次世代組込みLinuxに向けて

早稲田大学　理工学部
コンピュータネットワーク工学科
中島　達夫

組込みLinux製品

IBM/Citizen
WatchPad

Sony CoCoon Channel Server

Philips iPronto
Remote Control

Panasonic Broadnow
Broadband Set-top box

Panasonic broadband
terminal phone

Terapin mine
Media Jukebox

NEC AX-10 Home AV Server

Techsan TS20000
MHP Set-top box

MasterIA Beagle
PDA GPRS Phone

Zultys ZIP 4X4
VoIP phone

TiVo

Advanced Communications
ECCLE5000 MPEG 4 Set-top Box

組込みLinuxの現状

- 多くの製品で組込みLinuxが使われ始めた。
  - 応答性、バッテリ管理、ブートタイム、メモリフットプリントなど様々な問題が解決されつつある。
    - CELFによる標準化
  - まだまだ多くの問題が残されている。
    - QOS、セキュリティ、信頼性…
    - 特にセキュリティに関しては大きな問題がある。

技術的な改善

- 応答性の改善
- ブートタイムの改善
- QOSサポート
- 信頼性、セキュリティの改善
- センサーのサポート

応答性の改善(1)

- 基本的な問題点
  - システムコールの実行中はプリエンプションすることが出来ない
  - システムコールの実行時間が長いと応答性が低下する可能性がある
    - 現状のLinuxは100ms以上応答性が低下することがある。

応答性の改善(2)

- カーネルをプリエンプタブルにする
  - 割り込みハンドラを抜けるときにより優先度の高いプロセスが存在するときはコンテキストスイッチが起きるようにする
  - SMPロックを取得中はSMPロックが解放されるまでコンテキストスイッチを遅らせる
  - 応答時間がSMPロックを取得している時間にまで短くなる
  - SMPロックを極力短くする
  - 割り込み不可の部分を出来る限り短くする

ブートタイムの改善

- 組込みシステムでは電源をONにして瞬時に立ち上がる必要がある。
  - CPUクロックのキャリブレーション
  - 存在しないデバイスのプルーブの省略
  - 出来る限り不必要な初期化を省略
  - 初期化時に必要でない処理をブート後におこなうようにする

情報アプライアンスとは？

- 情報を扱う専用化されたデバイス
  - 携帯電話，カーナビ，デジタルテレビ
  - 特化した機能を組み合わせることで様々なサービスを提供する.
    - 機能を統合するミドルウエアが重要
      - UPnP，Jini，HAVi，Speakeasy，Personal Home Server
    - マルチメディアアプリケーションの時間制約の保証
    - ユーザ入力に対する応答性の保証
  - 信頼性やセキュリティ面での要求が厳しい
  - 通信制御など厳しい応答性を要求するものがある
  - 現状のLinuxでは不十分である

現在のLinuxを用いて情報アプライアンスを構築する際の問題点

- マルチメディアアプリケーションをリアルタイムスケジューラで実行すると応答性が悪くなる。
  - リアルタイムスケジューラだけでは十分でない。
- 悪意のあるダウンロードしたアプリケーションがリソースを占有する可能性がある。
  - リソースの使用量を制約する必要性がある。
- システムがオーバーロード状態となっていることを検出することが困難。
  - アプリケーションをシステムの状況に応じて適応可能とする必要がある。

QOS Support

- 各アプリケーションに割り当てるリソースを制御する。
  - マルチメディアアプリケーションにCPUキャパシティの40%を与える。
  - アプリケーションが指定した以上のCPUを使わないように制御する。
  - リアルタイムアプリケーションがデッドラインをミスしないようにリソースを割り当てる。
  - 悪意のあるアプリケーションが使用するCPU使用量を制御する

過去のQOSサポート (1)

- リソースリザベーション
  - 各プロセスが利用可能な最悪リソース量を指定する。
  - 各プロセスのリソース使用量が指定した量を超えないように制御する（リソースアカウンティング）
  - システム全体のリソース要求が100%を超えないように制御する（アドミッション制御）
    - CMU Real-Time Mach, CMU Linux/RK，TimeSys Linux/RT
  - 基本的にはハードリアルタイムに適している
  - 使用するリソース量を見積もることが大変
  - インタラクティブプロセスの扱いが困難

過去のQOSサポート (2)

