# ESP32-S3マイコンボード インストールマニュアル



1.0版 株式会社アールティ



# 目次

目次	1
ご使用になる前に	2
製品に対する注意事項	2
安全に関する注意事項	3
内容物一覧	4
主な仕様	5
本体の仕様	5
本製品の端子仕様	7
コンピュータの動作要件	8
組立	9
本製品を単体で使用する場合	9
環境設定	10
コマンドとArduinoスケッチの表記について	10
はじめに	11
Arduino IDEのインストール	11
Arduino IDEの初期設定	18
サンプルスケッチのビルドと書き込み	25
micro-ROSの環境設定	33
PCとPi:Co Classic3間の通信確認	38
使用しているツール、OSSのバージョン	40
製品保証	41
保証の内容	41
保証者の名称、所在地および電話番号	41
保証期間	41
保証の適用	41
保証の除外事項	42
保証の態様	42
お客様の費用負担	43
保証を受けるための手続き	43
お問い合わせ	44
改訂履歴	44
Copyright・知的財産権について	44



## ご使用になる前に

この度は、弊社の「ESP32-S3マイコンボード(以下「本製品」といいます)」をお買い 上げいただき、誠にありがとうございます。本製品をご使用になる前に、本書をお読みいた だきますようお願いいたします。

## 製品に対する注意事項

- 初めてロボットを使用される方は、経験者と一緒に作業することをお勧めします。
- 製品到着後、本書の内容物一覧と照らして必ず本体および付属品のご確認をお願いします。
- 製品の品質、検品および発送には万全を期していますが、万一お届けした製品に不良、破損(輸送中の本体破損も含みます)、付属品の不足がありましたら、製品到着後7日以内に弊社営業サポート(sales@rt-net.jp)までご連絡いただきますようお願いいたします。部品または本体の交換、不足品の発送等を行い、無償にて対応いたします。製品到着後8日以上が経過した場合、無償対応はいたしかねますのでご了承いただきますようお願いいたします。
- 製品の仕様および外観ならびにweb上で公開しているデータおよび情報は、改良のため予告なく変更することがあります。改良版は、ご購入時点の製品、データおよび情報と異なる可能性がありますが、異なる点について交換、返金、返品、改変等はいたしかねますのでご了承いただきますようお願いいたします。
- 本製品は、製造工程の特性により、フレームや部品に傷が付くことがあります。これらの傷は、本体の運転に支障を来さないため、保証の適用外とさせていただきます。
- 本書は、Windows、macOSまたはLinuxの基本的な使用方法を理解している方を対 象として構成されています。
- 本書は、本製品を対象としたmicro-ROSサンプルソフトウェアについて解説いたしません。
- 本書において記載されております画像、その他記載されている会社名、製品名等の 固有名詞は各社の登録商標または商標です。なお、本文中では、TM、(R)マークは省 略しております。



## 安全に関する注意事項

### 警告表示について

マーク	マークの定義
⚠️ 危険	「 <mark>危険</mark> 」を表します。 取り扱いを誤った場合に、死亡または重傷を負う状態が生じることが想定 され、かつ火災発生等の財物に重大な損害が生じる緊急性の高い事項を表 します。
⚠ 注意	「 <mark>注意</mark> 」を表します。 取り扱いを誤った場合に、軽傷または物理的損害が生じることが想定され る事項を表します。

## 危険内容および対応方法

	危険内容(行為、現象)	マーク	対応方法
作業時	Pi:Co Classic3に本製品を搭載し て走行した時、周辺の金属に触 れてショートする。または、本 製品の操作時に金属物(指輪や ネックレス等)が触れてショート する。	⚠ 危険	運用時、周辺に金属物がないことを確認し てください。また、操作時に金属製の指輪 等を外した状態で操作してください。
組立時	Pi:Co Classic3にバッテリを繋げ た状態で本製品を交換する。	⚠ 危険	Pi:Co Classic3の電源スイッチをOFFにし、 バッテリを外した状態で本製品を交換して ください。
                	本製品のJP1パッドをショート した状態で本製品をPi:Co Classic3に取り付け、Pi:Co Classic3の回路が破壊される。	⚠ 注意	JP1パッドをショートした場合は、本製品 をPi:Co Classic3に取り付けないでくださ い。



## 内容物一覧

本製品セットに含まれている物品は次のとおりです。

番号	物品	数量
1	ESP32-S3マイコンボード	1
2	USB Type-A - Type-C ケーブル	1
3	技適マークシール	1
4	入門テキストのダウンロードカード ※ <b>テキスト同梱版(RT-ESP32-S3-P3.0)のみに付属します</b>	1



