独自思想OS「Shizuku」

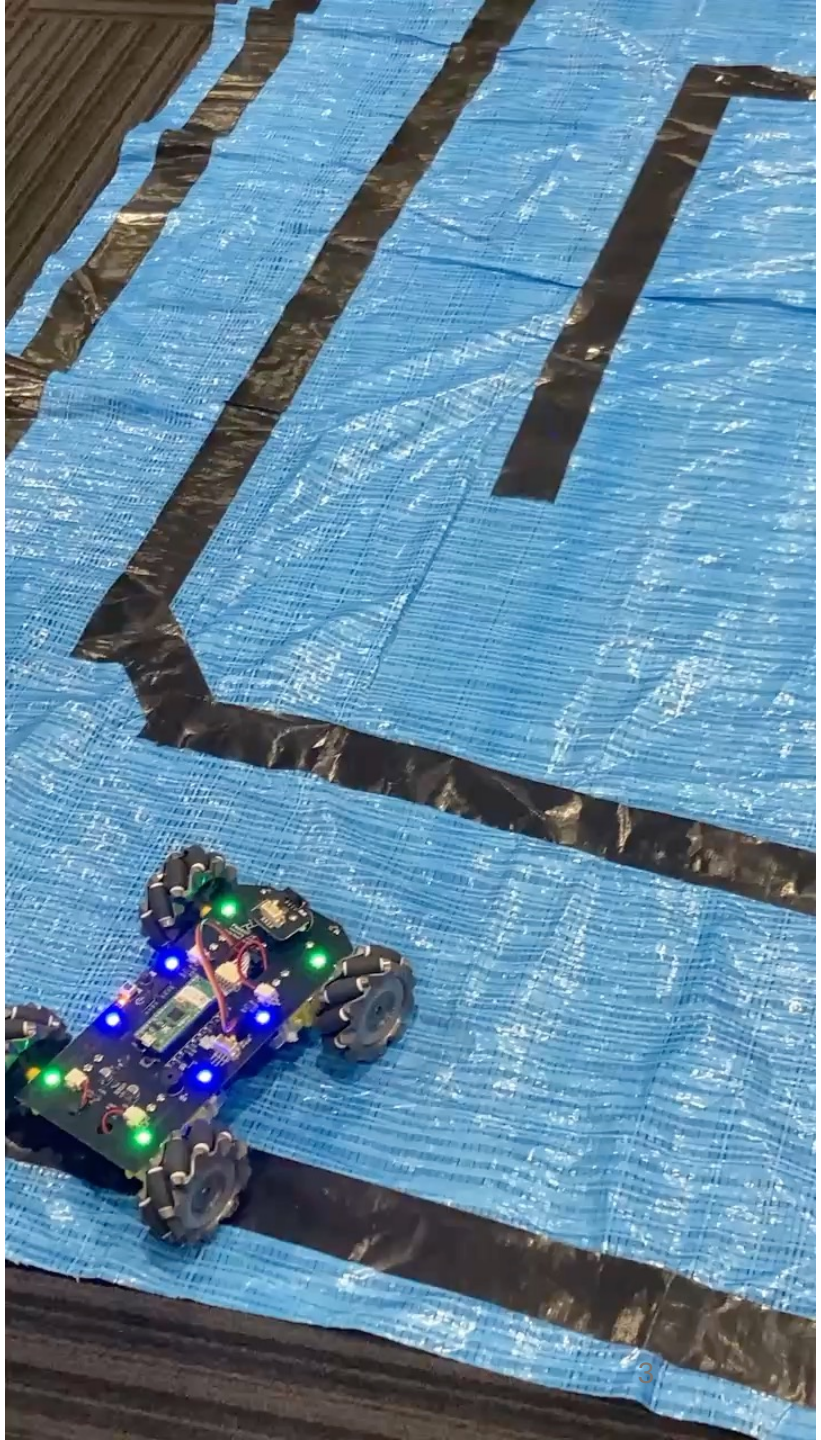
川合ゼミ 石垣有逢

自己紹介

- 氏名：石垣有逢
- ハンドルネーム：ゆあ
- 所属：東京理科大学創域理工学部電気電子情報工学科B1
- イベント：SecHack365’23 ,セキュキャンコネクト’26
- 好きなこと：電子工作大好き
- 嫌いなこと：エンジン整備・エンジン配線
- 趣味：プライベートクラウド運営
- サークル：学生フォーミュラ・飛行ロボコン・鳥人間
- トラウマ：メールサーバー攻撃受けたこと
- 最近嬉しかったこと：準中型運転免許取得したこと

