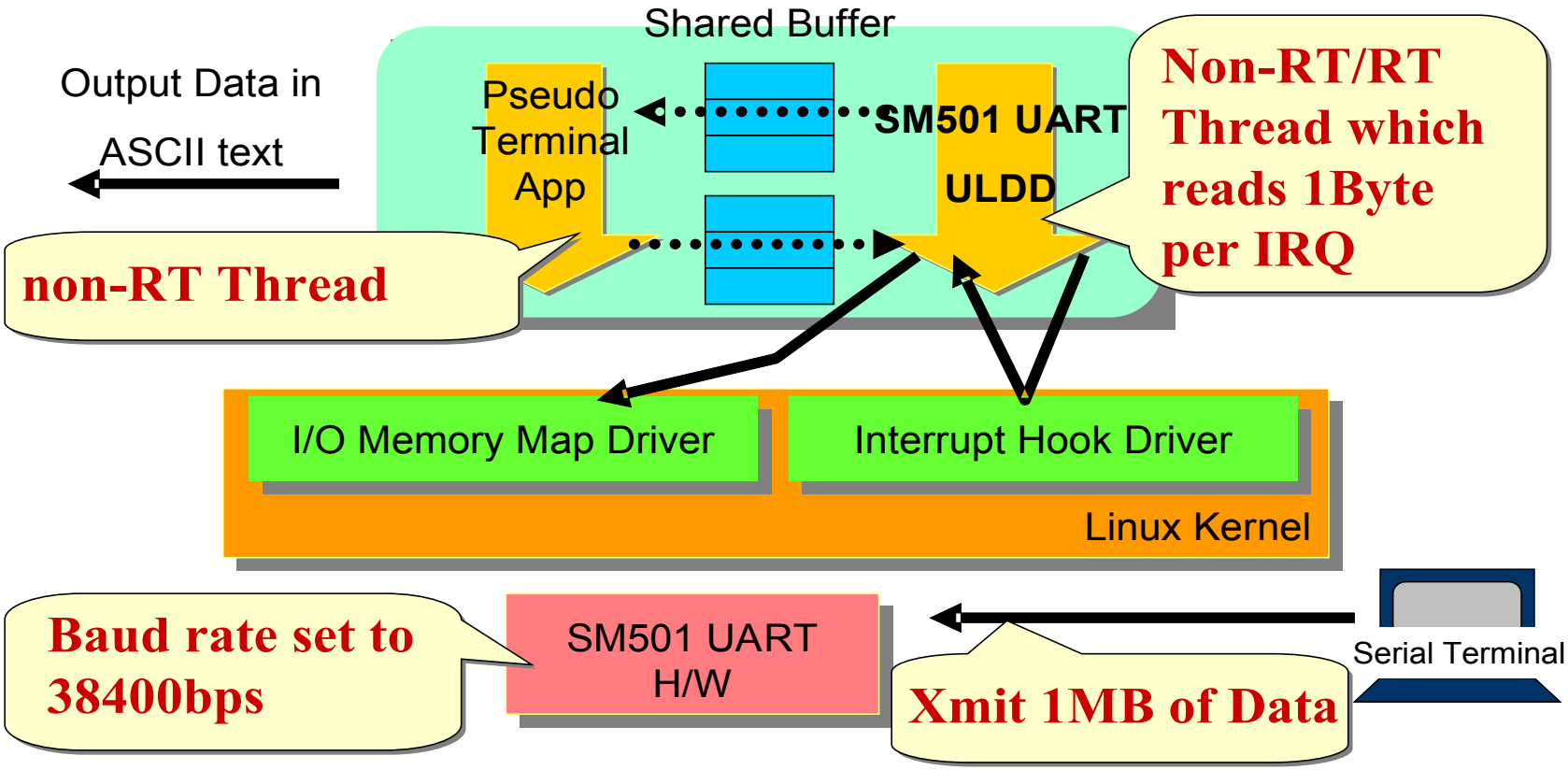
- RTS7751R2D評価ボード
  - Renesas SH7751R(SH-4) 240MHz
  - 64MB RAM
  - 100Mbps Ethernet
  - PCI Bus
  - ...



