OSC Nagoya 2024
2024/05/25

C言語やROSでLEGOを動かそう

樋山一樹
（南山大学 / TOPPERS）

目次

- 自己紹介
- SPIKE-RTの紹介
- SPIKEをROS 2で動かすための開発環境の紹介

自己紹介

樋山一樹（ひやま　いつき）

- 南山大学　理工学研究科 修士2年
- 学部3年時に研究室に配属されて以来，組込みシステムを中心に学習中
  - 研究室HP：https://honda-lab1.sakura.ne.jp
- 研究室配属後にTOPPESの活動に参加

目次

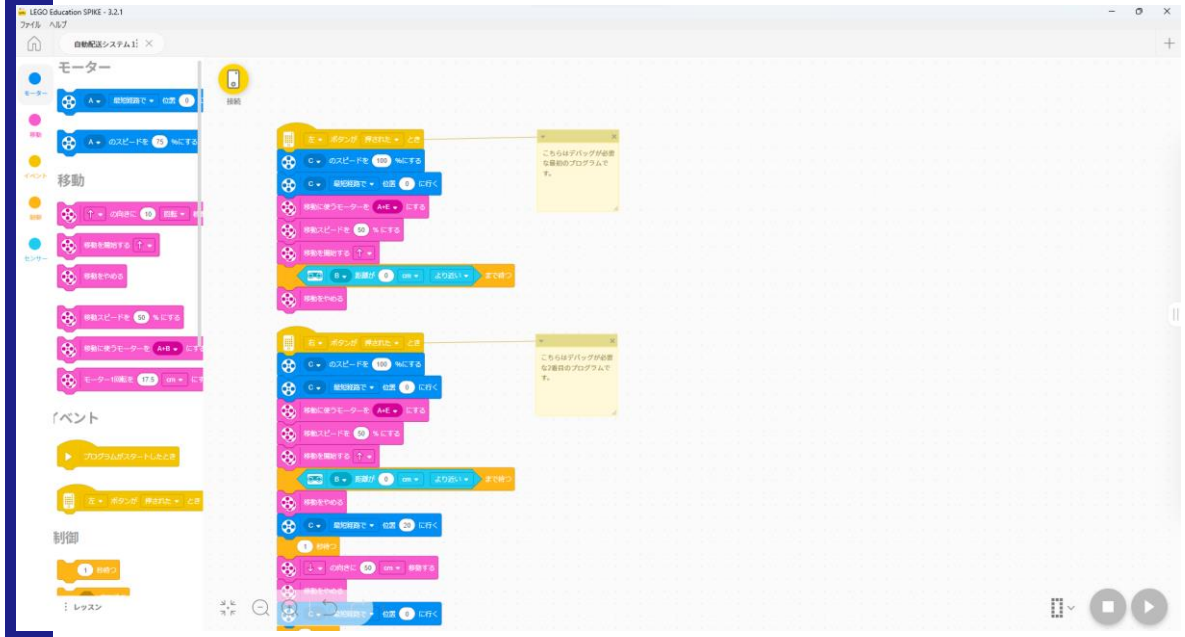
- 自己紹介
- SPIKE-RTの紹介
- SPIKEをROS 2で動かすための開発環境の紹介

LEGO Education SPIKE Primeとは

- SPIKE
  - LEGO社とMITが共同で開発
    - プログラミング教育キット
  - HubとPUPデバイスを組み合わせてロボットを制作
  - 公式ではScrachやPythonでのプログラミングをサポート
- Hub
  - STM32F413（Cortex-M4）

プログラミング可能なHub

- EV3よりスペックダウン
- Linuxの実行は難しい

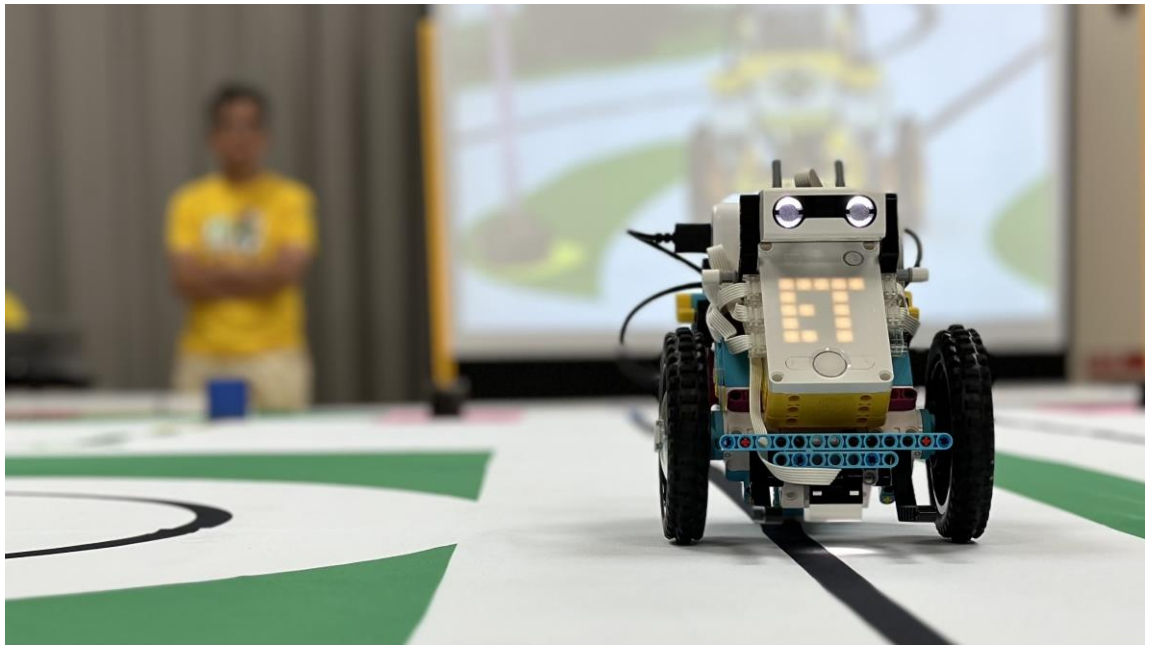


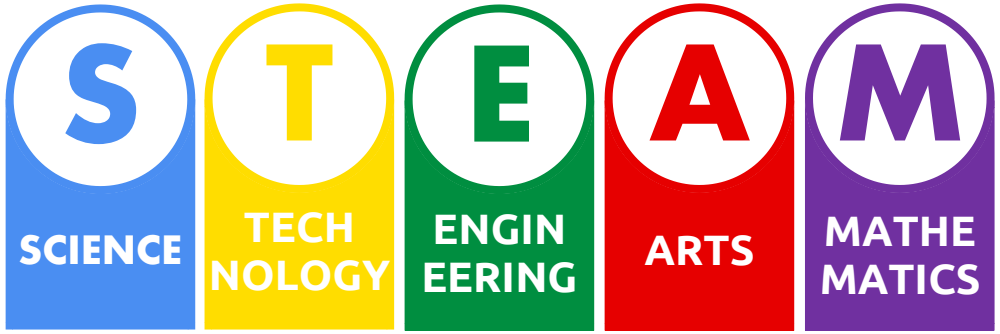
SPIKE Prime App

SPIKE Prime

SPIKEの活用

- 活用場面
  - 教育現場
  - ロボットコンテスト
    - ETロボコン
    - WRO（World Robot Olympiad）
    - など