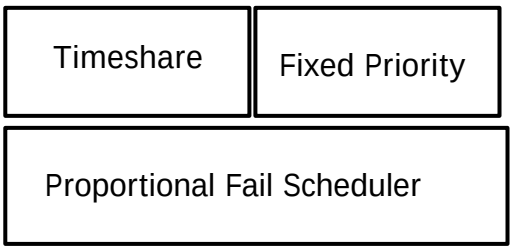
- プロポーショナルフェアスケジューリング
  - 各プロセスに重みを与える。
  - 指定した重みにした従いランラブルなプロセスに公平にリソースを割り当てる。
  - プロセスAに1，プロセスBに3の重みを与えた場合、両方のプロセスがランナブルの場合、プロセスAが25%，プロセスBが75%のリソースを利用する。
  - どちらかのプロセスがブロックしている場合は，100%のリソースを確保出来る。
    - UMASS　QLinux
  - 重みの指定が難しい。
  - 優先度の指定が難しく，マルチメディアアプリケーションなどの時間制約を持つアプリケーションの扱いが困難。

情報アプライアンスのためのQOSサポートに対する要求

- シンプルで使いやすい
  - 過去のアプローチは使用法が困難である。
- マルチメディアアプリケーションを動作させてもインタラクティブアプリケーションの応答性を低下させないようにする。
- システムがオーバーロード状態となっていることを検出可能とする。

情報アプライアンスのためのQOSサポート (1)

- 階層型スケジューラ
  - FPとTSが使用可能なCPUキャパシティに重みを与えることが出来る。（時間幅の指定も必要）
  - FPに100%の重みを与えると現在のLinuxと同じ動作をする。
  - TSに重みを与えることで応答性の低下を防ぐことが出来る。
  - 重みを設定するためのシステムコールを追加する必要がある。



情報アプライアンスのための QOSサポート (2)

- アカウンティング制御
  - プロセスの集合が指定したリソースを使用しないように制御をおこなう。
  - アカウンティングオブジェクトの生成、破壊、パラメータ設定のためのシステムコール
  - プロセスとアカウンティングオブジェクトをバインドするためのシステムコール
- オーバロード制御
  - 全体のCPU使用量が100%近くになるとアプリケーションにシグナルが送られる

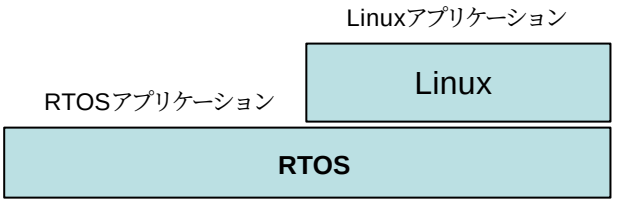
QOSサポートの実装

- int accounting_create (&object_id, &object_attr)
- int accounting_destroy (object_id)
- int accounting_object_get (object_id, &object_attr)
- int accounting_object_set (object_id, &object_attr)
- int accounting_pid_bind (object_id, &object_attr)
- int weight_ctl (weight)

- 現在モンタビスタジャパンと協力して実装を進めている
- 最終仕様に関しては日本エンベデッドリナックスコンソーシアム、リソースマネジメントWGにより検討をおこなっている

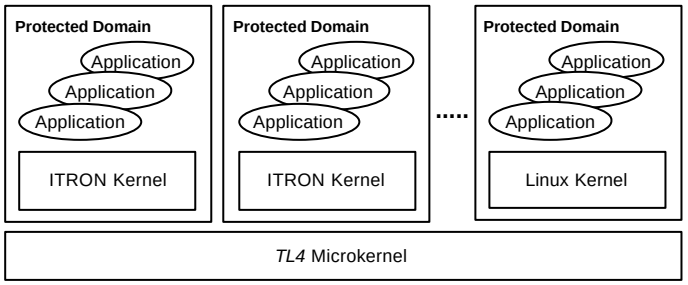
信頼性、セキュリティの改善

- RTLinux, Linux on ITRONのようなハイブリッドアーキテクチャはリアルタイムアプリケーションがクラッシュするとシステム全体がクラッシュする
- リアルタイムアプリケーションはカーネルスペースで実行するので、侵入により大きな被害が生じる
- LinuxもRTOSも1つしか動かない

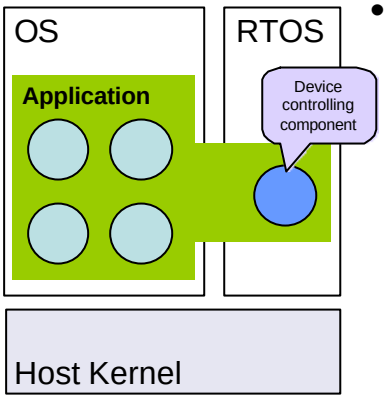


μカーネルベースのアプローチ

- マイクロカーネル上に複数のOSを起動させる
  - 各カーネルは独立のプロテクションドメインを持つ
  - マイクロカーネルは各プロテクションドメインの生成とスケジューリングをおこなう
  - カーネルAPIの変更は全くない