igel

実装を用いたULDDの評価

- ULDDのオーバヘッド＝スケジュール遅延を測定



igel

スケジュール遅延

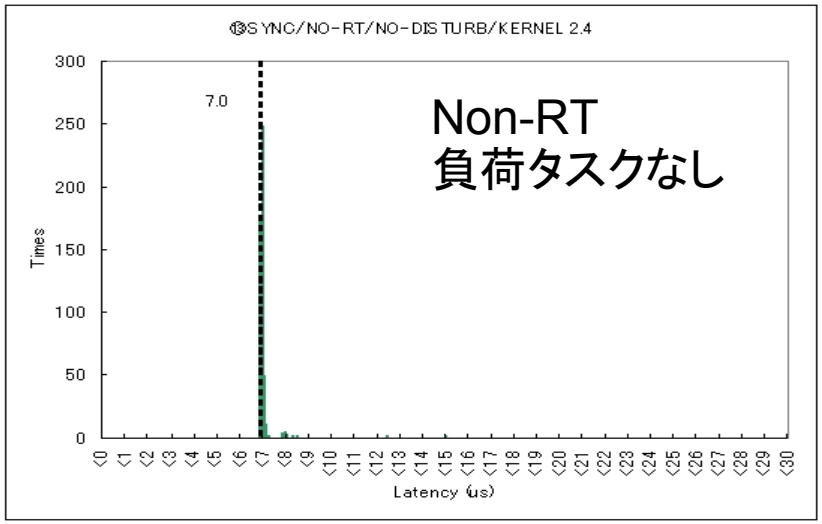
割込通知実現方法	スケジュールポリシー	負荷タスク	Linux 2.6.16.4 (CONFIG_PREEMPT=y)			Linux 2.4.20		
			MAX (us)	AVG (us)	MIN (us)	MAX (us)	AVG (us)	MIN (us)
File I/O (sync)	SCHED_OTHER	(none)	39.80	17.52	13.73	14.93	6.98	6.83
	SCHED_OTHER	lat_proc	253.27	42.57	10.93	2808.13	478.43	6.53
	SCHED_RR	(none)	38.13	15.73	14.07	17.13	6.34	6.20
	SCHED_RR	lat_proc	33.73	14.32	8.80	2697.13	485.29	6.60
SIGIO (async)	SCHED_OTHER	(none)	524.00	62.84	45.47			
	SCHED_OTHER	lat_proc	72317.40	2185.96	83.33			
	SCHED_RR	(none)	81.40	47.12	43.67			
	SCHED_RR	lat_proc	280.93	102.89	79.40			
SIGRT (async)	SCHED_OTHER	(none)	83.47	50.00	44.80			
	SCHED_OTHER	lat_proc	85973.93	2528.19	85.40			
	SCHED_RR	(none)	75.93	46.71	43.07			
	SCHED_RR	lat_proc	290.80	102.18	78.87			

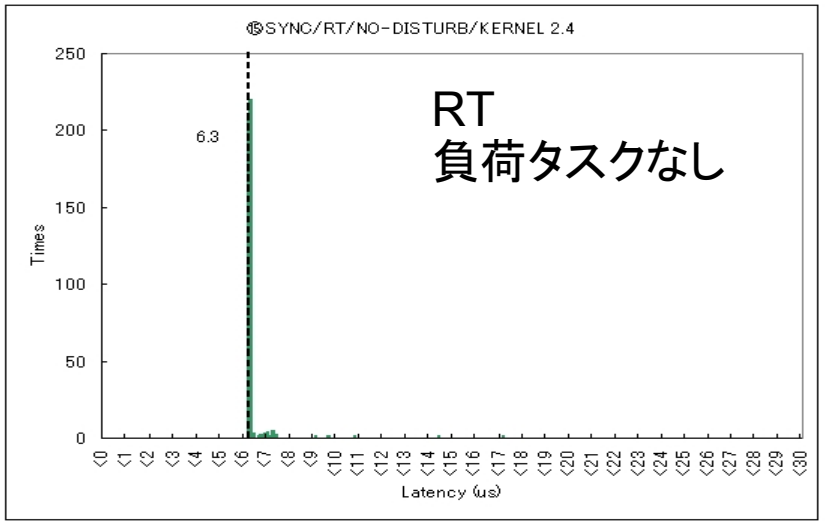
SM501 UART ULDDコード，実験プログラム，結果は下記で公開

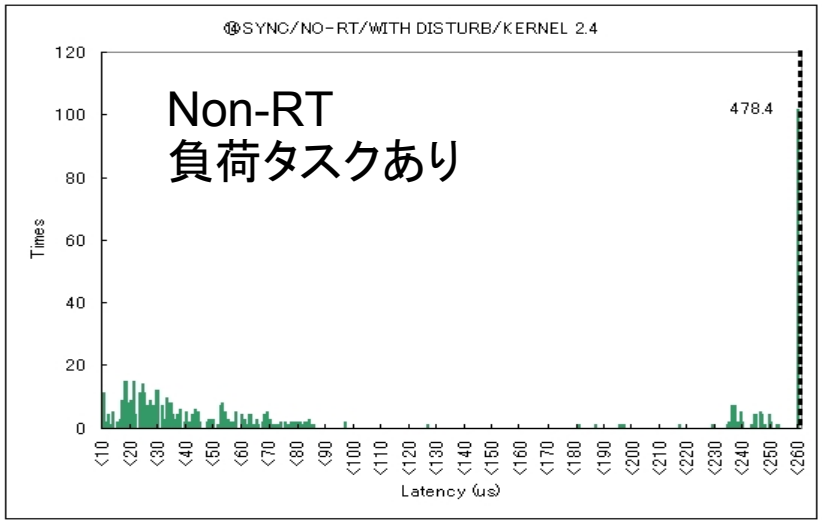
http://tree.celinuxforum.org/CelfPubWiki/UserLevelDeviceDriver