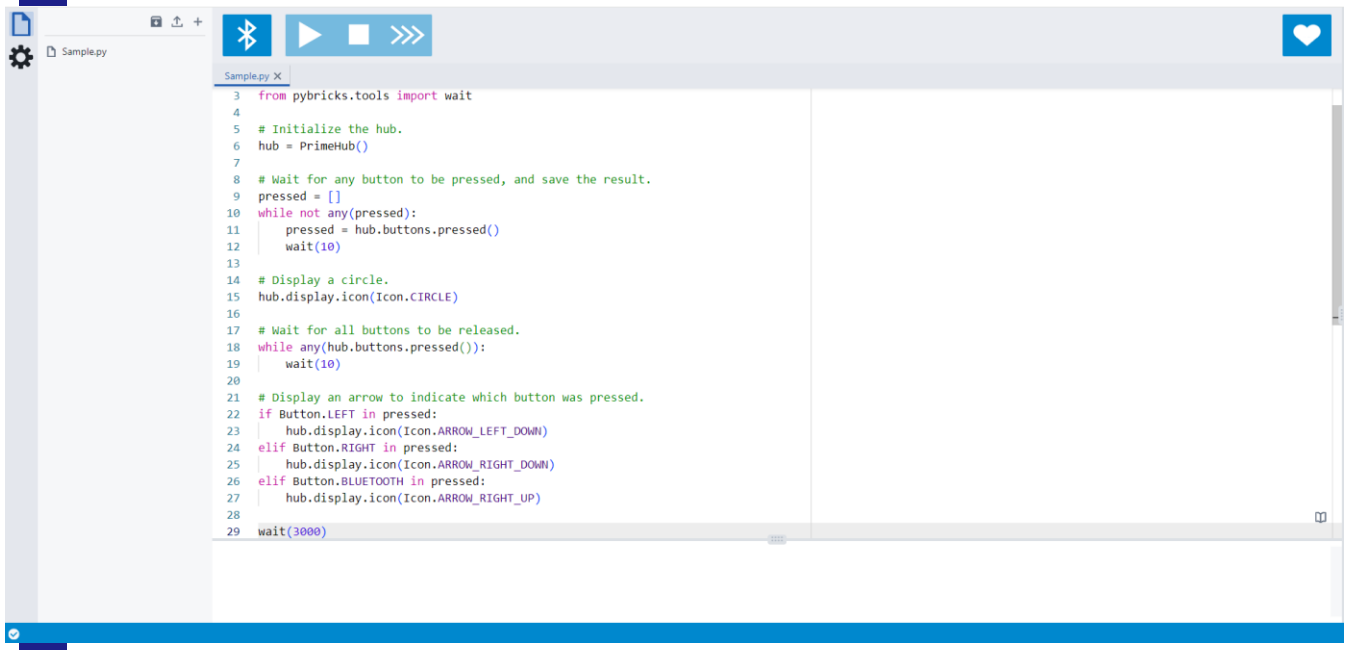




ETロボコン(https://www.etrobo.jp/)

OSSコミュニティとSPIKE

- 様々なOSSコミュニティによりSPIKE向けSWプラットホームが開発されている
  - Pybricks
    - ブラウザ上でPythonプログラミングが可能
  - SPIKE-RT
    - TOPPERSプロジェクト
    - この後紹介

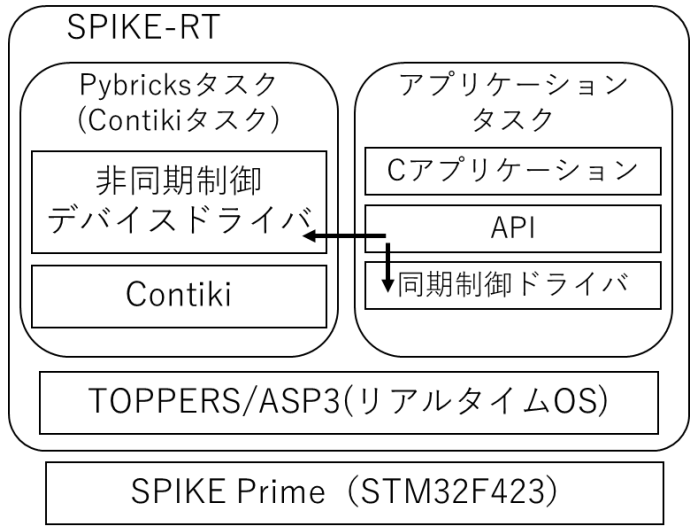


TOPPERS
Toyohashi OPen Platform
for Embedded Real-time Systems

SPIKE-RT

- SPIKE-RT
  - SPIKEで利用することを目的として開発されたRTOS
    - 軽量
      - メモリ使用量が搭載量の2割以下
    - C言語でのアプリケーション開発が可能
    - マルチタスクプログラミング
    - アプリケーションのリアルタイム性を確保
  - TOPPERS/ASP3カーネルがベース
    - ITRON系

SPIKE-RT



SPIKE-RTの内部構成

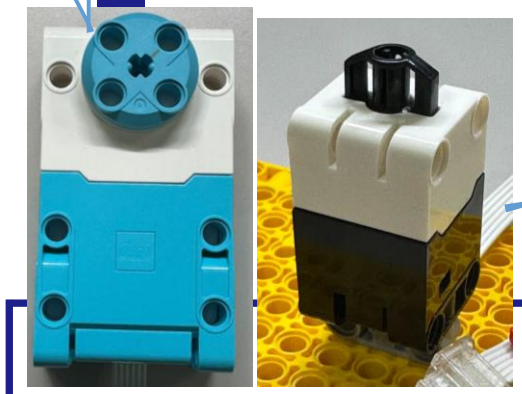
C言語

・処理速度が高速
・HW制御
・低レイヤの開発で多く活用
・自動車を始めとした多くの組込み機器で活用

SPIKE-RTプログラミング

- APIリファレンス：https://spike-rt.github.io/spike-rt/ja/html/modules.html

API REFERENCE



＜フォースセンサを押す力に応じてモータを駆動する＞

初期化処理

```
pbio_error_t err;
pup_motor_t *motor;
pup_device_t *force;

dly_tsk(3000000);

// Get pointer to device
motor = pup_motor_get_device(PBIO_PORT_ID_A);
force = pup_force_sensor_get_device(PBIO_PORT_ID_D);
```

