組込みCI研究WG presents

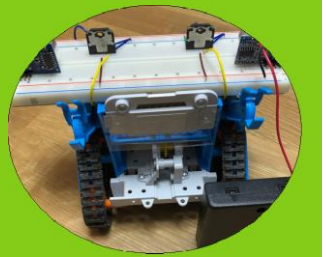
「組込み向けテスト実行フレームワークのご紹介」

組込みCI研究WG　見澤　広志

自己紹介

・見澤　広志
  ・大阪府の出身。
  ・半導体部品メーカを経て、
    今は自動車関連会社で
    ソフト開発に従事。
  ・JaSST Tokai実行委員
  ・組込みCI研究WG

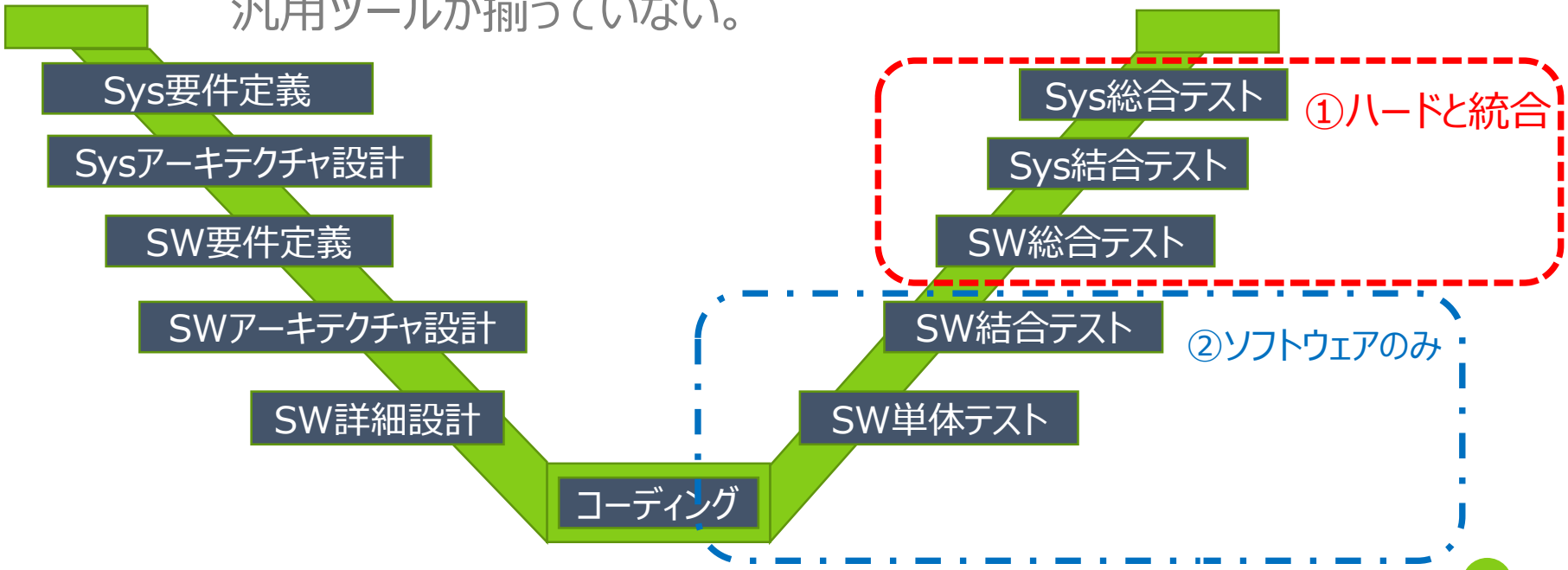
・最近の興味
  AWS、Docker、CI



・趣味
  子どもに動くおもちゃ作り
  リングフィットアドベンチャー(Lv 330)

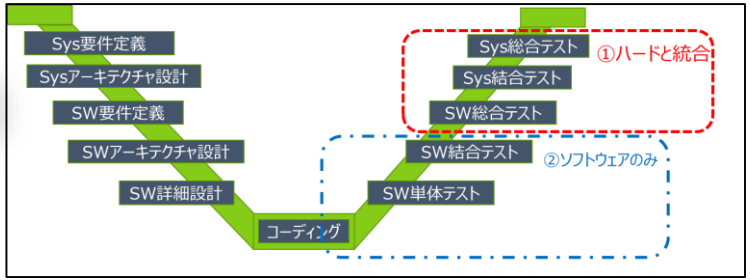
組込み製品のテストの課題

課題１　ハードと統合した状態でテストが必要。
課題２　ソフトウェアの機能確認を先行して不具合を潰しこみたいが、
　　汎用ツールが揃っていない。



組込みのテスト前倒しの課題

課題３　ソフトウェアだけのテスト環境を作ったとしても、
入出力が異なる為、テストケースを再利用しずらい。



【凡例】 実物　模擬

テスト環境②

テスト対象

入出力＝ソフトの入出力
（変数１＝250など）

マイコン

センサ　アクチュエータ

テスト環境
(SILS)