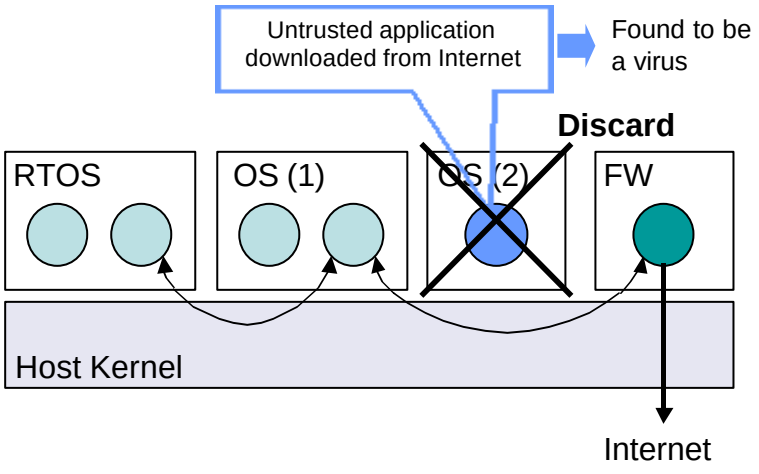


本提案の有効性 (1)

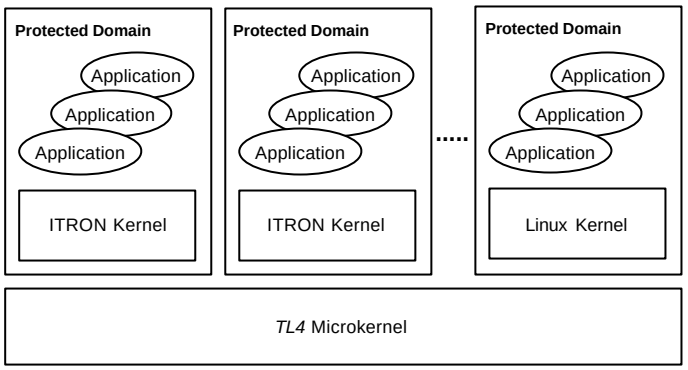


- コンポーネントの一部を異なるOS上で実行することが出来る。
  - リアルタイム実行のため
  - コンポーネントの障害から守るため
  - GPLの問題を避けるため

本提案の有効性 (2)



全体概要



- TL4マイクロカーネル
- 複数のμITRON kernel
- Linuxカーネル

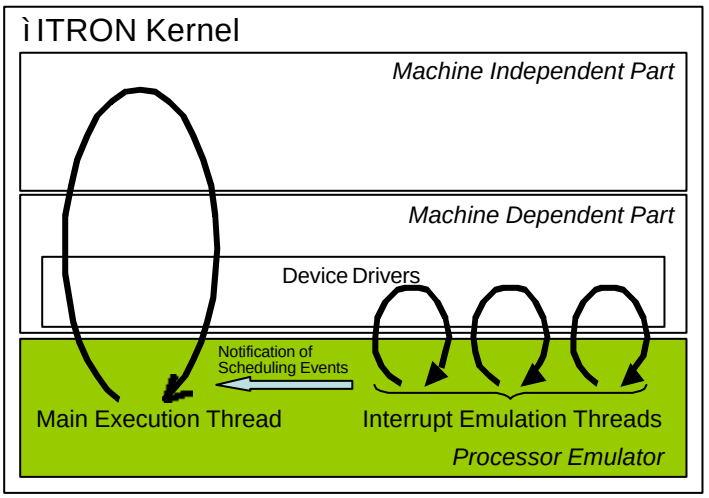
TL4マイクロカーネル

- ドイツカールスルーエ大学が開発したL4 μ-kernelをベース
- 以下の機能を提供する
  - スレッド
  - プロテクションドメイン
  - メモリページ
  - IPC
- オリジナルからの拡張
  - スケジューリング
  - リソース管理（将来課題)
  - セキュリティ（将来課題)

μITRONのスケジューリング

- TL4は複数のμITRON をスケジューリングする
  - TL4ははじめにどのμITRON カーネルを実行するかを決定する。
  - その後、選ばれたカーネルが管理する最高優先度のスレッドを実行する
- 2つのスケジューリングポリシをサポート
  - サイクリックエグゼクティブ
  - ディファラブルサーバ

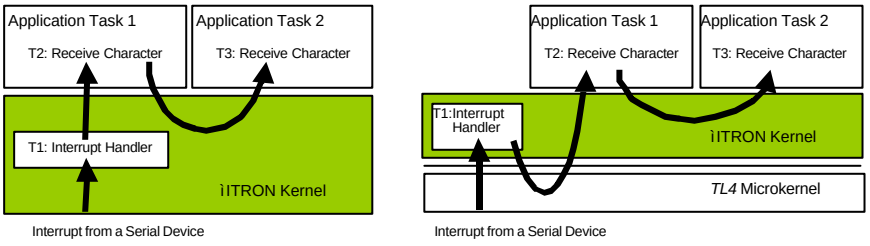
μITRON Kernel on TL4 (1)