デバイスへのポインタ

駆動処理

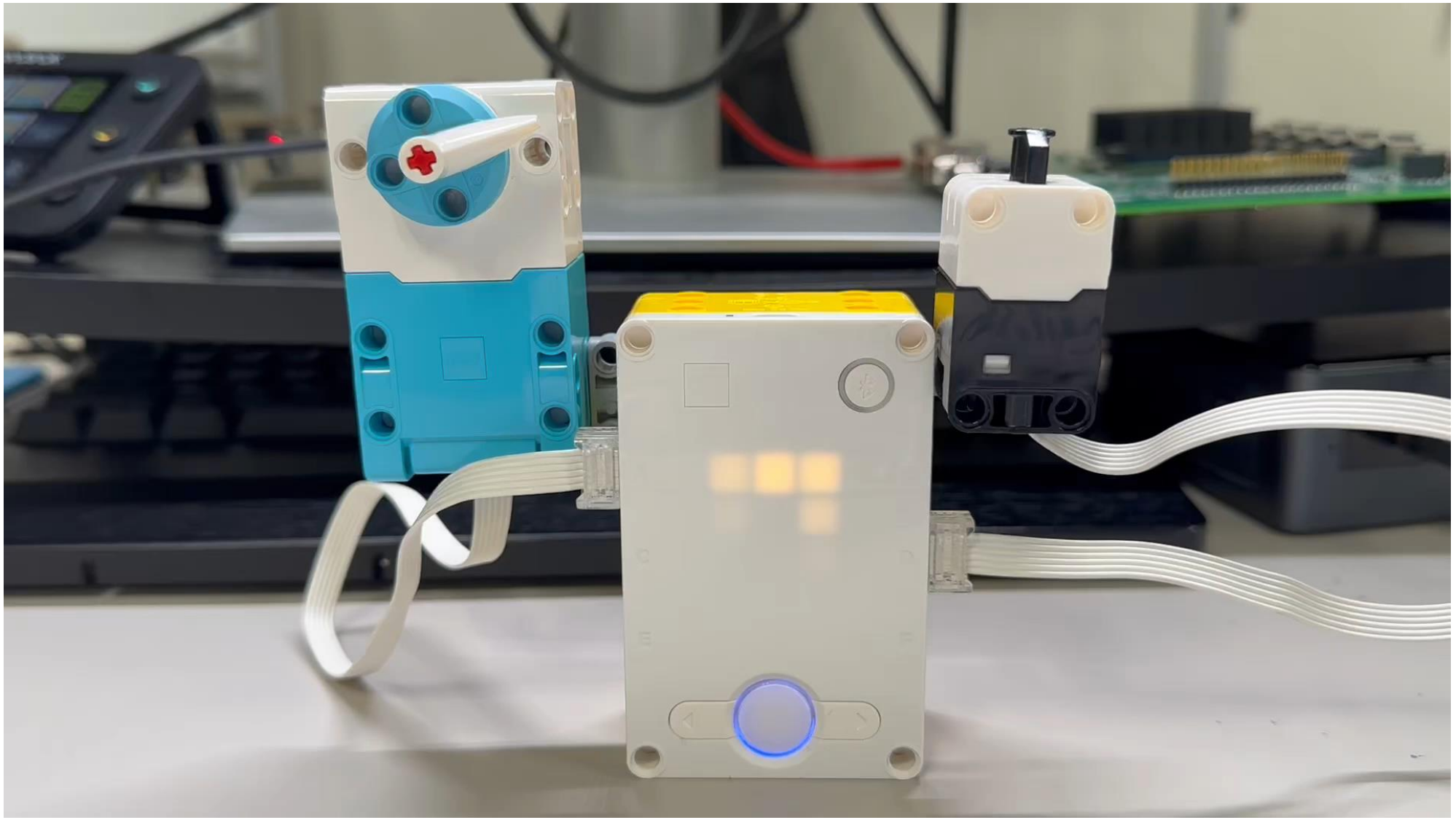
```
int force_val;

while (1)
{
  force_val = pup_force_sensor_force(force);
  pup_motor_set_speed(motor, force_val * 100);

  dly_tsk(10000);
}
```

力の大きさを取得

接続モータへのポインタ

指令値

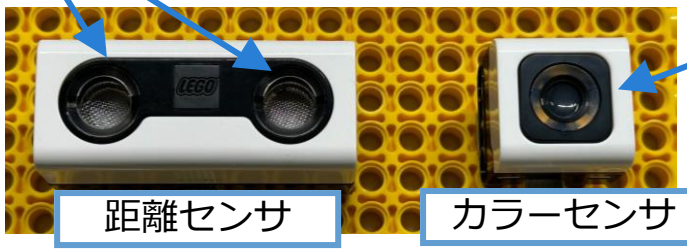
SPIKE-RTプログラミング



SPIKE-RTプログラミング

ライト

<カラーセンサで取得した周囲の明るさに応じて距離センサのライトを点灯する>



初期化処理

```
pup_device_t *col;
pup_device_t *ult;

dly_tsk(3000000);

// Get pointer to device
col = pup_color_sensor_get_device(PBIO_PORT_ID_A);
ult = pup_ultrasonic_sensor_get_device(PBIO_PORT_ID_B);
```

デバイスへの
ポインタ

ポインタ取得

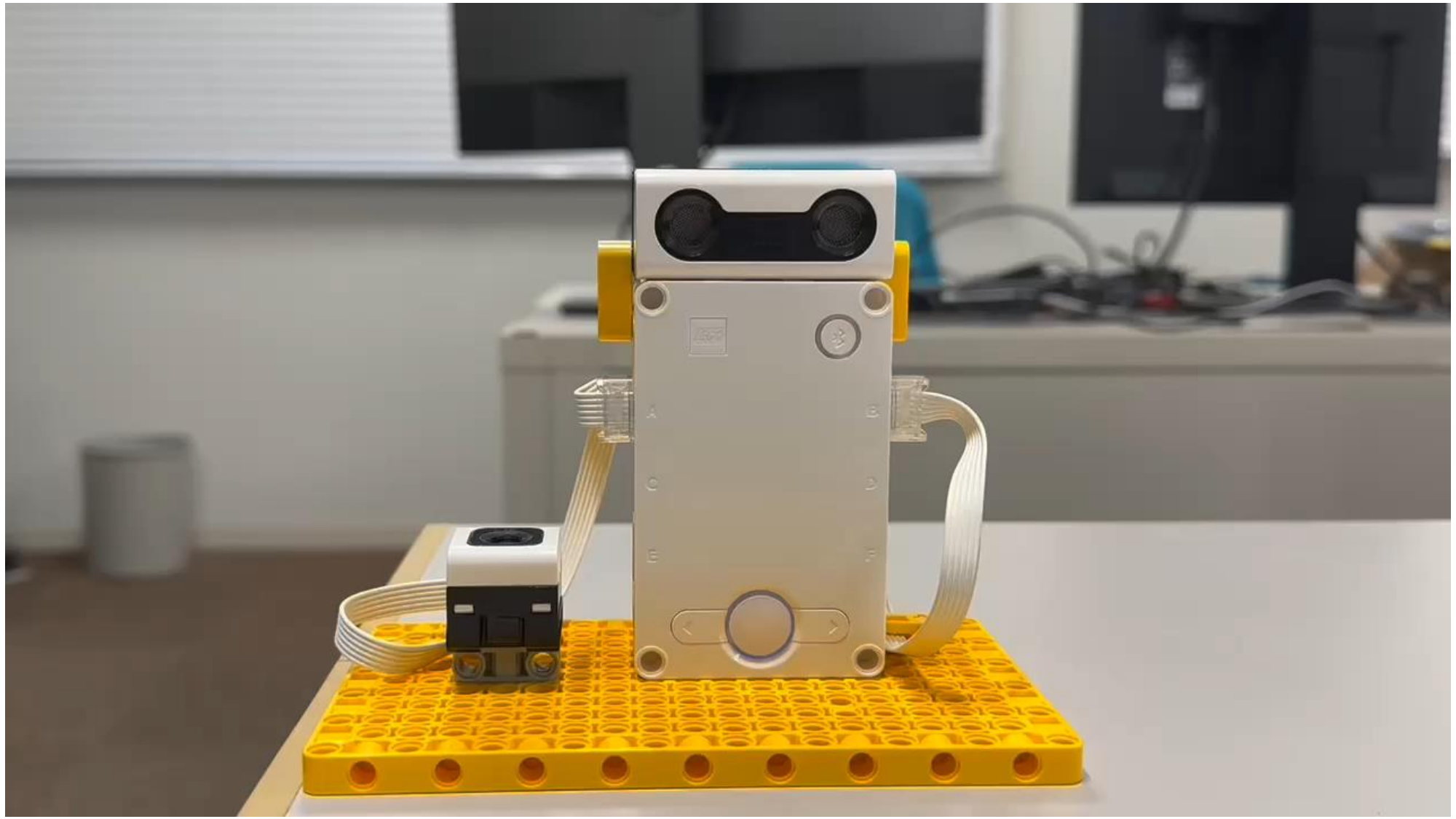
駆動処理

明るさを取得

```
while (1)
  {
    amb = pup_color_sensor_ambient(col);
    hub_display_off();

    if(amb > 10 && amb <= 40)
      pup_ultrasonic_sensor_light_set(ult, 0, 30, 0, 30);
    else if(amb <= 10)
      pup_ultrasonic_sensor_light_set(ult, 30, 30, 30, 30);
    else
      pup_ultrasonic_sensor_light_off(ult);

    dly_tsk(100000);
  }
```

