TOPPERS/EV3RT
RTOSでMindstorms EV3開発

名古屋大学大学院情報科学研究科
高田本田研究室博士後期課程２年
李　奕驍

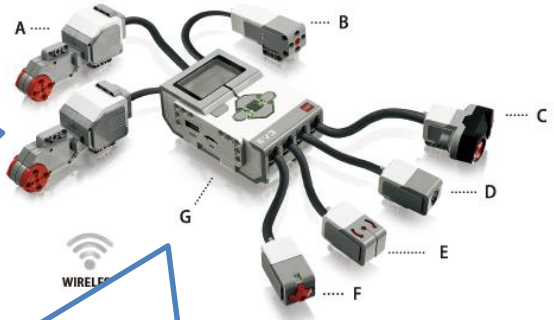
背景：Mindstorms EV3とは

- 多くの教育や研究で活用されているロボット開発キットシリーズ「Mindstorms」の最新版

Motor

Large Servo Motor

Medium Servo Motor



Sensor

Ultrasonic

Gyroscope

Color

Touch

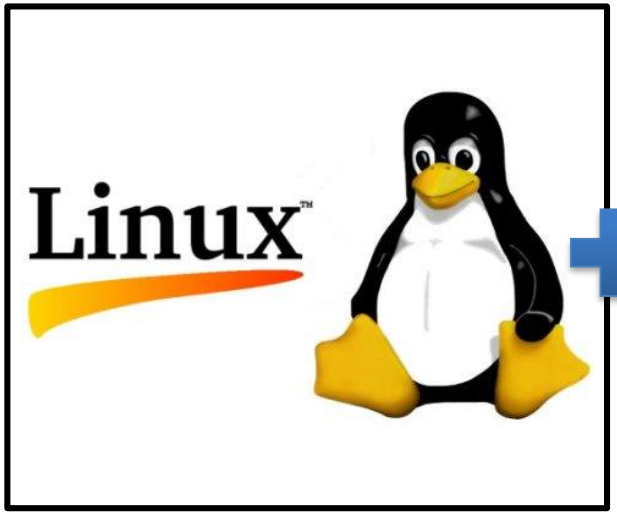
etc.

EV3 Intelligent Brick

Button, LED, Speaker, LCD, SD Card, USB, Wi-Fi, Bluetooth, etc.

背景：Mindstorms EV3の標準開発環境の課題

Operating System



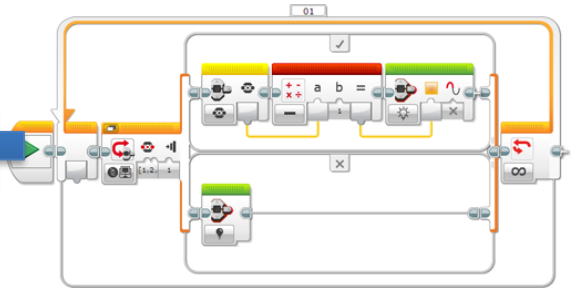
- 起動が遅い
- メモリ消費量が多い
- リアルタイム性不足
- デバドラのOverhead