テスト環境②

テスト対象

入出力＝ソフトの入出力
（変数１＝250など）

センサ　アクチュエータ

テスト環境
(PILS/S-PILS)

テスト環境①

テスト対象
（ハードと統合）

センサ　アクチュエータ

入出力＝ハードの入出力
（ピン１＝２．５Vなど）

(HILS)

課題と対策

《課題》

- ハードと統合した状態でテストが必要。
- ソフトウェアの機能確認を先行して不具合を潰しこみたいが、汎用ツールが揃っていない。
- テスト環境毎に入出力が異なる為、テストケースを再利用しずらい。

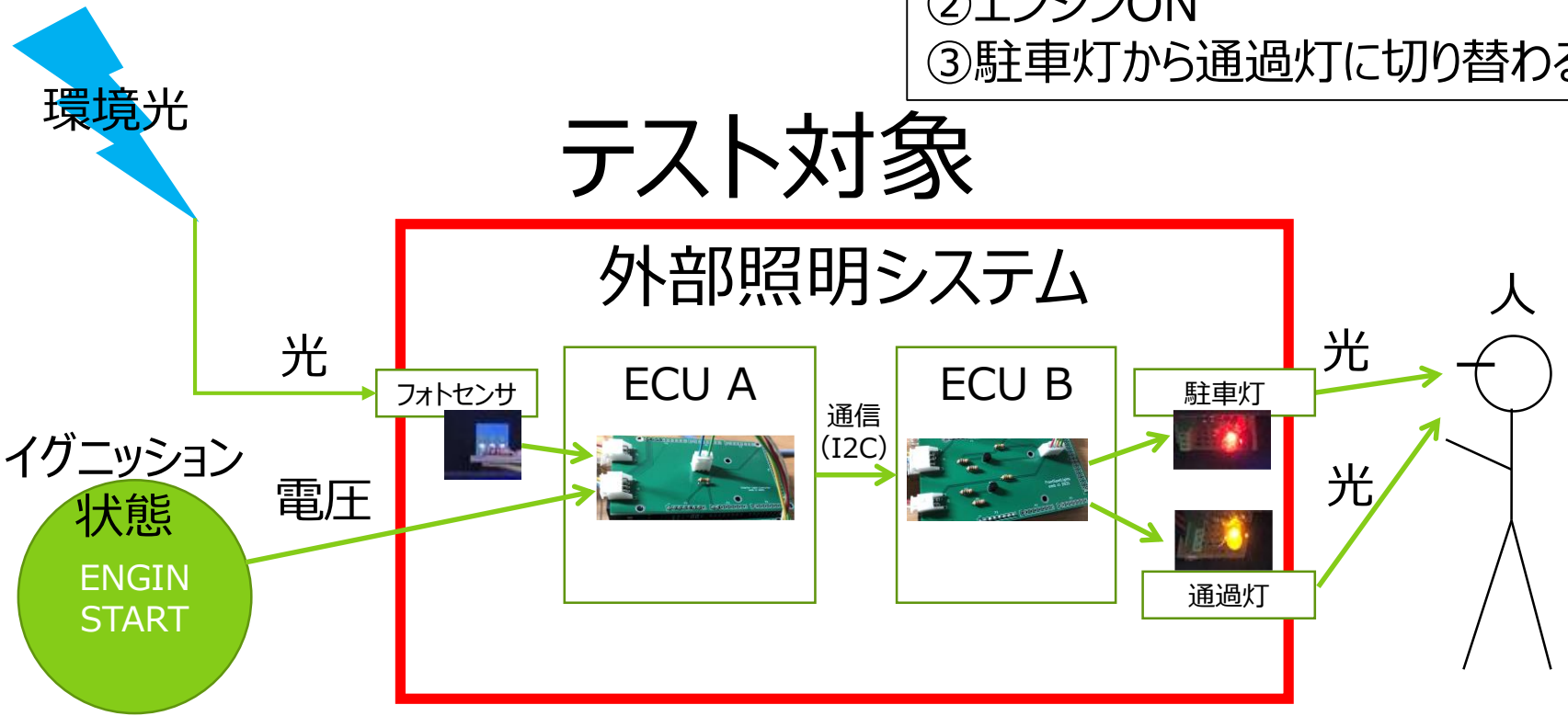
《対策》

- SILS環境をオープンソースを活用して作成する。
- シナリオレベルのテストケースをテスト環境で共通化する。
- テスト環境の違いは実装側（ライブラリ）で吸収する。