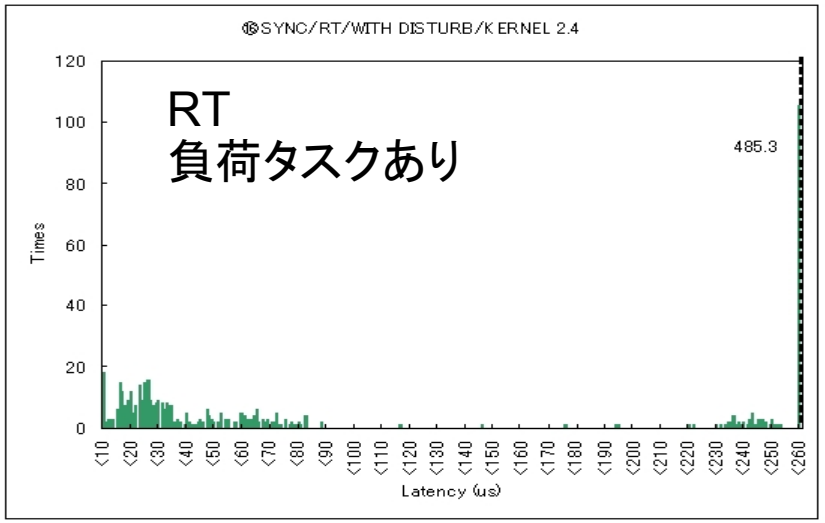
igel

Linux 2.4におけるスケジュール遅延



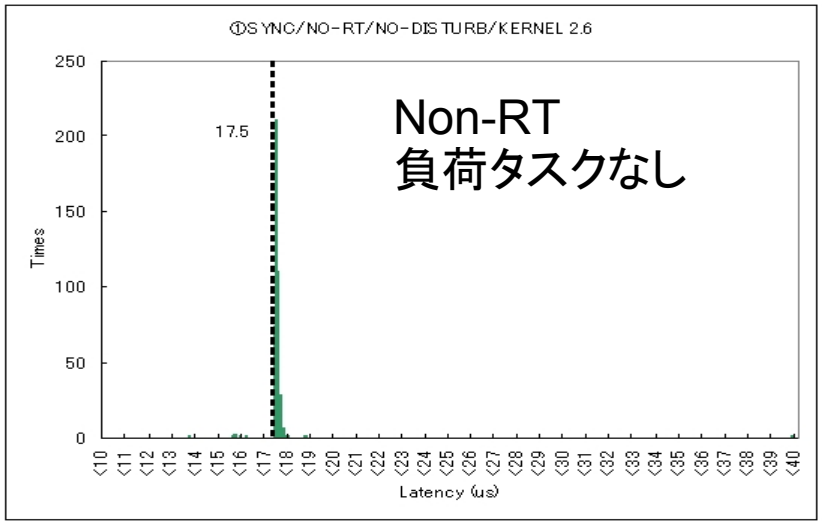


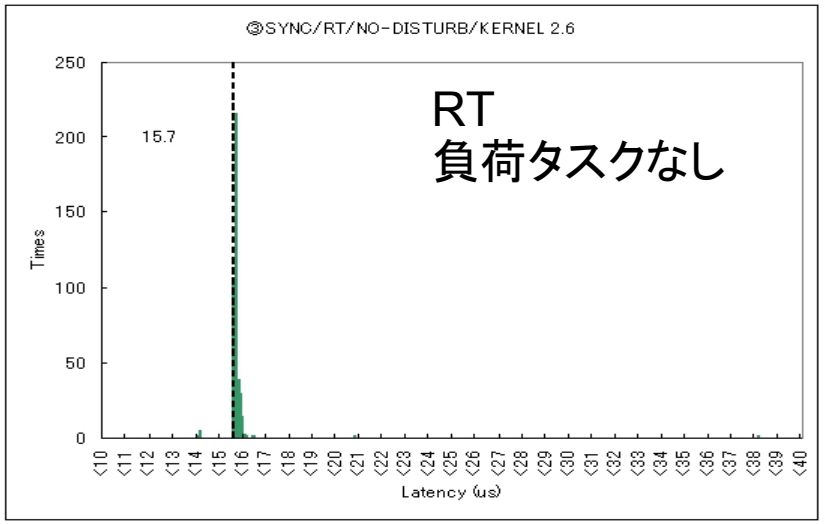


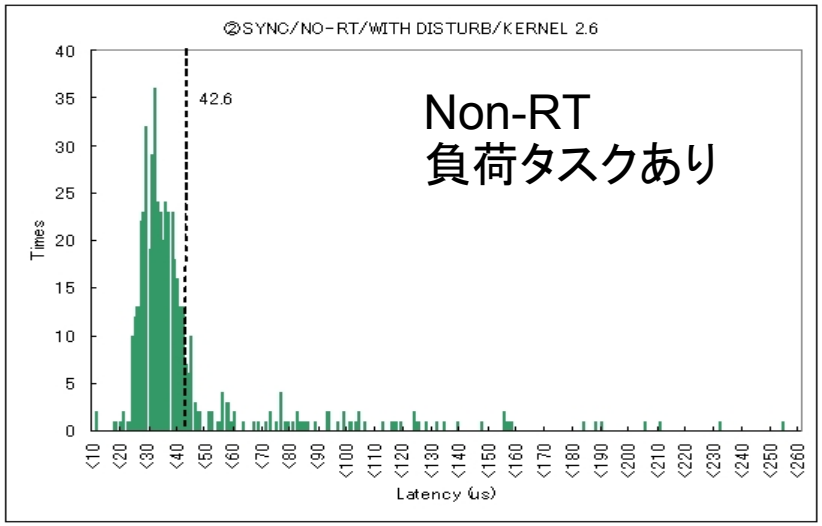


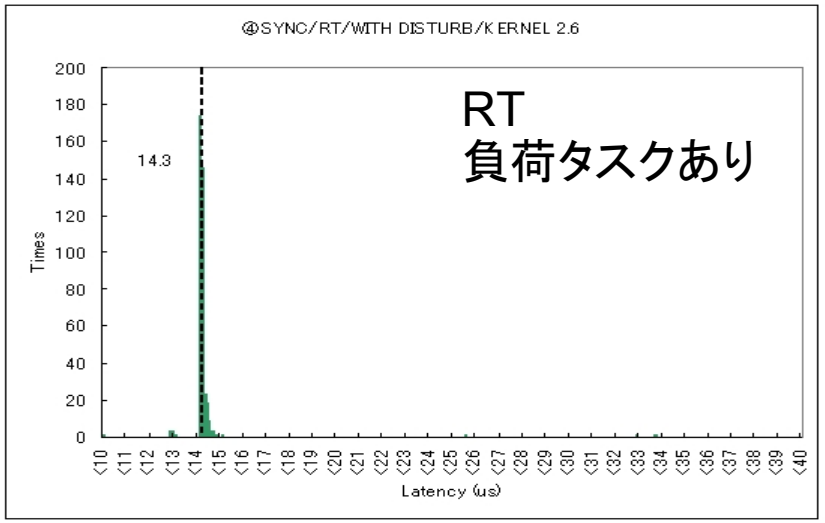
igel

Linux 2.6(w CONFIG_PREEMPT) におけるスケジュール遅延









igel

4. Linuxのリアルタイム対応

- 完全リアルタイム
  - 全タスクのデッドラインを保証 ⇒ (非リアルタイム)既存OSへの接木では困難
  - Nested OS, Dual OS/Dual Core, ART-Linux
- 部分リアルタイム
  - 一部のタスクのデッドラインを努力保証
  - たいていのリアルタイムタスクは限られた資源のみを使用する。