距離センサ
ライト点灯

SPIKE-RTプログラミング



マルチタスクプログラミング

- 教材：NCES Education Program /「組込みソフトウェア開発技術の基礎」など
  - https://www.nces.i.nagoya-u.ac.jp/NEP/materials/about.html
- データキューを使用してタスク間通信を行う
  - 本体ボタンの押下状態に応じてデータをデータキューに送信(but_cyc_handler)
  - 受信側は受信データに応じてモータを駆動する(motor_task)
- 周期ハンドラを使用してボタンの押下状態を確認する

タスク生成

周期ハンドラ生成

データキュー

生成時にタスクを起動

コンフィギュレーションファイル(.cfg)

```
CRE_TSK(MAIN_TASK, { TA_ACT, 0, main_task, MAIN_PRIORITY, STACK_SIZE, NULL });
CRE_TSK(MOTOR_TASK, { TA_NULL , 0, motor_task, MOTOR_PRIORITY, STACK_SIZE, NULL });
CRE_TSK(BUT_CYC_HANDLER, { TA_NULL , 0, but_cyc_handler, BUTTON_CYC_PRIORITY, STACK_SIZE, NULL });

CRE_CYC(BUT_CYC, {TA_NULL, {TNFY_ACTTSK, BUT_CYC_HANDLER}, 1000, 0});

CRE_DTQ(MOTOR_DTQ,{TA_NULL, 10, NULL});       // NULL -> データキューの管理領域をカーネル等が確保
```

※TOPPERS代3世代カーネル統合仕様書：https://www.toppers.jp/documents.html

マルチタスクプログラミング (タスク間通信)

送信データ型（構造体）

```
struct data_packet{
  pup_motor_t *motor;
  int speed;
  int idx;
  bool is_setup;
};
```

main_task()

```
void
main_task(intptr_t exinf)
{
  act_tsk(MOTOR_TASK);
  sta_cyc(BUT_CYC);

  <省略>
    while (1)
  {
    slp_tsk();
  }
}
```

起動

タスク・周期ハンドラ起動

自身は休止状態に

データ送信

but_cyc_handler()

```
void
but_cyc_handler(intptr_t exinf)   //1ms周期
{
  struct data_packet send_pkt;
  static hub_button_t pressed_ptn, pre_ptn;
  intptr_t send_data;

  hub_button_is_pressed(&pressed_ptn);

  if ((pressed_ptn & HUB_BUTTON_LEFT)
       && pre_ptn == 0) {
    create_pkt(&send_pkt, MOTOR_A);
    send_data = &send_pkt;

    snd_dtq(MOTOR_DTQ, send_data);
  }
  else if ((pressed_ptn & HUB_BUTTON_RIGHT)
            && pre_ptn == 0){
    create_pkt(&send_pkt, MOTOR_B);
    send_data = &send_pkt;

    snd_dtq(MOTOR_DTQ, send_data);
  }
  pre_ptn = pressed_ptn;
}
```

データ(構造体)生成

データ(構造体)生成

Queue

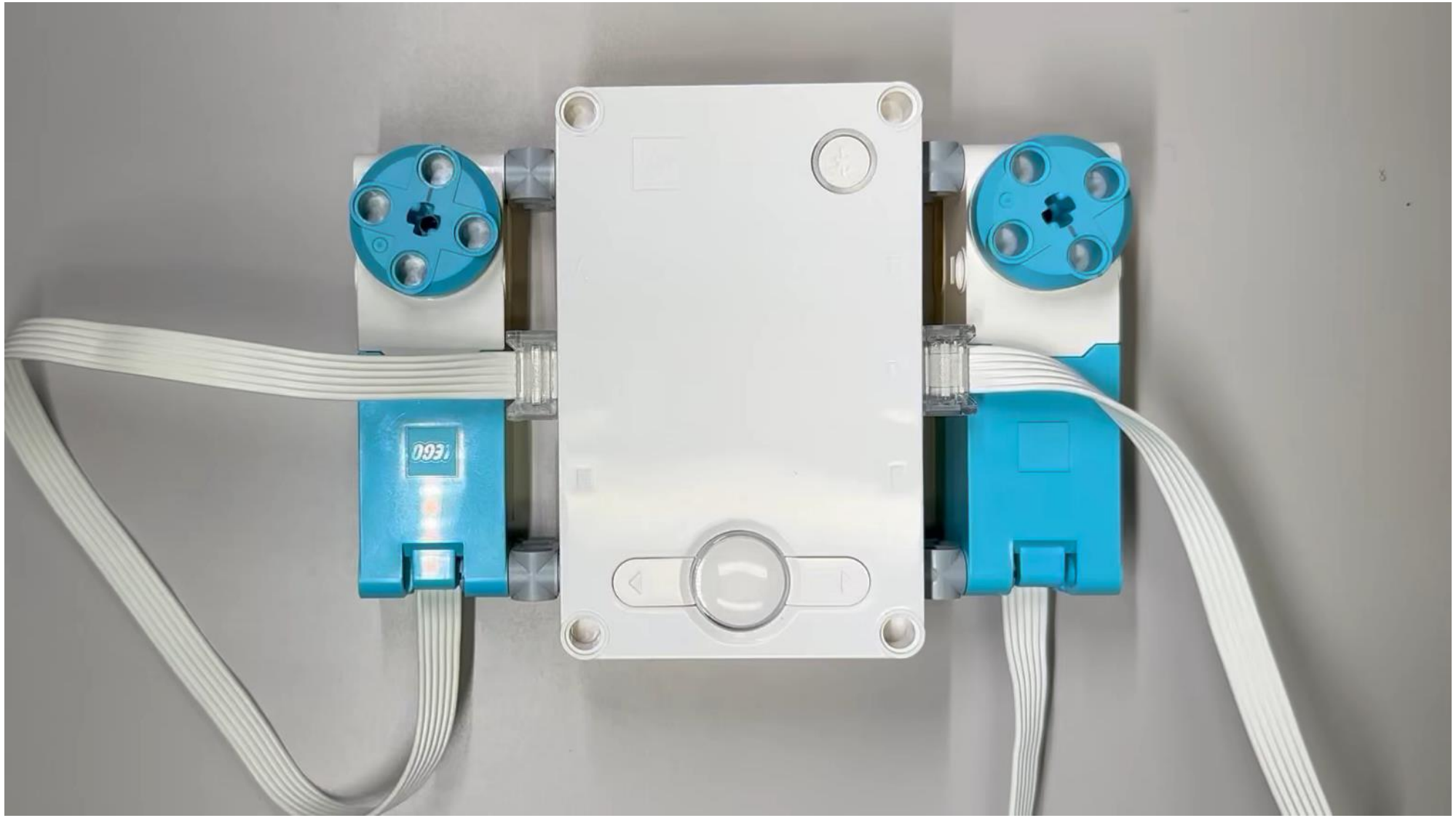
motor_task()

```
void
motor_task(intptr_t exinf)
{
  intptr_t rev_datta_pkt;
  struct data_packet *receive_pkt;
  pbio_error_t m_err;

  while (1) {
    rcv_dtq(MOTOR_DTQ,  &rev_datta_pkt);
    receive_pkt = rev_datta_pkt;

    if (!receive_pkt->is_setup)
     <省略(モータセットアップ処理)>

    if ((receive_pkt->idx % 2) == 0)
      pup_motor_set_speed(receive_pkt->motor,
                          receive_pkt->speed);
    else
      pup_motor_brake(receive_pkt->motor);
  }
}
```

データ受信

モータ出力

マルチタスクプログラミング (タスク間通信)



SPIKE-RTの環境構築方法

- 環境構築方法をYoutubeやGitHubで公開中
  - WSL環境で環境構築可能
  - https://www.youtube.com/watch?v=XJ7rltuf3Jc&t=5s