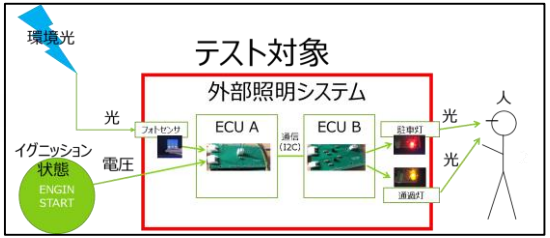
外部照明システム

《テストのシナリオ》

①夜間、駐車灯ON状態
②エンジンON
③駐車灯から通過灯に切り替わること



外部照明システムのテスト環境（HIL）



テスト環境①
テスト対象
（ハードと統合）
センサ
アクチュエータ
テスト環境
(HILS)

模擬
環境光
光
光

実物
フォトセンサ

外部照明システム

実物
外部照明
ECU

実物
通信（I2C）

実物
フロントスポット
ライトECU

実物
駐車灯

実物
通過灯

光
光

模擬
イグニッション状態

電圧

HIL装置
ライブラリ(HILS用)
テストケース（シナリオ）
ロボットフレームワーク

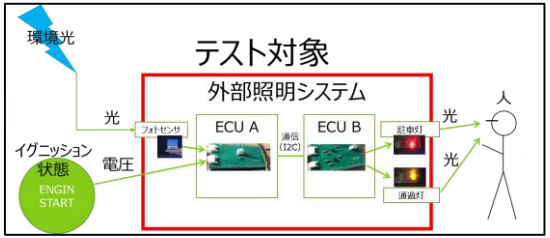
PWM
PWM

電圧
電圧

模擬(人)
フォトセンサ

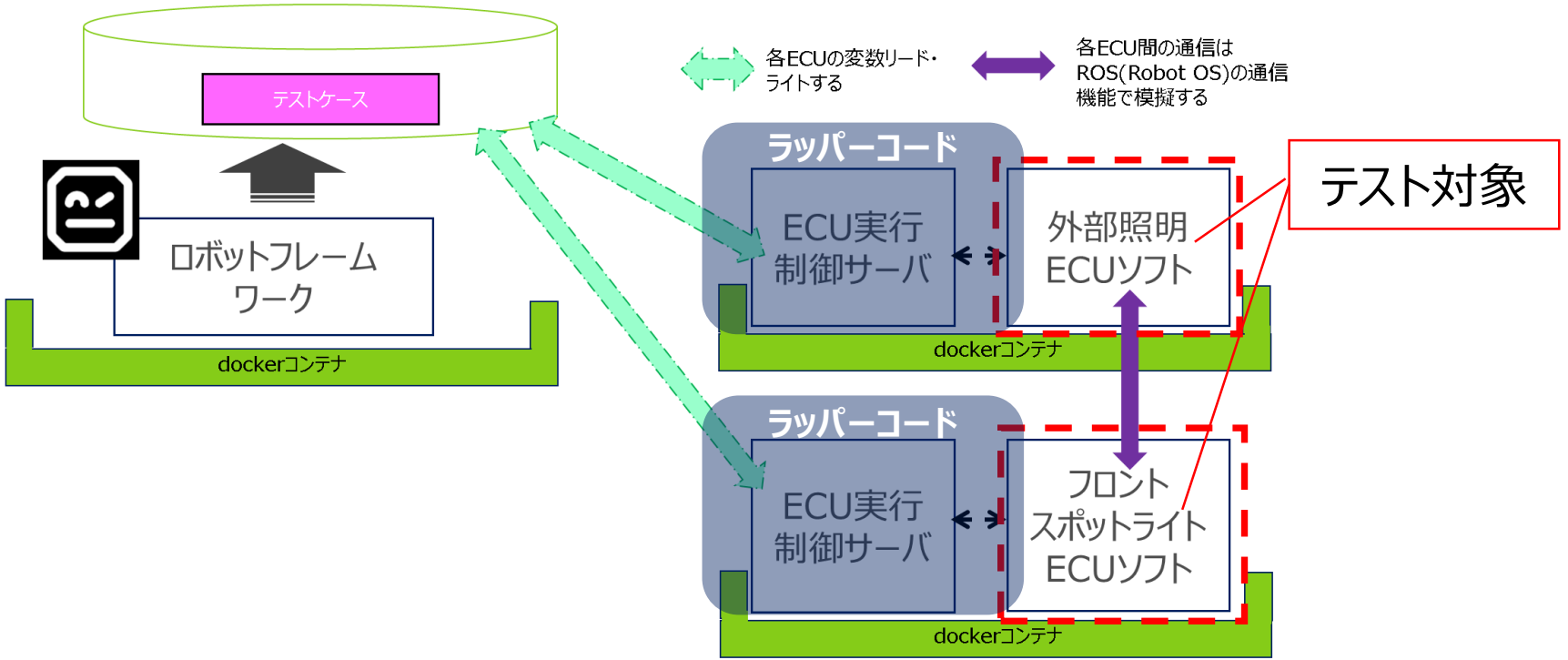
模擬(人)
フォトセンサ

外部照明システムのテスト環境（SIL）



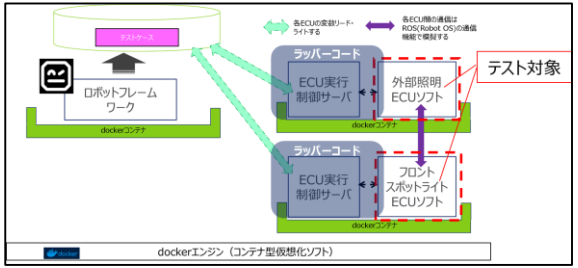
模擬
環境光
ライブラリ
数値（光）
模擬
フォトセンサ
ライブラリ
ラッパーコード
外部照明システム
外部照明ECU
模擬
スタブ
実物
ソフト
模擬
スタブ
模擬
ROSの
TCP/IP
フロントスポットライトECU
模擬
スタブ
実物
ソフト
模擬
スタブ
ラッパーコード
模擬
駐車灯
ライブラリ
模擬
通過灯
ライブラリ
数値（光）
数値（光）
模擬
イグニッション状態
ライブラリ
数値（電圧）
PWM（数値）
PWM（数値）
SIL装置
ライブラリ(SILS用)
テストケース（シナリオ）
ロボットフレームワーク
電圧（数値）
電圧（数値）
模擬(人)
フォトセンサ
ライブラリ
模擬(人)
フォトセンサ
ライブラリ