はじめに：
作ったデモラジコンカーの紹介

- 某Bluetoothコントローラーで動く！
- 真横や斜めにも移動できる！
- うまく走らせればドリフトも…



なぜ、独自思想OS？

- 端末に応じて、理想的な「思想」が存在するから
  - セキュリティの思想
  - 操作の思想
  - リソース分配方法の思想
  - モジュール間通信の思想
- 現状の各OSは思想に選択肢がない(OSを使い分けないといけない)→変換機構が必要！

例えば、ラジコンカーを作るだけでも…

Linuxだと…

- デバイスを操作するためだけでもファイルシステムが必要→重い
- 特定のソケットに対してメッセージパッシングが必要→遅い
- ↑による速度低下を緩和するために規格が乱立→互換性崩壊

FreeRTOSだと…

- タスク固有のメモリという概念がない→グローバル変数の濫用
- 考えられる全ての機器を考えてドライバ制作→再利用困難
- プロジェクト固有規格でシステム構築→再開発の連鎖

どうしてこうなった？

僕の答え：思想(理想)と実装(現実)のズレ

- 互換性やわかりやすさのために考えやすい思想を求める…
- しかしそれはコンピューターにも考えさせている
- 結果的にコンピューターと人間の妥協点へ局所性を失っていく
- 結局機能や性能を持て余したり無駄使いするソフトウェアへ…

↓

環境負荷低減やエコが求められる今の時代、このままでいいのか？

Shizukuの思想→異なる思想の同居

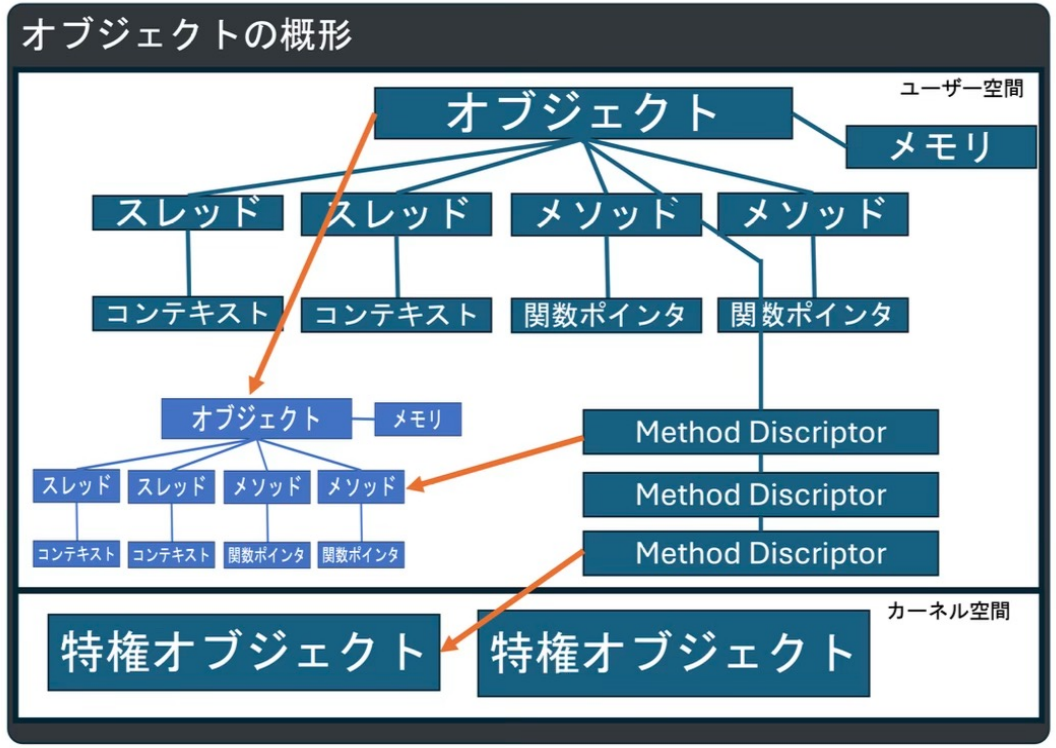
- Shizukuネイティブなソフトウェア(モジュール)は単純な思想で作ることを前提として、余計な機能を削ぎ落とす(セキュリティ機構など)→代わりにカーネルが後付けセキュリティ機構をつけるコネクタ(フック)を提供
- POSIXなど複雑な思想で作られたソフトウェアのために、システムコールを非カーネルソフトウェアでハンドリングできるようにする(これがセキュリティも提供する)
- 複雑な命令を定義して単純命令に変換後実行する考え方をOSにも取り入れる。
- もちろんパフォーマンスも妥協しない