  ⇒遅延を最小に ＝プリエンプション

  ⇒ 遅延度合いをより小さく ＝ リアルタイムプリエンプション
  - CONFIG_PREEMPT, CONFIG_PREEMPT_RT

igel

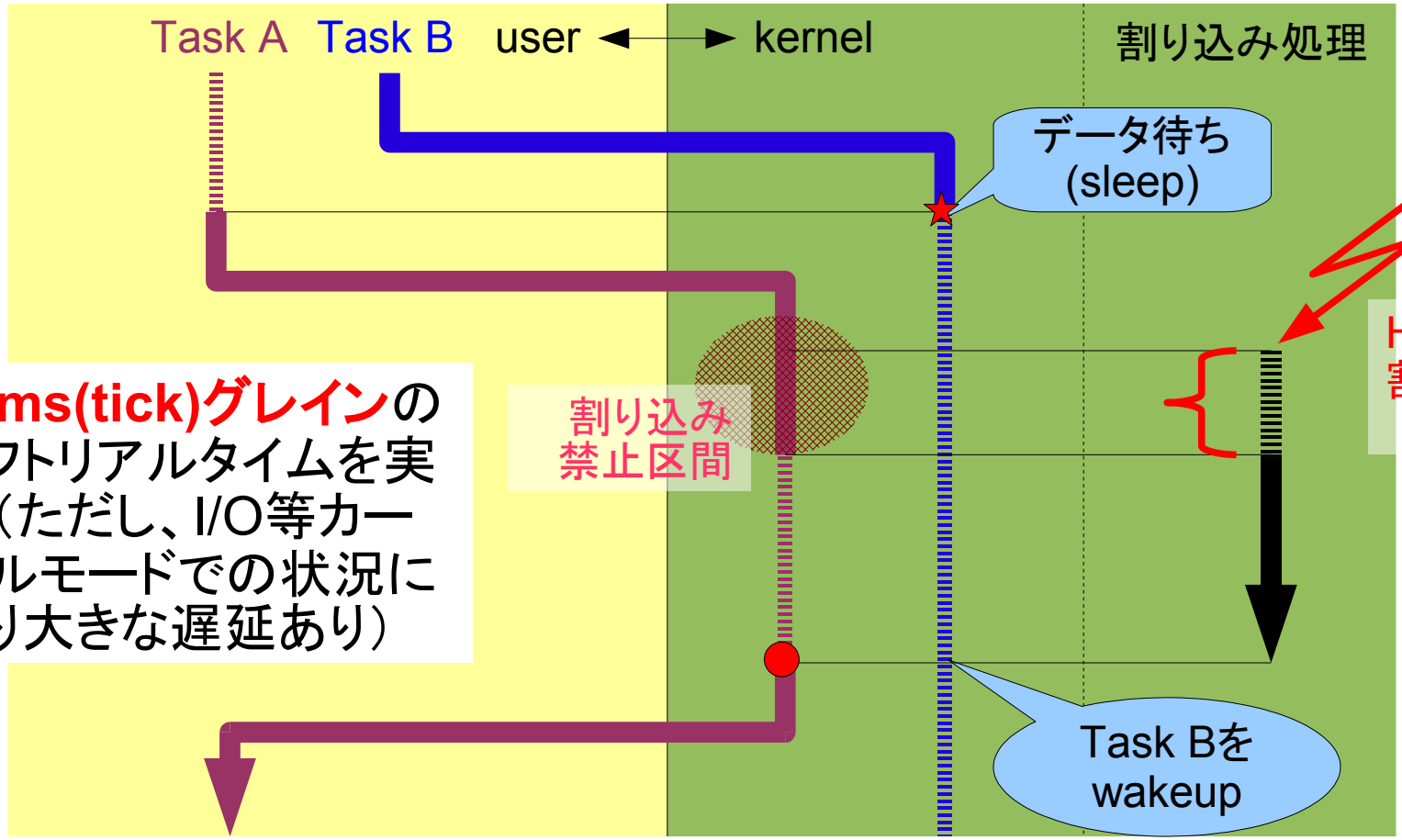
Linuxのプリエンプション

- プリエンプションのタイミング
  - 割り込み処理からの復帰時
  - システムコールからの復帰時
  - タスクが自主的に休眠した時

  ⇒リターン時に別タスクのスタックポインタをセットし、pop
- 2.4 / CONFIG_PREEMPTなしの2.6では、カーネルモードではプリエンプション(タスク切り替え)が起きない(割り込み処理は起きる)
  - カーネルモード時の割り込みからの復帰は元の場所に戻る(プリエンプション非対応コードのため)