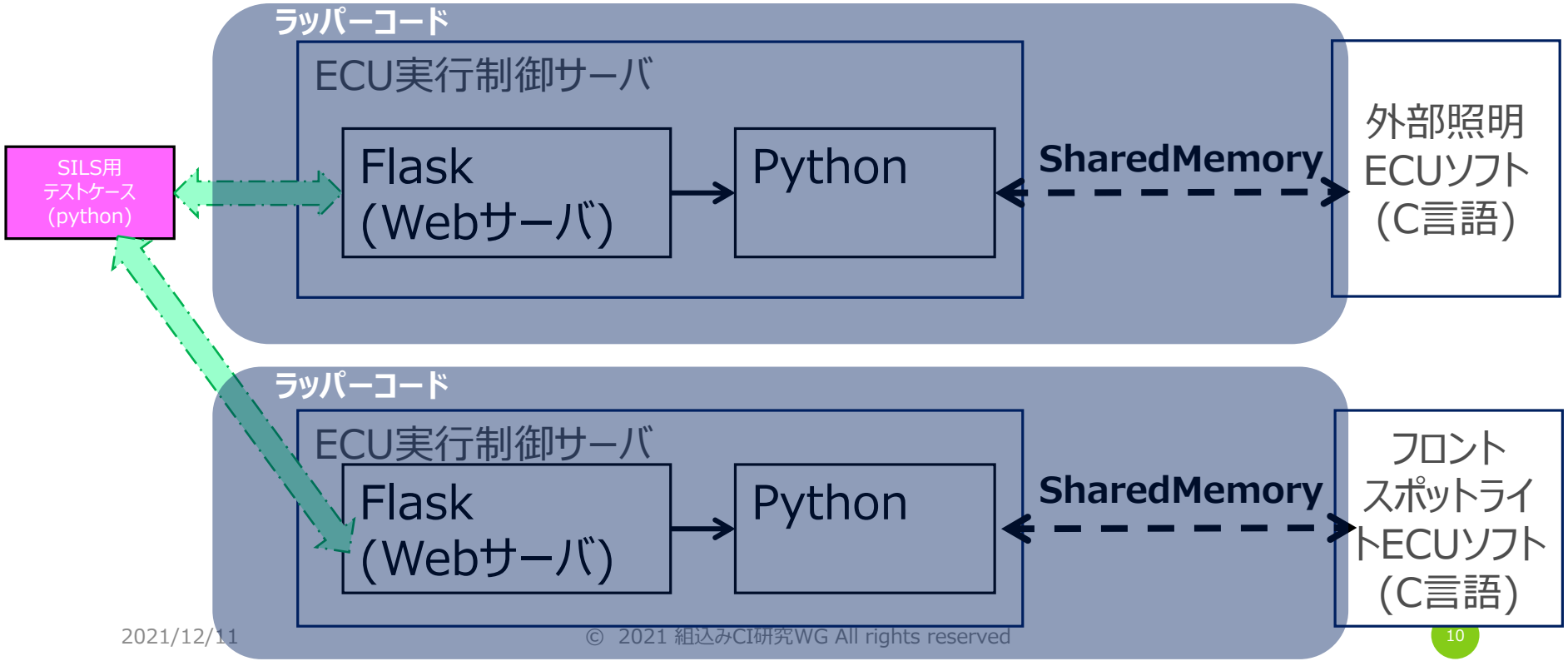
ラッパーコードの補足



dockerエンジン（コンテナ型仮想化ソフト）

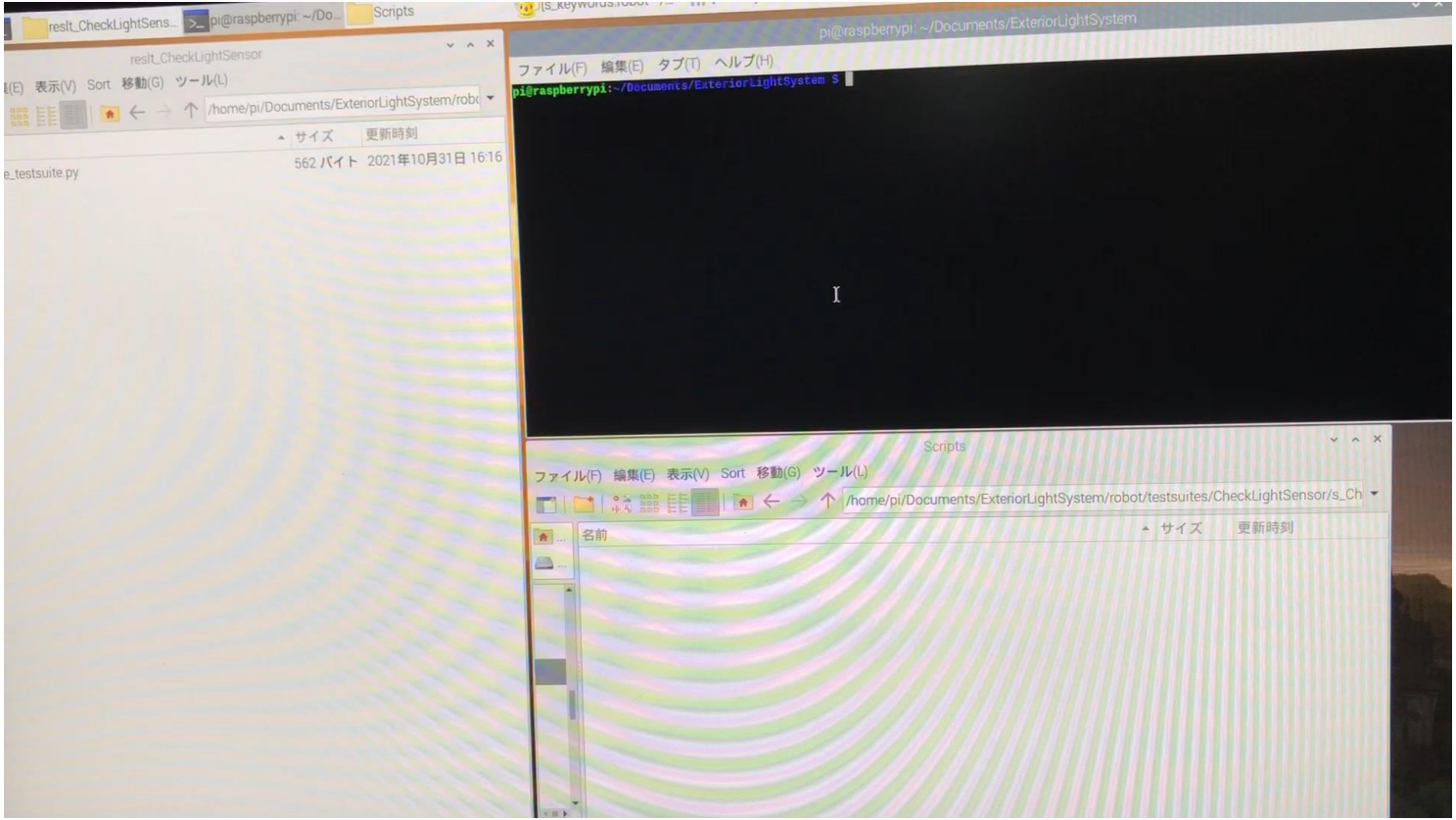
ラッパーコードの補足　～続き～





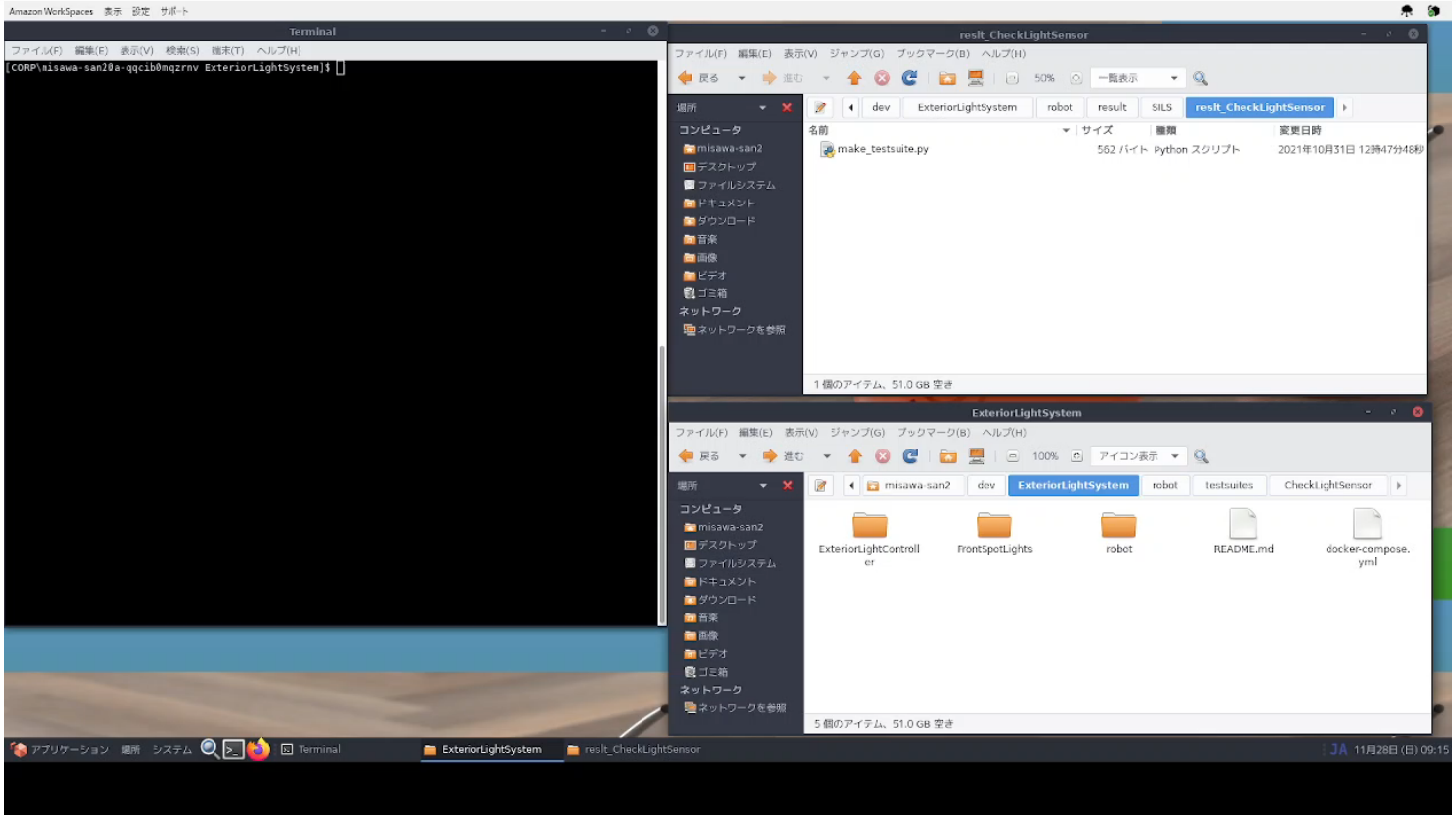
https://www.youtube.com/channel/UCMS_fCjRiB2XjakCR5fGLGQ

デモ（HILS環境）



https://www.youtube.com/channel/UCMS_fCjRiB2XjakCR5fGLGQ

デモ（SILS環境）



本ツールのメリット、デメリット

《メリット》

- ソースコードがあれば、**早期にテストを始めることができる。**（品質）
- 機能的な不具合を**実機が無くても**、潰すことができる。（品質）
- ソフトウェアの為、**テスト環境を増やすことができる**。（品質）
- シナリオレベルでテストケースを**再利用することができる。**（費用）
- オープンソースを使用している為、**安価に作成できる**。（費用）