Runtime

lms2012仮想マシン

- Task優先度つけない
- Task同期機能がない
- VMのOverhead

Programming language

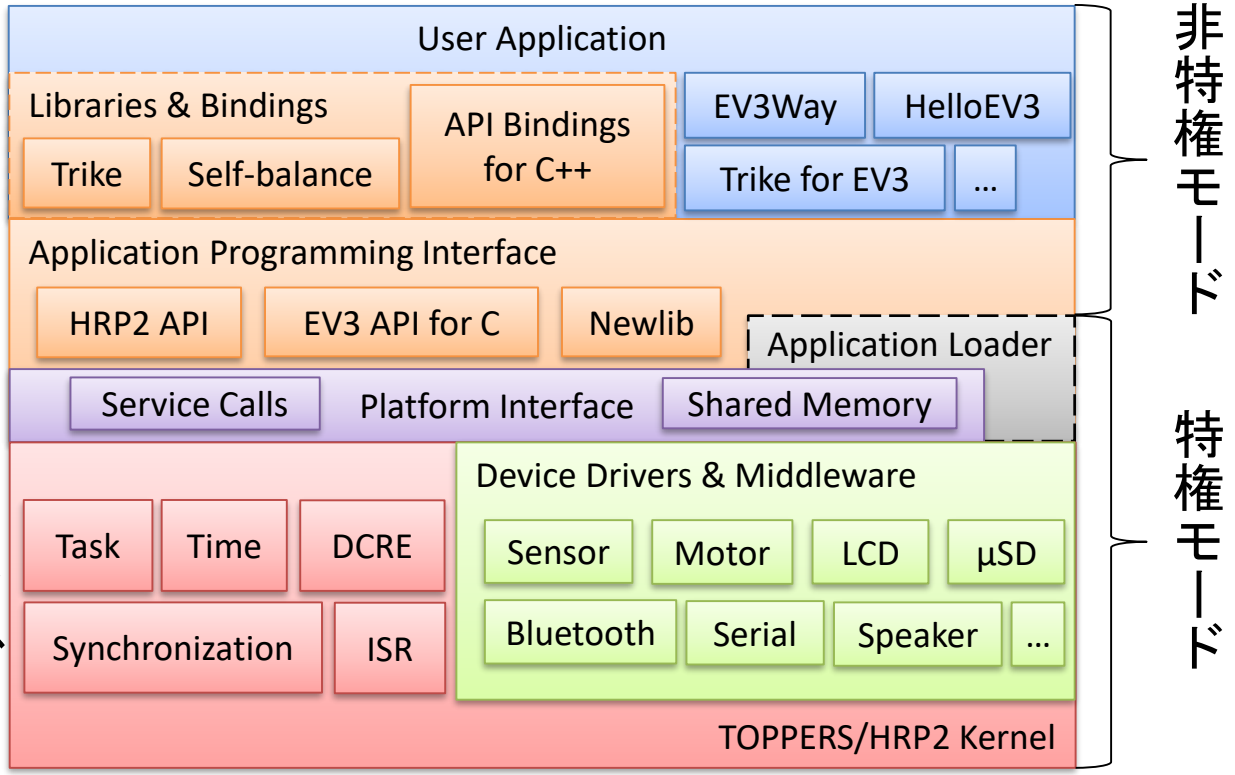


LabVIEW

- 複雑アプリ作り難い
- オープンソースではないため、拡張性があまりない

TOPPERS/EV3RTの特徴・アーキテクチャ

- HRP2カーネルをベースとしたプラットフォーム
  - TOPPERS第二世代カーネル仕様（μITRON4.0）リアルタイムOS
  - 保護機能対応でユーザアプリケーションの障害・バグはプラットフォームに波及しない、不具合が検出しやすくなる
- デバイスドライバ
  - 低オーバヘッド
- C/C++で開発
  - 豊富なAPI
- 高速起動
- メモリ消費少ない
- 動的ローディング



ETロボコンの公式プラットフォーム（2015以降）

なお、ETロボコンの技術教育は**TOPPERS/EV3RT(C++言語）でのみ**実施します。どのプラットフォームで参加するかお悩みの場合、特にプライマリークラスへ参加される場合には、**まずはTOPPERS/EV3RTをお試しください**。

プラットフォーム	**OS**	言語	開発環境構築
TOPPERS/EV3RT	TOPPERSリアルタイム	C, C++	Windows/Mac/Linux この先*TOPPERS*サイトとなります
mruby on EV3RT + TECS	TOPPERSリアルタイム	mruby	Windows7/8 この先*TOPPERS*サイトとなります
MonoBrick	Linux	C#	Windows7/8/10
leJOS EV3	Linux	Java	Windows7/8/10　Mac
※ ev3dev	Linux	Python 他	Windows/Mac/Linux この先*ev3dev*サイトとなります