Youtube

GitHub

展示ブースで配布のチラシにも掲載

SPIKE-RTについてのまとめ

- このような人におすすめ
  - C言語の学習をしたい
  - 組込みシステムに興味がある
  - RTOS(マルチタスク)を学習したい
  - SPIKEを使用したロボットコンテストに出場する

SPIKE-RTのサンプルを公開中

目次

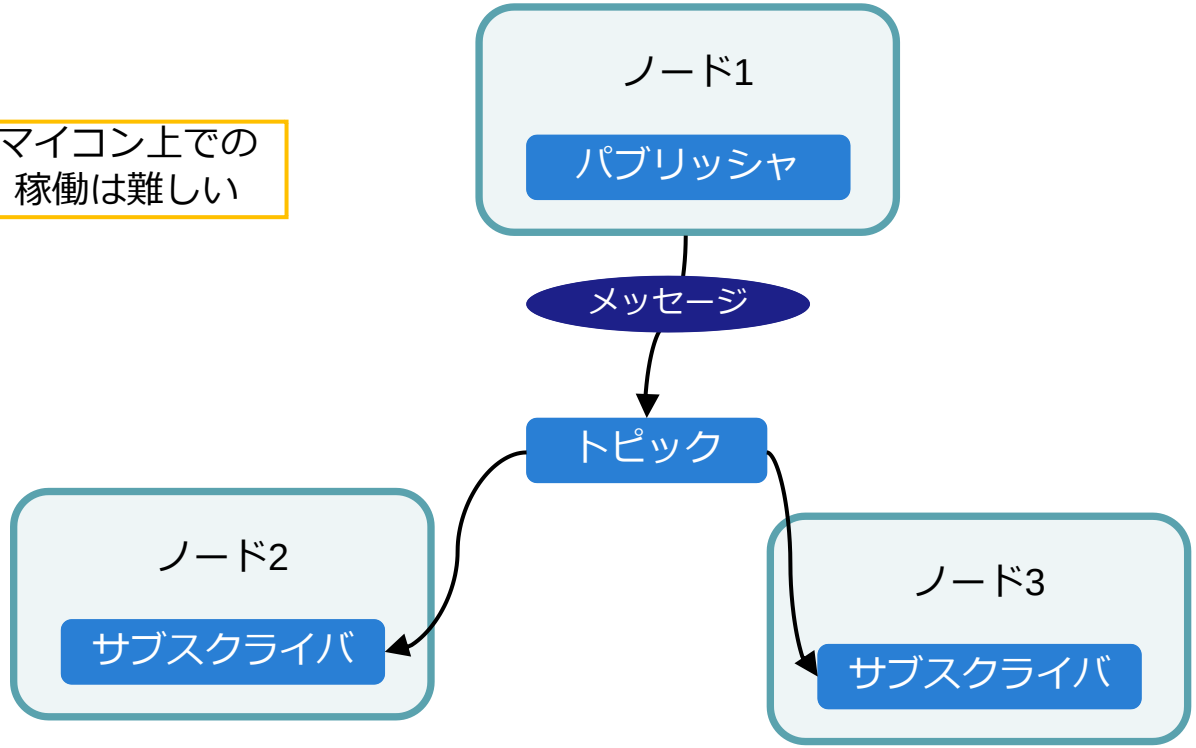
- 自己紹介
- SPIKE-RTの紹介
- SPIKEをROS 2で動かすための開発環境の紹介

ROS 2

- ROS 2とは
  - Robot Operating Systemの略
  - ロボットや自動運転向けの分散処理フレームワーク
    - 通信ミドルウェア
    - publish/subscribe通信が可能
  - UNIX系OS上での稼働を想定