《デメリット》

- IT系のツールを駆使する為、組込み系の人材には敷居が高い。
- ライブラリのメンテナンスするのが難しい。
- リアルタイム性能の検証に使用できない。

テスト実行結果レポート（SIL環境）

テストケース（HIL、SIL共通）

```
*** Settings ***
Resource  s_keywords.robot

*** Variables ***

*** Test Cases ***
エンジンの運転状態が検出された場合に、夜間の駐車灯から通過灯への自動移行することを確認(松モデル)
    [Tags]  HILS  SILS  req1  vari_matsu
    待ち時間[s]  =  0.5
    イグニッション状態  =  「キー未挿入」
    環境光  =  「夜」

    待ち時間[s]  =  0.5
    イグニッション状態  =  「IG OFF」

    待ち時間[s]  =  0.5
    イグニッション状態  =  「IG ON」

    待ち時間[s]  =  0.5
    Given 駐車灯状態  ==  「ON」
    And 通過灯状態  ==  「OFF」

    When イグニッション状態  =  「エンジン始動」
    And 待ち時間[s]  =  2

    Then 駐車灯状態  ==  「OFF」
    And 通過灯状態  ==  「ON」
    [Teardown]  テスト環境初期化
```

Full Name: CheckLightSensor.t CheckLightSensor.Scripts.t CheckLightSensor
Source: /usr/ecu_src/robot/testsuites/CheckLightSensor/t_CheckLightSensor/Scripts/t_CheckLightSensor.robot
Start / End / Elapsed: 20211031 12:50:15.271 / 20211031 12:50:15.748 / 00:00:00.477
Status: 3 tests total, 3 passed, 0 failed, 0 skipped

TEST **エンジンの運転状態が検出された場合に、夜間の駐車灯から通過灯への自動移行することを確認(松モデル)**

Full Name: CheckLightSensor.t CheckLightSensor.Scripts.t CheckLightSensor.エンジンの運転状態が検出された場合に、夜間の駐車灯から通過灯への自動移行することを確認(松モデル)
Tags: HILS, req1, SILS, vari_matsu
Start / End / Elapsed: 20211031 12:50:15.275 / 20211031 12:50:15.546 / 00:00:00.271
Status: PASS

KEYWORD s_keywords . **待ち時間[s]** = 0.5
KEYWORD s_keywords . **イグニッション状態** = 「キー未挿入」
KEYWORD s_keywords . **環境光** = 「夜」
KEYWORD s_keywords . **待ち時間[s]** = 0.5
KEYWORD s_keywords . **イグニッション状態** = 「IG OFF」
Start / End / Elapsed: 20211031 12:50:15.367 / 20211031 12:50:15.376 / 00:00:00.009
KEYWORD slib . **Set Ig Stat** @{igstat}
Documentation: イグニッション状態を設定
Start / End / Elapsed: 20211031 12:50:15.368 / 20211031 12:50:15.376 / 00:00:00.008
12:50:15.376 INFO Web API Write Response : {'ecuname': 'ExteriorLightSystem', 'err': '0', 'mem_idx': 3, 'readback': '143', 'res_ecu_time': 1.0000000000000007} : memname : adc_data[IGV]
12:50:15.376 INFO IG設定値 : 「IG OFF」 duty:14[%] memname:adc_data[IGV]
KEYWORD s_keywords . **待ち時間[s]** = 0.5
KEYWORD s_keywords . **イグニッション状態** = 「IG ON」
KEYWORD s_keywords . **待ち時間[s]** = 0.5
KEYWORD s_keywords . **Given 駐車灯状態** == 「ON」
KEYWORD s_keywords . **And 通過灯状態** == 「OFF」
KEYWORD s_keywords . **When イグニッション状態** = 「エンジン始動」
KEYWORD s_keywords . **And 待ち時間[s]** = 2
KEYWORD s_keywords . **Then 駐車灯状態** == 「OFF」
KEYWORD s_keywords . **And 通過灯状態** == 「ON」

テスト実行結果レポート（SIL環境）

GCC Code Coverage Report

Directory: src/pubsub/src/app/
Date: 2021-10-31 03:50:16
Legend: low: >= 0% medium: >= 75.0% high: >= 90.0%

	Exec	Total	Coverage
Lines:	67	73	91.8%
Branches:	8	12	66.7%

File	Lines		Branches	
ExteriorLightController.c	100.0%	26 / 26	100.0%	4 / 4
adc.c	73.3%	11 / 15	-%	0 / 0
calc_v.c	93.8%	30 / 32	50.0%	4 / 8

Generated by: GCOVR (Version 5.0)

GCC Code Coverage Report

Directory: src/pubsub/src/app/
File: src/pubsub/src/app/calc_v.c
Date: 2021-10-31 03:50:16