Ref: https://github.com/ETrobocon/etroboEV3/wiki

EV3RTのインストール

- Intelligent Brick（EV3本体）にインストール
  - リリースパッケージ（ev3rt-beta6-3-release.zip）の「sdcard」フォルダ内のファイルをEV3のmicroSDカードにコピーするだけ

- 開発環境構築（Ubuntu 14.04・Bash on Windowsの例）

```
$ sudo add-apt-repository ppa:team-gcc-arm-embedded/ppa -y
$ sudo apt-get update
$ sudo apt-get install gcc-arm-embedded u-boot-tools libboost1.55-all-dev -y
$ tar xvf hrp2.tar.xz && cd hrp2/cfg
$ make
```

  - Mac OS X、Cygwin等もサポート
    - 詳細は「http://dev.toppers.jp/trac_user/ev3pf/wiki/WhatsEV3RT」

アプリケーションプロジェクト

- EV3RTのアプリは「hrp2/sdk/workspace」の下で管理
  - デフォルトはいくつかのサンプルアプリが入っている

- アプリのプロジェクトフォルダに以下のファイルが存在
  - app.c or app.cpp : デフォルトのソースファイル
  - app.cfg : アプリ用cfgファイル（タスクの生成等）
  - Makefile.inc : アプリ用Makefile
- アプリケーションのビルド
  - 動的ローディング用モジュール
    - make app=<フォルダ名>
  - スタンドアローン形式イメージ
    - make img=<フォルダ名>

```
1  APPL_COBJS += balancer.o balancer_param.o
2  APPL_CXXOBJS +=
3  SRCLANG := c++
4  ifdef CONFIG_EV3RT_APPLICATION
5  # Include libraries
6  include $(EV3RT_SDK_LIB_DIR)/libcpp-ev3/Makefile
7  endif
```

Makefile.incの例

アプリケーションの2つの実行形式

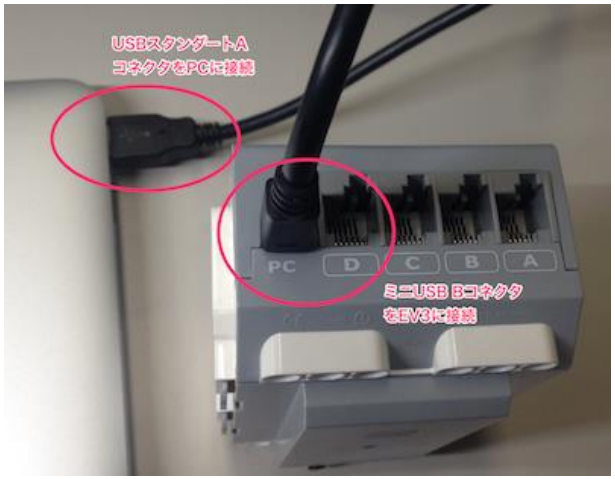
- 動的ローディング形式
  - 動的ローディング用モジュールとしてビルド
  - EV3RTのアプリケーションローダーでロードして実行する
  - アプリ更新・変更の時、EV3を再起動する必要がない
    - 再起動、Bluetooth再接続等の手間を省く
  - アプリは特定の保護ドメインしか使えない
    - HRP2の保護機能を意識しなくても良い（ASPカーネルのように開発）

```
DOMAIN(TDOM_APP) {
CRE_TSK(MAIN_TASK, { TA_ACT , 0, main_task, TMIN_APP_TPRI + 1, STACK_SIZE, NULL });
CRE_TSK(BT_TASK  , { TA_NULL, 0, bt_task  , TMIN_APP_TPRI + 2, STACK_SIZE, NULL });
}
```

- スタンドアローン形式
  - EV3のブートローダ（U-Boot）で直接に起動できるイメージ
  - 全ての機能を使える、プラットフォームの開発・拡張向け
    - 動的ローディングのローダーは一つのスタンドアローン形式のアプリ