システムの現状

- システムの実装
  - TL4 は L4Ka::Hazelnutをベースに実装
  - μITRON としては TOPPERS/JSP を利用
- 現状
  - 複数のμITRON を実行可能
  - L4/Linuxをそのまま利用することが可能
  - 現状では、LinuxとITRONはどちらかしかブートできないので両方を同時にブートできるようにする必要がある
  - L4/Linuxのプロセスの実装法の再検討
- 現状の評価
  - IBM ThinkPad X23 ラップトップPC
    - Mobile Pentium III 866 MHz

評価方法



(a) Original ìITRON Kernel

(b) ìITRON Kernel on *TL4* Microkernel

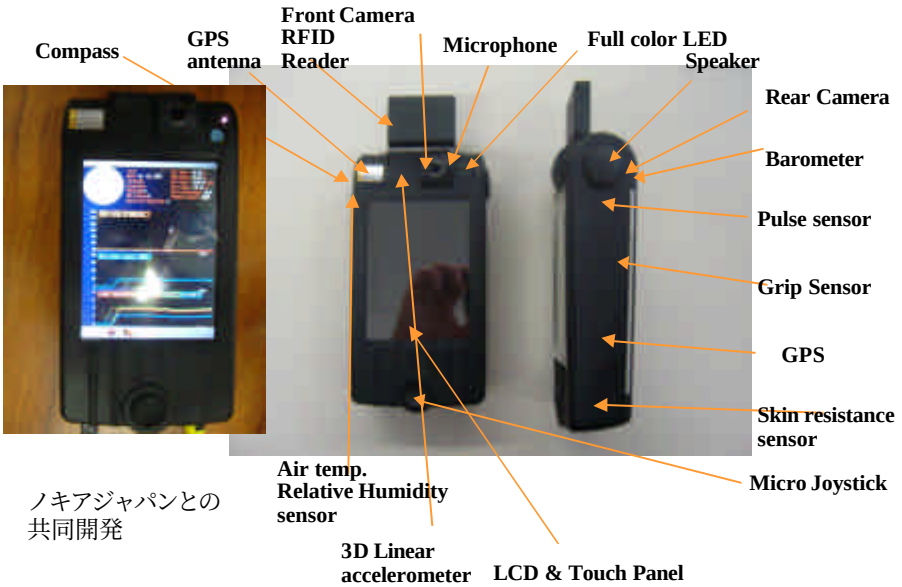
- 割り込みの応答性とコンテキストスイッチ時間を測定
  - 割り込みハンドラの時間
  - アプリケーション間のコンテキストスイッチ

評価結果

	Original ITRON	ìITRON on TL4
T1: µITRON Interrupt Handler	3.75 ìsec	2.64 ìsec
T2: Task 1 Received a Character	9.83 ìsec	9.48 ìsec
T3: Task 2 Received a Character	10.72 ìsec	10.56 ìsec

- µITRON on TL4 オリジナルよりも高速である
- 割り込みエミュレーションのオーバヘッドは重いハードウエア操作のコストに隠蔽されてしまう
- µITRON on TL4のコンテキストスイッチのオーバヘッドを今後検討が必要である

センサーのサポート(1)



ノキアジャパンとの共同開発

センサーのサポート(2)

- 各種センサーデバイスのアクセス
- センサーデバイスI/Oの標準API検討のためのプロトタイプ作成
- 様々なアプリケーションの構築からセンサーデータをどのように扱えるように出来るといいかを検討する
  - ポーリング
  - 定期的な割り込み
  - 複数イベントの条件待ち

センサーのサポート(3)

- 身の回りの人工物にコンピュータとセンサーを埋め込んで何が出来るのか？
  - ユーザの感情や振る舞いを抽出する

今後の課題

- QOSサポートとセキュリティ機構の統合
  - 現状はスーパユーザのみアカントオブジェクトの操作が可能
    - SELinux，Vserverとの統合を検討
  - 有効性を示すアプリケーションの実装
- μカーネルベースOS
  - 複数のμITRONとLinuxを同時に起動
    - デバイス共有フレームワーク
  - カーネル間のセキュリティの強化
  - 障害検出と高速カーネルリブート機構
  - コンポーネントベースの高信頼、セキュアで柔軟なOS
- センサーの扱い
  - センサーイベントをうまく扱うAPI
  - センサーデータからの情報抽出

結論

- 組込みLinuxの現状に関して簡単に解説し
- 組込みLinuxの今後のアプローチに関していくつか紹介した
  - QOSサポート拡張
  - μカーネルベースのOS
  - センサーの利用