マイコン上での
稼働は難しい

- 活用例
  - Amazon Roboticsの物流補助ロボット
  - aibo（SONY）

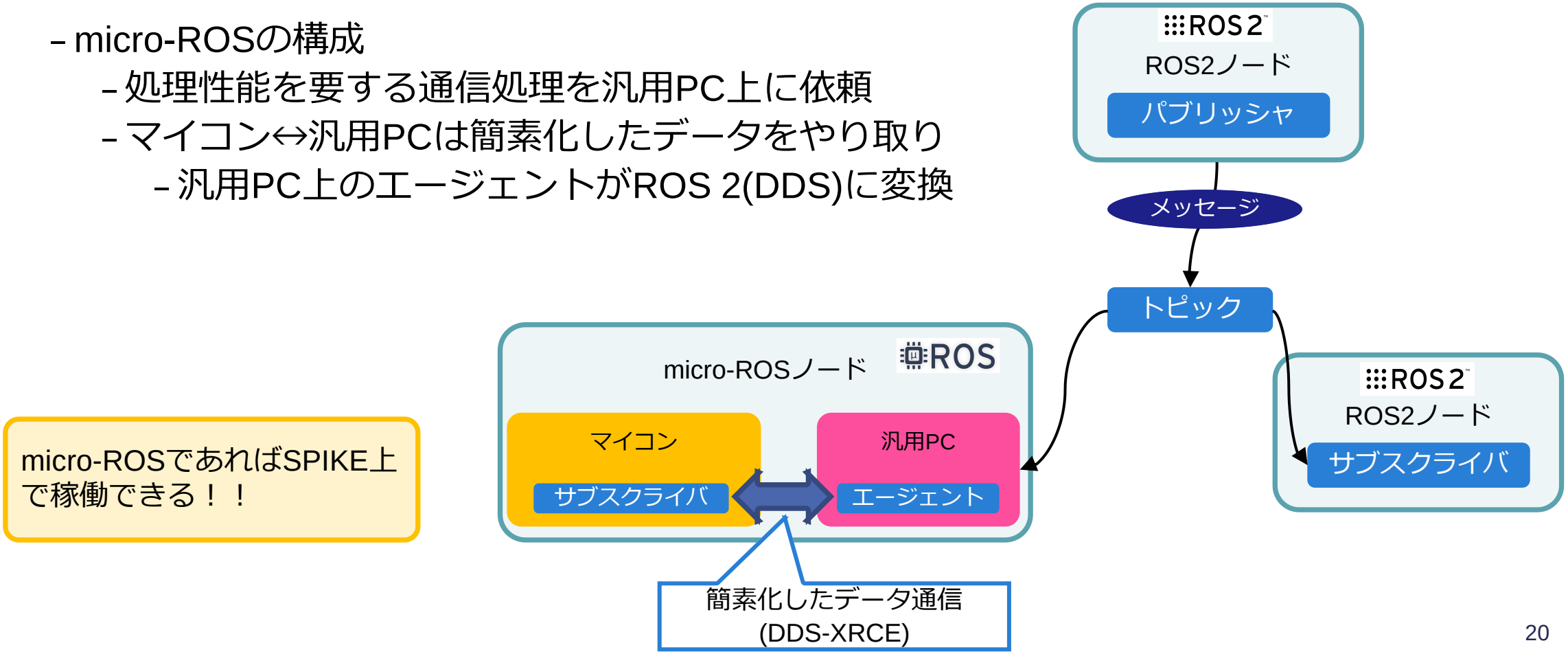


- SPIKE Prime Hubもマイコン
- ROS2の稼働は難しい

micro-ROS

- ROS 2との関係性
  - ROS 2のマイコン上での稼働は厳しい
  - マイコンをROS2に接続する為の機構

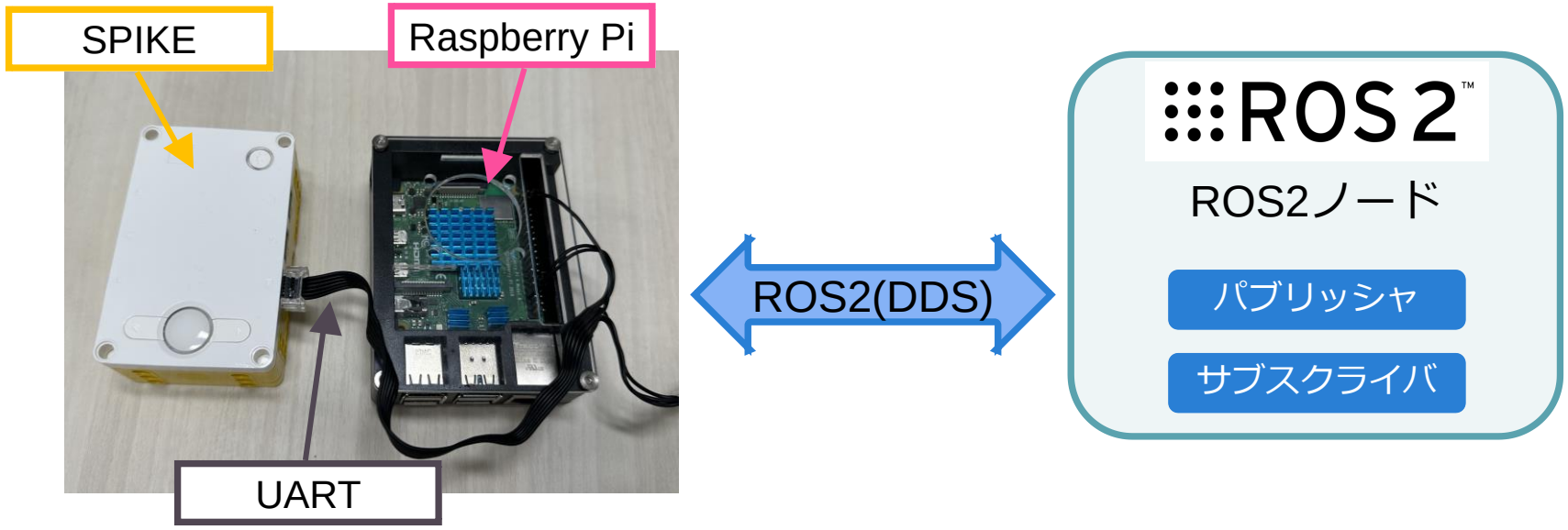
- micro-ROSの構成
  - 処理性能を要する通信処理を汎用PC上に依頼
  - マイコン↔汎用PCは簡素化したデータをやり取り
    - 汎用PC上のエージェントがROS 2(DDS)に変換



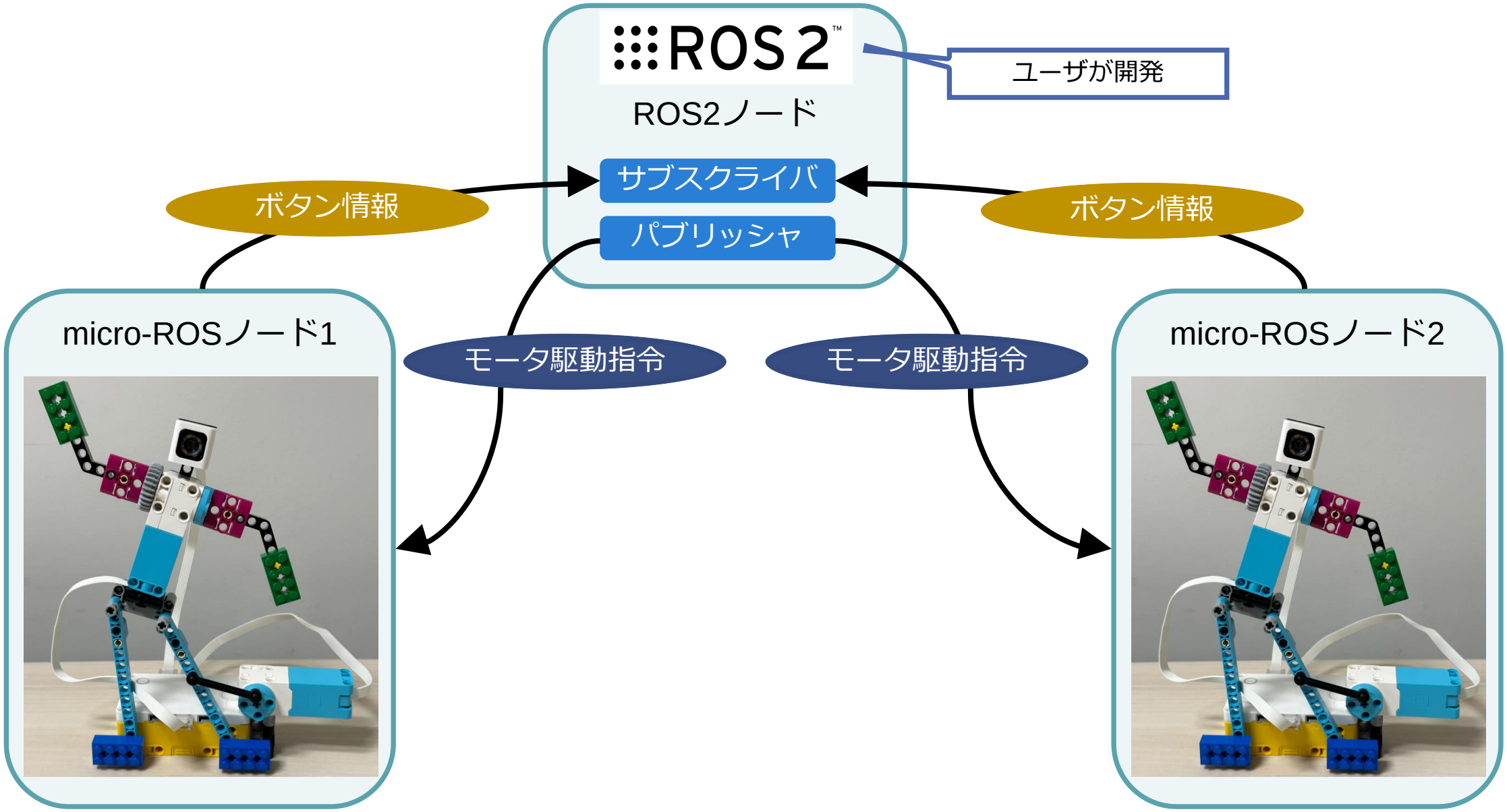
TOPPERSのROSへの取り組み

- micro-ROS_ASP3
  - TOPPERS/ASP3カーネル上で動作するmicro-ROSミドルウェア
  - LEGO SPIKE（SPIKE-RT）上での利用もサポート
  - SPIKE上でmicro-ROSプログラミングが可能
    - SPIKEをROS 2に接続する事が可能
    - ROSの教材として活用可能