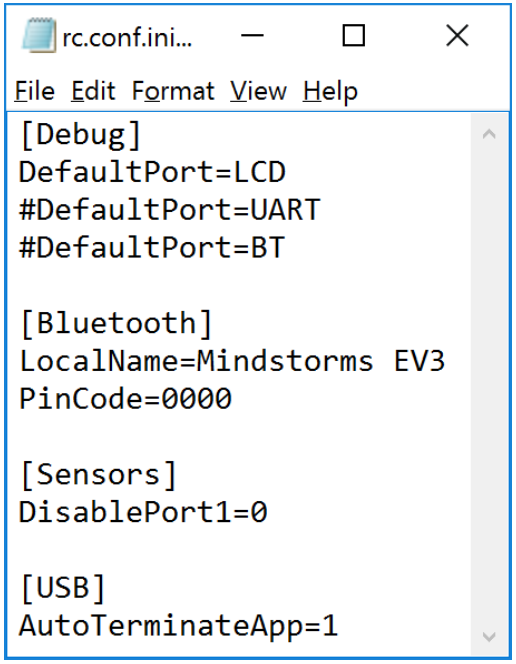
アプリケーションローダー

- EV3RT起動後EV3RT Consoleの画面が表示される
  - EV3本体でタスクログを確認できる
  - 「Load App」でアプリをロードできる
    - SDカードからファイルを選択
      - ✓「/ev3rt/apps/」フォルダ内
    - Bluetooth SPP（仮想シリアル）で転送
      - ✓Tera Term -> Transfer -> ZModem
  - USBでPCに接続してSDカードを管理
    - アプリをSDカードにコピーして実行
  - アプリ実行中をConsoleを呼び出す
    - BACKボタン長押し
    - アプリを終了（アンロード）できる



プラットフォームの設定変更

- 設定ファイルでいくつかの設定が可能
  - 設定ファイル位置：SDカードの「/ev3rt/etc/rc.conf.ini」
  - タスクログの出力先
    - デフォルトはLCD（EV3RT Console）
  - Bluetoothのデバイス名とPINコード
  - センサポート1の有効化
    - 無効化するとシリアルポートとして使える
  - USB接続で自動的にアプリを終了するか
    - 排他制御のため、アプリ実行中にUSBでSDカードをアクセスできない



Application Programming Interface

- EV3RTのアプリはデフォルトで以下３つのAPIが使える
  - TOPPERS/HRP2カーネルAPI
    - RTOS機能を提供
    - サービスコール、cfgファイルに使う静的API
  - 標準Cライブラリ
    - fopen()等でmicroSDカードを操作可能、パス「/」はSDカードのルート
    - 動的メモリ確保も対応、TLSF Memory Allocatorを採用
  - EV3用C言語API
    - モータ、センサ、スピーカ、LCD等EV3の機能を提供
    - http://www.toppers.jp/ev3pf/EV3RT_C_API_Reference
- 上記のAPIの他に、静的ライブラリもインポート可能
  - 例：EV3用C++言語API（libcpp-ev3）
    - http://www.toppers.jp/ev3pf/EV3RT_CXX_API_Reference

Application Programming Interface

- Bluetooth通信機能
  - EV3RTとPCはBluetoothのSerial Port Profile（SPP）で通信可能
    - オープンソースのプロトコルスタックBTstackを採用
  - BTstackを操作するハードルが高いため、簡単な方式を提供
    - TOPPERS/HRP2カーネルのシリアルインターフェスを使う

```
// Bluetooth仮想シリアルポートからbufが指すバッファへ最大lenバイトを読み込む
serial_rea_dat(SIO_PORT_BT, buf, len);

// bufが指すバッファからBluetooth仮想シリアルポートへ最大lenバイトを書き込む
serial_wri_dat(SIO_PORT_BT, buf, len);
```