igel

2.4 / 2.6 without CONFIG_PREEMPT



10ms(tick)グレインの
ソフトリアルタイムを実現（ただし、I/O等カーネルモードでの状況により大きな遅延あり）

igel

割り込み処理における遅延

- 割り込みハンドラ起動遅延
  - 割り込みが発生してからハンドラが起動されるまで

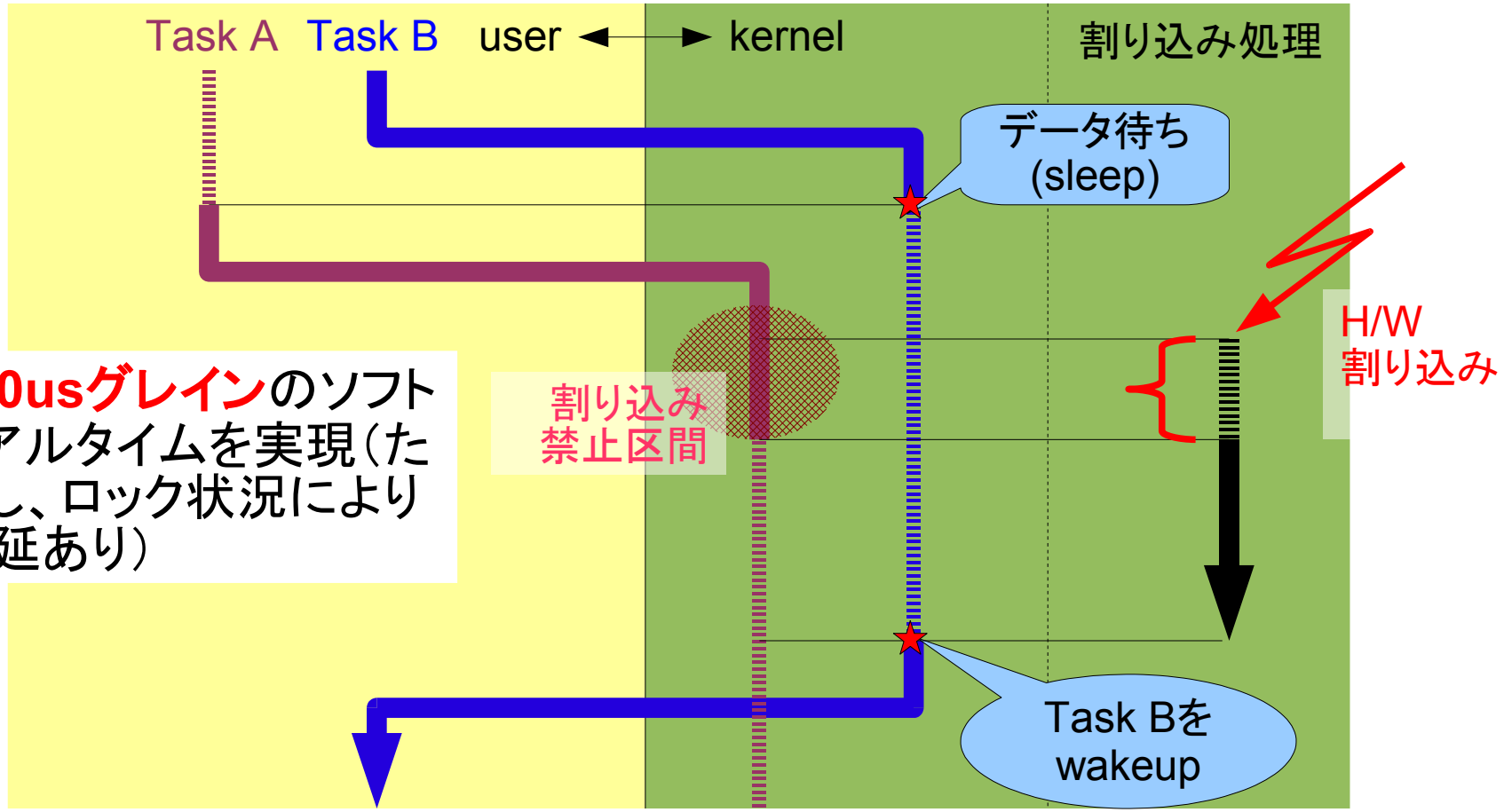
  ⇒ 割り込み禁止区間が影響
- タスク起床遅延
  - 割り込みハンドラがタスクの起床を指示してから、実際に起床するまで