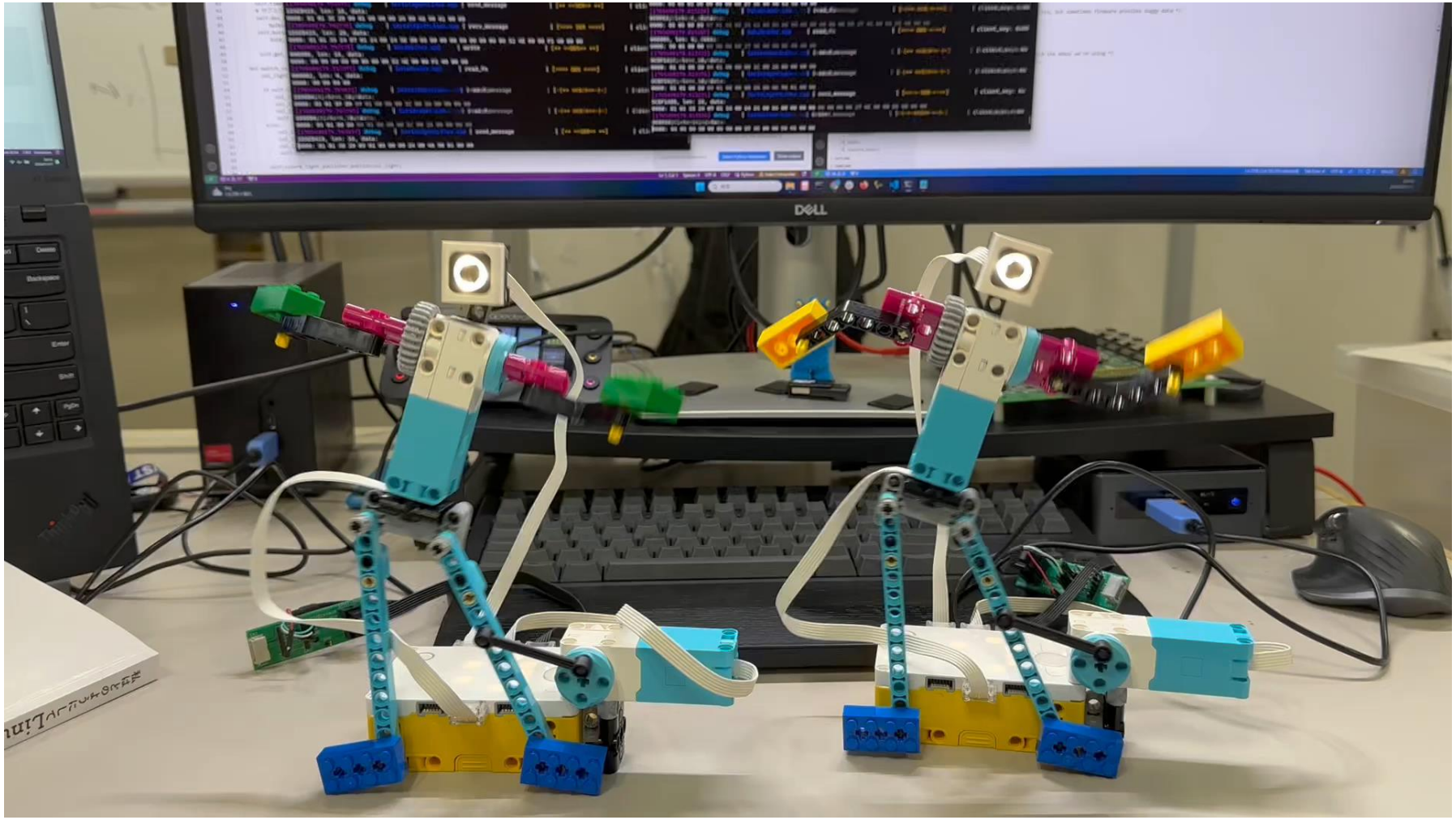
micro-ROS_ASP3



サンプル：ブレイクダンサー

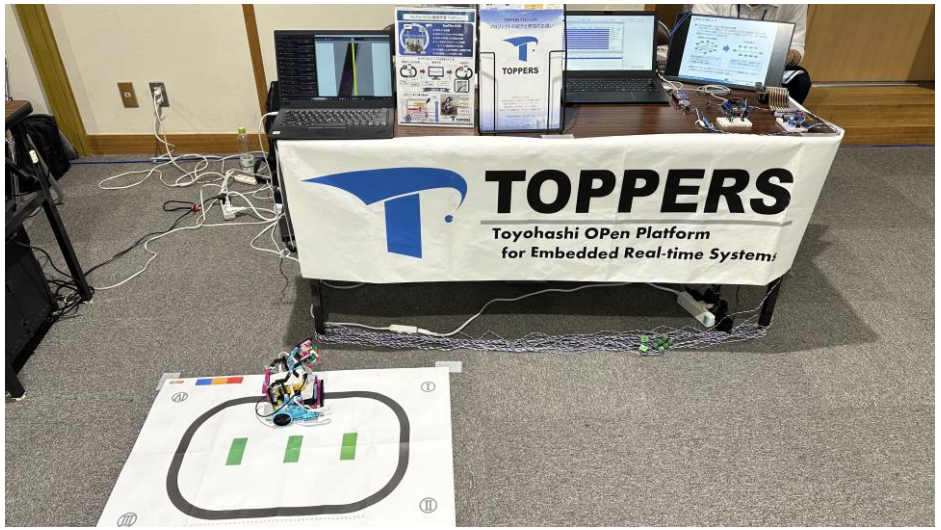


実行の様子



展示ブースの紹介

- LEGO SPIKEロボットを機械学習で自動走行するデモを展示中
  - SPIKE-RTと画像認識(Tensorflow)，ROSを使用



OSC Nagoya 2024
2024/05/25

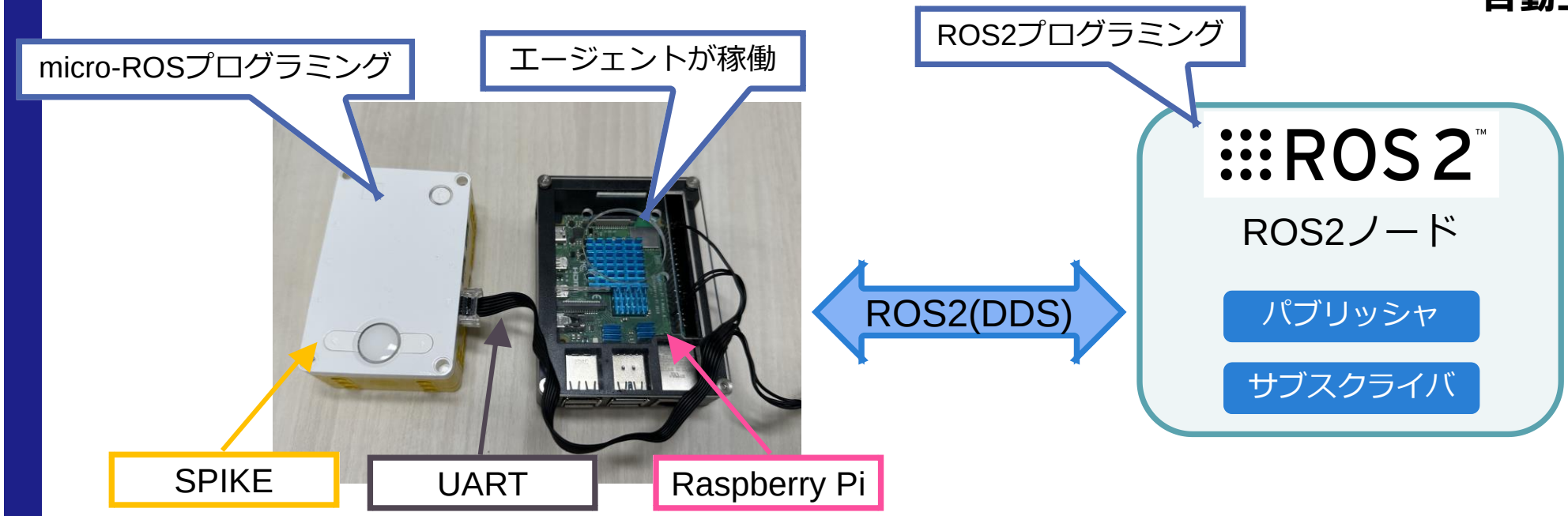
付録

樋山一樹
（南山大学 / TOPPERS）

micro-ROSファームウェアの自動生成ツール

- 自動生成ツール
  - micro-ROSプログラミングをせずにLEGOをROS 2で動かす
  - ユーザはHub側の構成を設定ファイルに記述
  - micro-ROSファームウェアを自動生成