    - 標準Cライブラリのファイル操作関数を使う

```
// Bluetooth仮想シリアルポートのファイルをオープンする
FILE *bt = ev3_serial_open_file(EV3_SERIAL_BT);

// 書式化した文字列をBluetooth仮想シリアルポートへ書き込む
fprintf(bt, "Bluetooth SPP ID: %d\n", EV3_SERIAL_BT);

// Bluetooth仮想シリアルポートから1文字を読み取る
int c = fgetc(bt);
```

性能評価

- プラットフォームの基本特徴

Table 3 Basic characteristics of EV3RT, leJOS and MonoBrick

	EV3RT	leJOS	MonoBrick
Boot time	2.0s	65.5s	105.4s
Memory footprint	2,230 KiB (3.4%)	58,888 KiB (89.9%)	43,224 KiB (66.0%)
CPU utilization	1%	6%	5%

- EV3RTの起動は他のプラットフォームと比べて遥かに速い
  - leJOSの30倍、MonoBrickの50倍以上
- EV3RTのメモリ消費量が少ない（EV3全体RAM容量の3.4%）
  - より大規模なデータを処理するアプリを開発可能
  - ファイル・データのインメモリ化によりリアルタイム性向上
- EV3RTのCPU負荷率は一番低い
  - プラットフォームの機能がユーザアプリの性能に影響しにくい