  ⇒ プリエンプション禁止区間(割り込み禁止区間を含む)が影響

  ⇒スケジューリングポリシーが影響

igel

CONFIG_PREEMPT



100usグレインのソフトリアルタイムを実現（ただし、ロック状況により遅延あり）

igel

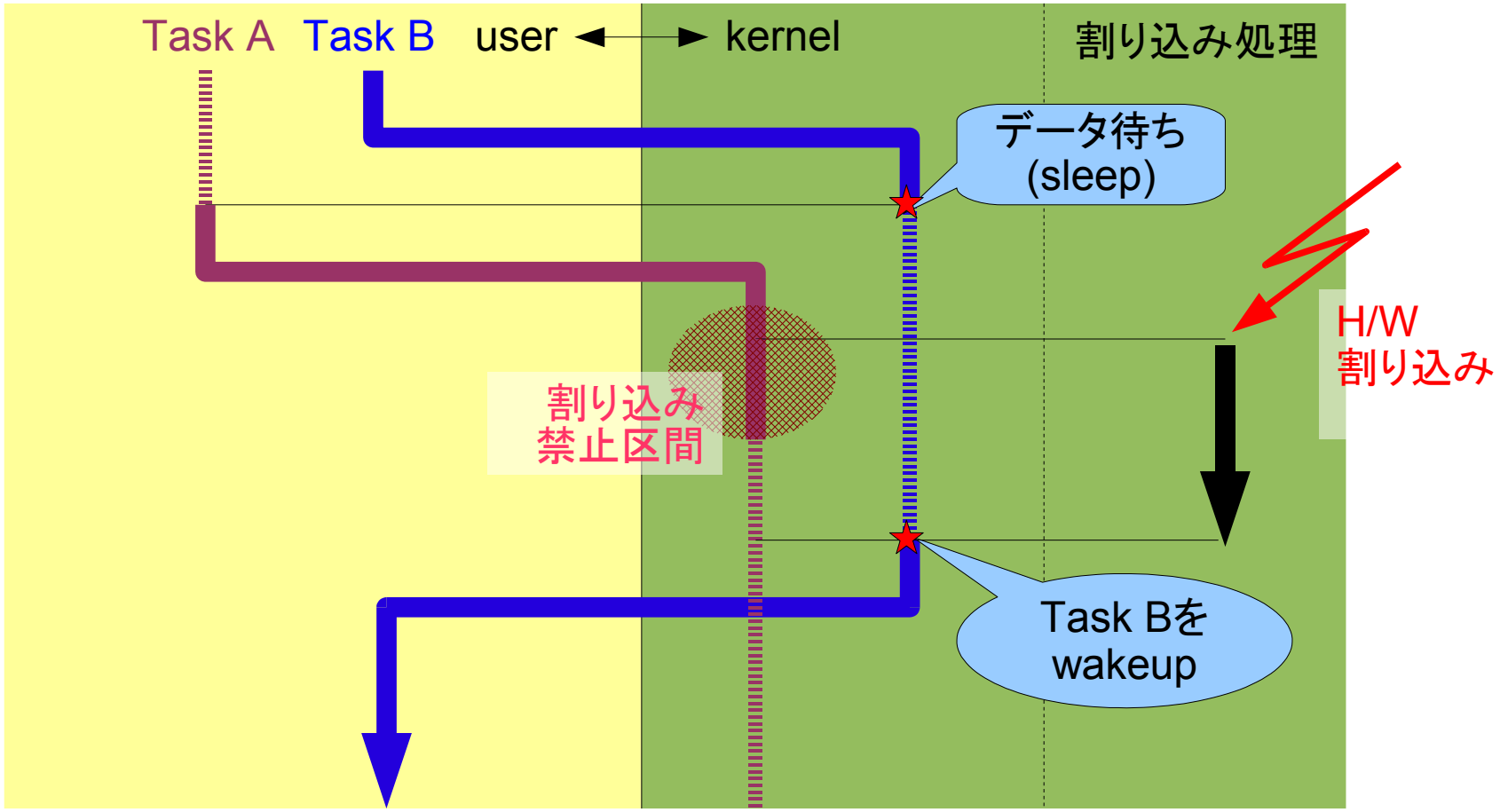
(CONFIG_)PREEMPT_RT

Ingo MolnarらのLinux realtime preemptionパッチ

- プリエンプションの機会を、割込み処理中やクリティカル領域内にも増やすことで遅延を軽減し、タスク起床および割り込み処理において、20～30usのソフトリアルタイムを実現
- 優先度逆転を防ぐため、優先度継承ロックを提供
- 他コード部分のリファクタリング・最適化
- http://people.redhat.com/mingo/realtime-preempt/

igel

PREEMPT_RT



igel

PREEMPT_RT により追加・変更される特徴

i. クリティカル領域内でのプリエンプション
ii. 割り込み処理内でのプリエンプション
iii. 割り込み禁止コード領域内でのプリエンプション
iv. スピンロックとセマフォの優先度継承
v. スピンロックが必要なプリエンプション禁止コードの処理
vi. その他の最適化

igel

i. クリティカル領域内でのプリエンプション

- spinlock_t, rwlock_t, rcu_read_lock() and rcu_read_unlock()、セマフォで囲まれた領域内のプリエンプション可能に
- spin_lock_irqsave()等*_irq()は、実際にはハードウェア割り込みを禁止しない
- 従来のスピンロックはraw_spinlock_tを使うことで継続可

igel

ii. 割り込み処理内でのプリエンプション

- 割り込み処理はプロセスコンテキストで実行
  - redirect_hardirq()によりirqdに処理をリダイレクト
  - プリエンプション可能
- SA_NODELAY宣言された割り込み処理のみ、割り込みコンテキストで実行
  - 例）CPUタイマ割り込み、fpu割り込み等
  - ただし、起床遅延に影響の可能性あり
  - コーディングに最新の注意が必要＝できるだけ使わない

igel

iii. 割り込み禁止コード領域内でのプリエンプション

- 「割り込み禁止」なのにプリエンプション？！
  - SMPでは、複数のCPUにより割り込み処理の同時進行が可能に
- spin_lock_irqsave()はプリエンプションを禁止しない。
  - 大丈夫？