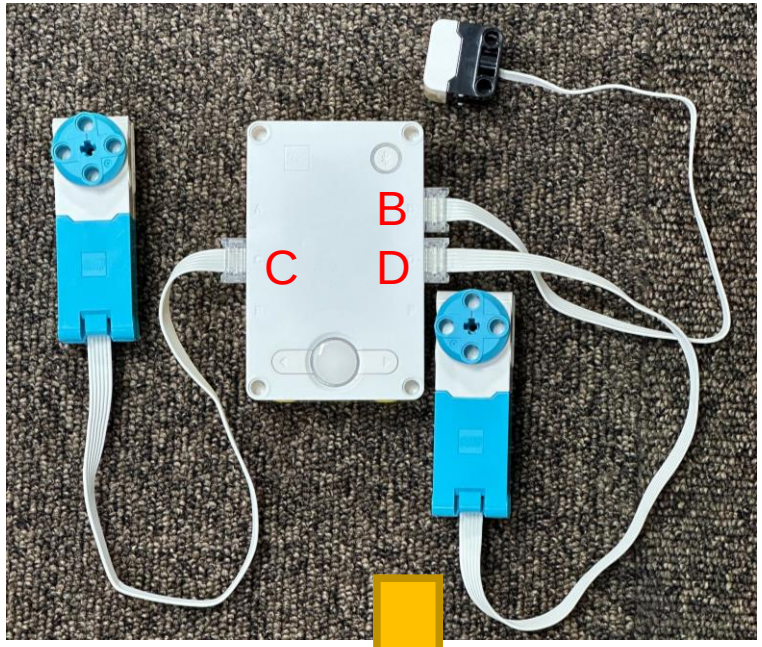
自動生成ツール

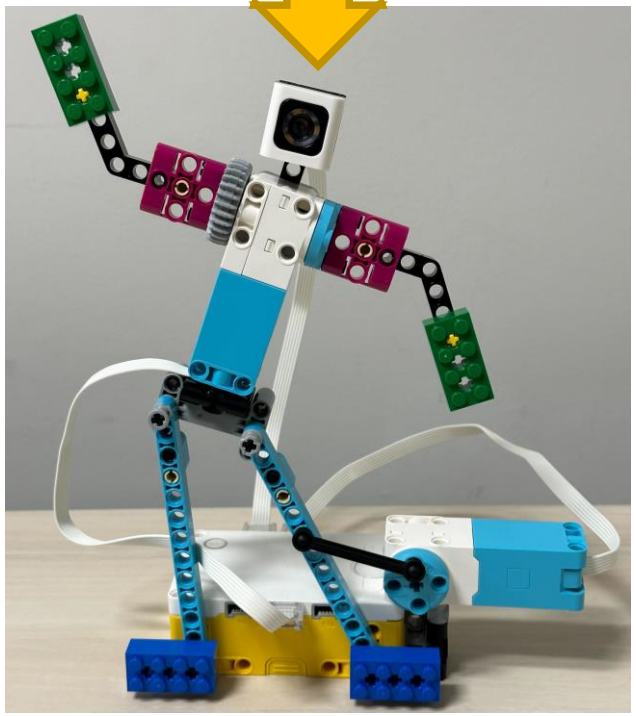


サンプル「ブレイクダンサー」

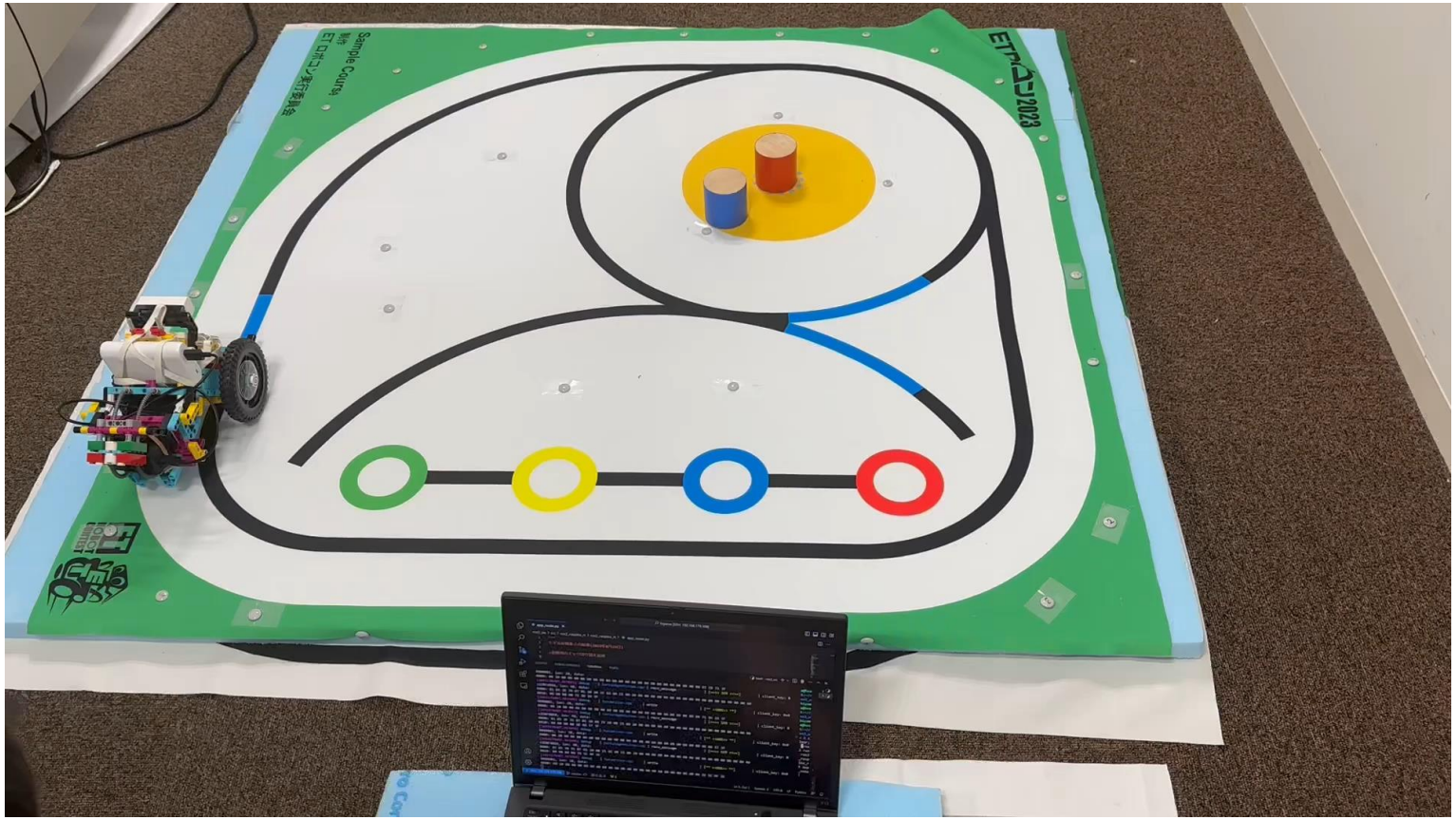
Yaml設定ファイル

```
PortB:
  device: color-sensor
  qos: best-effort
  enable_lights: True
  light_qos: best-effort

PortC:
  device: motor
  qos: best-effort
  wise: counter-clock
  run_mode: set-speed

PortD:
  device: motor
  qos: best-effort
  wise: clock
  run_mode: set-speed

hub:
  hub_program_cycle: 10
  enable_imu: False
  enable_battery_management: False
  enable_button: True
  enable_speaker: False
  opening: True
```





ETロボコン走行体をROS 2で動かす



動画：https://www.youtube.com/watch?v=RoaVhumuqcQ