性能評価

- タスク切り替えの遅延

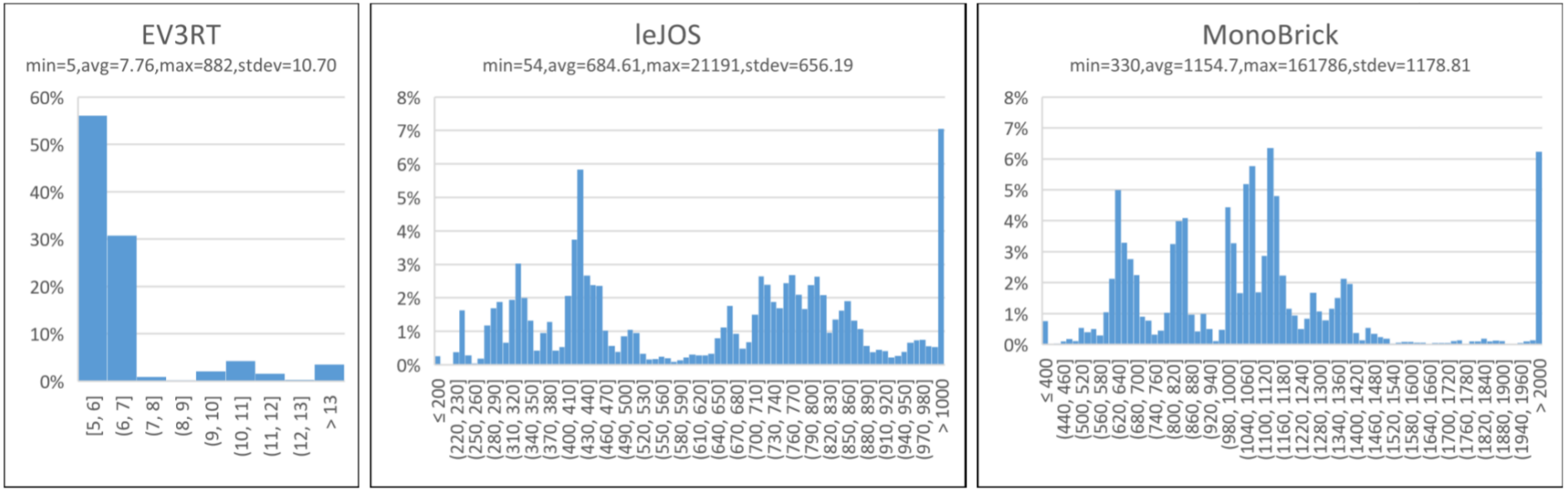


Fig. 14 Histograms of observed thread switching latency in μs

- EV3RTの平均性能は他のプラットフォームの約百倍
- EV3RTの性能ばらつきは一番小さい
  - 約85%は7us以下、約96%は12us以下

性能評価

- モータ制御APIのオーバヘッド

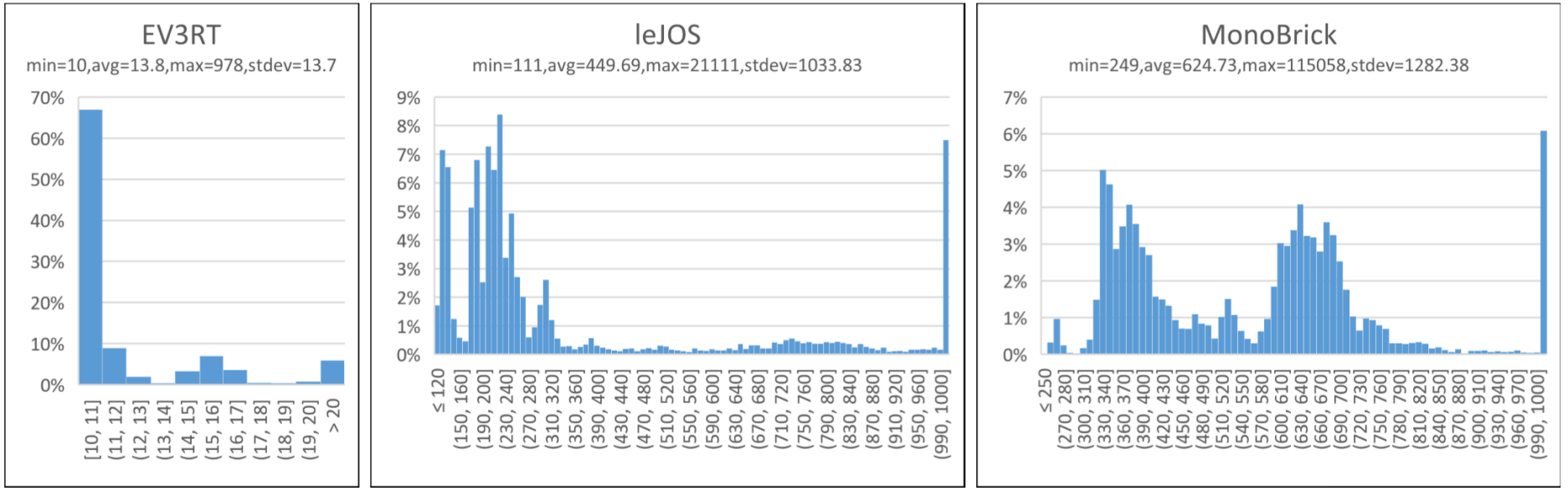


Fig. 15 Histograms of motor control API's overhead in μs

- EV3RTの平均性能は他のプラットフォームの36倍以上
- EV3RTの性能ばらつきは一番小さい
  - 約76%は12us以下、約94%は20us以下

性能評価

- センサ取得APIのオーバヘッド

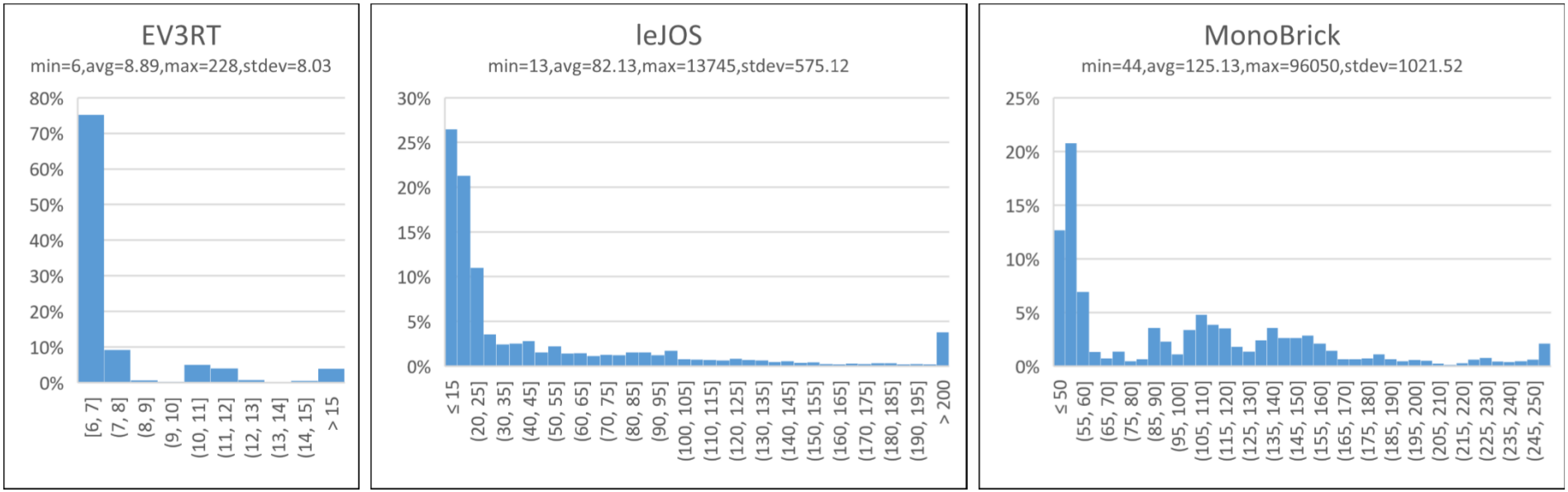


Fig. 16 Histograms of sensor access API's overhead in μs

- EV3RTの平均性能は他のプラットフォームの約10倍
- EV3RTの性能ばらつきは一番小さい
  - 約85%は8us以下、約96%は12us以下

今後の展開・Work in Progress

- リアルタイム制御に十分な性能を達成したため、今はユーザビリティの向上を目指している
  - 通信機能の強化
    - TCP/IP
    - Bluetooth PAN（Personal Area Network）
    - USB Wi-Fi Dongle（ESPr Developer等）
  - Eclipse CDTのサポート
    - ソースファイル管理（Makefile.incの自動生成）
    - Run（or Upload and Execute）Application
    - Remote Debug（GDB server over TCP/IP）
  - TOPPERS/HRP3カーネルの対応