  ⇒別の割り込み処理がスタートしても、同じロックを取りにいくところでブロック ＝ クリティカル領域は設定可
- local_irq_save()はプリエンプション禁止
  - 対応するロックがない
  - ロックを使うほうが遅延を減らせる。ただし、SMPの性能低下の可能性あり

igel

iv. カーネル内スピンロックとセマフォの優先度継承

- ロック保持タスクの優先度をロックを必要とする高優先度タスクに合わせてboostする。
- 優先度継承は繰り返し行われる(推移する)。
- 「プリエンプションポイント」を設定して、そのポイントでロックを手放すこともできる(例:JBDジャーナルレイヤ)。
- Writerからreaderへの優先度継承は難しい。
  - PREEMPT_RTでは、readロックは一度に1タスクのみが保持できるような制限を加えている。
- セマフォは(ロックではなく)イベント機構を用いているため、優先度継承が難しい。
  - Postするタスクを特定できない
  - compat_semaphore() and compat_rw_semaphore()を提供 (未調査 m(__)m)

igel

PREEMPT_RTの移植

- 現在サポートされているアーキテクチャ
  - Intel x86
  - PowerPC
  - ARM
  - MIPS
  - 等
- 新しいアーキテクチャの追加
  - arch/*アーキテクチャ*、include/asm-*アーキテクチャ*内の変更
    - 割り込み禁止、プリエンプション禁止にすべきロックやセマフォに「raw_」「compat_」を付与
  - タイマハンドラ等の割り込みコンテキストで実行すべき割り込みハンドラの登録にSA_NODELAYを指定
  - その他、アーキテクチャ別(if defined(*アーキテクチャ*))の処理を追加

igel

新しいアーキテクチャ対応の手間

- 簡単に対応させるだけなら、すべてのロックやlocal_irqsaveに「_raw」、セマフォに「_compat」をつければ動く(はず)。
  - プリエンプションの機会が増えない＝意味なし
- 各クリティカル領域、割り込み処理でプリエンプション禁止の必要性を解析する必要がある。

igel

ULDDへの影響

- 予測
  - 割り込み禁止領域の削減
  - プリエンプション禁止領域の削減

  ⇒プリエンプションの機会が増えるので、スケジュール遅延が少なくなりそう。
  - 優先度継承つきロック

  ⇒優先度逆転が起きなくなるので、より遅延のブレが小さくなるはず。
- 実際に測定してみると‥

本日はここまでです。m(__)m

igel

まとめ

- Linux 2.6において、遅延に対する対策行われ、リアルタイム性の改善が見られる。
  - O(1)スケジューラ
  - NPTL
  - CONFIG_PREEMPT / CONFIG_PREEMPT_RT
- スケジュール機構は複雑だが、きちんと理解することで得られる効果も大きい。
  - 厳密な優先度設定を行わなくても、スループットと遅延、各プロセスに対する公平性を考慮したスケジューリングが行われる。
- ULDDの実現性・有効性は低くない。
  - ただし、すべてのKLDDが置き換えられるではない。
  - 特徴を理解し、最適な機構を準備・選択し、適切なところで使用することが現実的である。

igel

状況

- Linux 2.6.16
  + linuxsh patch
  + RTS7751R2D用パッチ
  + patch-2.6.16-rt29
  - raw_, compat_を追加＋αは動作(Thanks to Lineo)
  - チューニング(不必要なraw_, compat_の削除)中
- ULDD
  - H/W割り込みをユーザタスクに通知するドライバは実装済み
  - SM501 UARTドライバ on RTS7751R2D は実装済み
  - 外乱プロセスの選定中(lat_proc in Lmbenchは、Linux プリエンプションの効果が見られないようだ)

igel

TODO

- PREEMPT_RTの移植
  - SH4に対応
  - 2.6.18への移行？
- Intel x86との比較
  - Intel x86上のH/Wデバイスを対象とULDDの実装
- ULDD v2の設計
  - 割り込みの判別や割り込み停止等の基本処理をカーネル側で行う拡張
  - カーネルドライバでのAPIとの差異を減らす
  - wlan, libusb等のすでにULDDを実現している仕組みを参考に