Shizukuの主要メカニズム

- オブジェクトシステム
- メソッド呼び出し(動的スレッド生成)IPC
- ↑の高速化技術としてのコンテキストリユース
- マルチABI

オブジェクトシステム

- オブジェクト(ソフトウェアの主体)を前提とする抽象化機構
- オブジェクトはファイルらしくあり、プロセスらしくもある
- ファイルシステムやプロセスの代替
- オブジェクトは資源(実行者としてのスレッド・メモリ・他のオブジェクトに対する呼び出し権など)を持つ

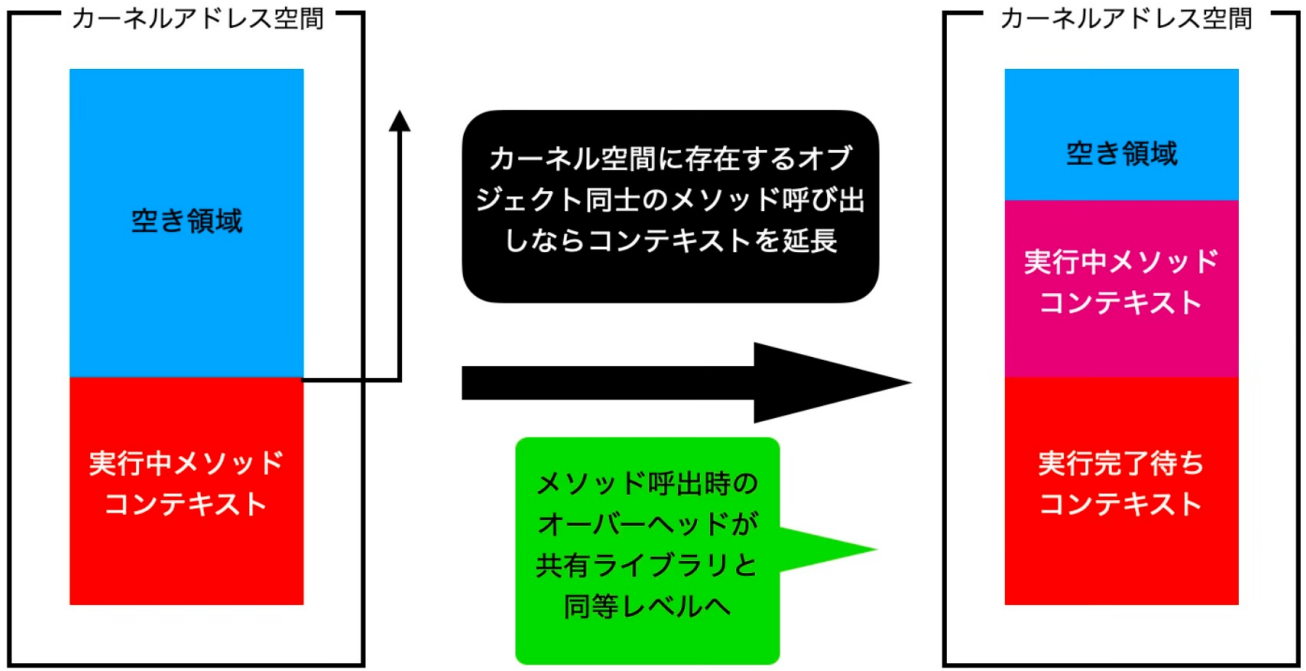


メソッド呼び出しIPC

- メッセージパッシングは毎度メッセージの解釈とディスパッチが必要
- 静的に作られたスレッドでは処理の依存関係を考慮するスケジューリングと相性が悪い(呼び出し元の実行時間を渡しにくい)
- 呼び出される側はメソッド(エントリーポイント)を公開してカーネルがそれを元にスレッドを動的に生成することで呼び出し元のスレッドに与えられた時間をそのまま呼び出し先スレッドの実行時間へ転送することを可能に！

コンテキストリユース

- 組み込み向けの機能
- アドレス空間を共有するスタックならメソッド呼び出ししたあとの空き領域を再利用できる→呼び出しコスト削減！



マルチABI

- アプリはシステムコールを発行することでOSとやりとりする
- つまり、OSがオブジェクトだとしたらそれに対するメソッド呼び出しなのでは？
- オブジェクトによるシステムコールハンドリングを行うことでオブジェクトごと異なるABIのシステムコールを同居させることに成功&デモカーで動作！
- 副次効果として割り込みを伴う組み込みprintfでシステムコールをデバッグできて嬉しい！！！

最後に...

- デモカーを遊びにきてくださいー
- キビキビ動いてくれます！
- オブジェクトによるシステムコールハンドリングは大体600cycle(Raspberry Pi pico 2の定格で2us)くらい...