	Exec	Total	Coverage
Lines:	30	32	93.8%
Branches:	4	8	50.0%

Line	Branch	Exec	Source
1			#include <adc.h>
2			#include <calc_v.h>
3			
4			uint16_t calc_v_env_light = 0U; /* ENV_LIGHT : unit mV */
5			uint16_t calc_v_igv = 0U; /* IGV : unit mV */
6			
7			void calc_v_step(void);
8			void calc_v_set_env_light(uint16_t env_light);
9			uint16_t calc_v_get_env_light(void);
10			void calc_v_set_igv(uint16_t igv);
11			uint16_t calc_v_get_igv(void);
12			
13		451	static void calc_v_ad2env_light(void)
14			{
15		451	uint32_t val = 0UL;
16			
17		451	val = adc_get_env_light();
18			
19	▶ 1/2	451	if(val < 1024UL)
20			{
21		451	val = CALC_V_ENV_LIGHT_LSB * val;
22	▶ 1/2	451	if(val > 65535UL)
23			{
24		✗	val = 65535UL;
25			}
26		451	calc_v_set_env_light((uint16_t)val);
27			}
28			else
29			{
30			/* 前回値保持 */
31			}
32		451	}

異常系処理の為、
実行されなかった。

入手方法

SILS/HILS環境で共通テストケースを使用できるか検証するため、
下記のテスト実行フレームワークを作成しました。
下記URLから、デモで使用したテストケース付きの、環境一式が入手できます。

「組込み向けテスト実行フレームワーク」
https://github.com/misawa-san/ExteriorLightSystem

実行方法（SIL環境）

docker環境で動作させてください。

■操作手順

①git clone https://github.com/misawa-san/ExteriorLightSystem.git

②cd ExteriorLightSystem

③sudo chmod -R 777 .

④docker-compose -f docker-compose.yml up -d --build

⑤docker-compose -f docker-compose.yml up -d --build

参考

テストケース

```
*** Settings ***
Resource  s_keywords.robot

*** Variables ***

*** Test Cases ***
エンジンの運転状態が検出された場合に、夜間の駐車灯から通過灯への自動移行することを確認(松モデル)
    [Tags]  HILS  SILS  req1  vari_matsu
    待ち時間[s]  =  0.5
    イグニッション状態  =  「キー未挿入」
    環境光  =  「夜」

    待ち時間[s]  =  0.5
    イグニッション状態  =  「IG OFF」

    When
    イグニッション状態  =  「IG ON」

    待ち時間[s]  =  0.5
    Given 駐車灯状態  ==  「ON」
    And 通過灯状態  ==  「OFF」

    When イグニッション状態  =  「エンジン始動」
    And 待ち時間[s]  =  2

    Then 駐車灯状態  ==  「OFF」
    And 通過灯状態  ==  「ON」
    [Teardown]  テスト環境初期化
```

キーワードを並べてテストケースを作る（キーワード駆動スクリプティング）

```
イグニッション状態  =
    [Arguments]          @{igstat}
    slib.set_ig_stat     @{igstat}
```

```
def set_ig_stat(self, key):
    """イグニッション状態を設定"""

    file = open(DPATH + 'd_set_ig_stat.json')
    reqs = json.load(file)

    req = reqs[self.tenv][key]
    self.t.set_ig_stat(key, req)
```

SILS環境の場合

```
def set_ig_stat(self, key, req):
    """イグニッション状態を設定"""

    # Exterior Light ECUの adc_data[IGV]にA/D値を設定
    w_val = int(1024 * (req['duty[%]']/100))
    response = self.set_ecu_value(req["hostname"], req["memname"], w_val)

    # Web APIの戻り値が正常(0)の場合、
    if "0" == response["err"]:
        logger.info(f"IG設定値  :  {key}  duty:{req['duty[%]']}[%]  memname:{req['memname']}")
    # Web APIの戻り値が正常(0)以外の場合、A/D範囲外を設定
    else:
        logger.info("Web API response Error : ")
```

HILS環境の場合

```
def set_ig_stat(self, key, req):
    """イグニッション状態を設定"""

    # PWM出力
    self.pi.hardware_PWM( req['Port'], req['freq[Hz]'], req['duty[%]']*10000 )

    logger.info(f"IG設定値  :  {key}  Port:{req['Port']}  freq:{req['freq[Hz]']}[Hz]  duty:{req['duty[%]']}[%]")
```

実装テクニックにより、
SILS環境とHILS環境のどちらも、
同じテストケースを使用する。

組込みCI研究WG参加者募集

組込みCI研究WGでは、継続的インテグレーション、継続的テスト
継続的デプロイなどの研究を実施しております。
もし、興味がありましたら、ご参加ください。

遠隔地からはオンラインでの参加もOKです。

月1度、主に日曜日の13:30～17:00で実施しております。
最近ではオンラインが主体となってきています。

以下のメールアドレスまで連絡をお願いします。
aster-embci@qualab.jp

ご清聴ありがとうございました