    - ティックレスの高分解能時間管理
    - タスク終了要求機能
    - 時間パーティショニング機能

今後の展開・Work in Progress

- 通信機能の強化
  - 課題
    - Bluetooth SPP経由アプリを転送する時、EV3本体とPC両方操作する必要がある。更新頻度が高い場合、思った以上に手間がかかる
    - EV3間の通信機能がないため、複数台のEV3から構成される作品は開発しにくい（例：6軸ロボットアームV760）
    - インターネットに接続できないため、Open Roberta Lab等Webベースの開発環境はサポートできない
  - ネットワーク通信機能で解決したいが、ミドルウェア不足
    - EV3RTのBluetoothプロトコルスタックBTstack
      - ✓パケット通信まではサポート、TCP/IPやDHCPサーバ等がない
    - ESPr Developer等USBシリアルで通信するWiFiモジュール
      - ✓HRP2カーネル用USBホストスタックがない

今後の展開・Work in Progress

- 通信機能の強化
  - mbed-on-toppers（仮）
    - TOPPERSカーネル用ARM mbed OS 5の互換レイヤ
      - ✓mbed communityの沢山のミドルウェアを簡単に使える
      - ✓ライセンスは比較的に緩い（Apache v2/BSD/MIT…）
    - EV3のような元々mbedが対応しないターゲットでも動作
    - EV3RTはEthernetInterface、lwIP、DhcpServer、USB Host Stackを使用
  - Bluetooth PANでのTCP/IP通信は対応済み
    - 例：HTTPでアプリを転送「curl -H "Content-Type: ev3rt/app" --data-binary @app http://10.0.10.1/upload」
  - USB Host ControllerのHardware Abstraction Layerを実装中

- ご清聴ありがとうございました