①ESP32-S3マイコンボード



②USB Type-A - Type-C ケーブル



③技適マークシール



④ダウンロードカード



# 主な仕様

## 本体の仕様

本体の仕様は次のとおりです。

項目	仕様	
製品名	ESP32-S3マイコンボード	
型番	RT-ESP32-S3-P3.0、RT-ESP32-S3-P3.0_NT	
入力電源(定格電圧)	ピンヘッダからの供給:3.3V±10% 200mA以上 USBコネクタからの供給:5V±10%	
CPU	Espressif Systems ESP32-S3-WROOM-1-N16R8	
動作周波数	240MHz	
ROM	384kByte	
SRAM	512kByte	
SPI Flashメモリ	16MByte	
PSRAM	8MByte	
書き込み	USB	
寸法	34x37x14mm	
重量	10g	
モード切替スイッチ	ESP32-S3の起動モードを切り替えるスイッチです。RUN1のシ ルク側にスライドするとスケッチ実行モードで、反対側にスラ イドするとスケッチ書き込みモードで起動します。	
リセットボタン	ESP32-S3のプログラムを再起動するボタンです。モード切替 スイッチ操作後にこのボタンを押すと、切り替え後のモードで 起動します。Pi:Co Classic3の電源をOFF->ONしてもプログラ ムを再起動できます。	
JP1パッド	本製品をUSB電源で駆動する際に使用します。 JP1パッドをショートするとUSB電源から生成した3.3Vを ESP32-S3に供給します。 JP1パッドをショートした後は、本製品をPi:Co Classic3に取 り付けないでください。	

本製品の本体仕様一覧



#### ESP32-S3マイコンボード インストールマニュアル



本製品の外観



## 本製品の端子仕様

ESP32-S3の端子仕様に対して本製品で追加した機能の注意点を記載します。下記の接続 先一覧表を参照してください。接続先の括弧内にESP32-S3のピン名を記載しています。NC はNo Connectの略称です。

IO19(PR12)、IO20(PR9)はUSBにつながっています。USBを使わないときにIO19と IO20を使用できます。

IOO(PR6)は、書き込みモード、ユーザブートモードの切り替えピンです。ブート時に GNDと接続されるため、GPIOとして使用することは推奨しません。

IO45(PL15)とIO46(PR7)はGNDに対して10kΩでプルダウンしてあります。IO46は Pi:Co Classic3の機能と接続していないためGPIOとして使用できます。GPIOとして使用す る際はアクティブHIGHの出力として使用することを推奨します。

IO4 (PL22) 、IO5 (PL23) 、IO6 (PL24) 、IO7 (PL25) は3.3Vに対して1MΩでプル アップしてあります。

外側のピン	内側のピン		内側のピン	外側のピン
NC	NC	PL PR 28 27 RUN1 1 1 2	SW_R(IO12)	SW_C(I011)
NC	AD4(IO7)	<b>26 25</b>	SW_L(IO10)	MOTOR_EN(IO9)
AD3(IO6)	AD2(IO5)	24 23	NC	MD(100)
AD1(IO4)	AD0(IO8)	-22 21 U5 7 8	(IO46)	NC
(IO48)	GND	20,19 , 288822222222 9.10	(IO20,USB D+)	(103)
(IO47)	NC	18, 17 3 889LNOW 11, 12	RESET(EN)	(IO19,USB D-)
NC	PWM_L(IO45)	16.15 (ET) ENGLEY 13.14	NC	NC
NC	CW_L(IO21)	14-13 (2) (10 maximization of 2) (15, 16)	NC	RXD0
CW_R(IO14)	PWM_R(IO13)	12 11 B	TXD0	NC
AD0(IO8)	buzzer(IO38)	02 19 20	SLED_L(IO17)	SLED_R(IO18)
BLED1(IO39)	BLED0(IO40)		SLED_FL(IO15)	SLED_FR(IO16)
LED3(IO41)	LED2(IO42)	6 5 <sup>1</sup> 1 12	NC	NC
LED1(IO2)	LED0(IO1)	<sup>4</sup> <sup>3</sup> <sup>5</sup> <sup>25</sup> <sup>26</sup>	NC	NC
3.3V	3.3V	2 <sup>2</sup> 1 27 28	NC	NC

ピンからPi:Co Classic3への接続先一覧(括弧内にはESP32-S3のピン名を記載)



## コンピュータの動作要件

本製品を使用するに当たり、下記の要件を満たすPCをご用意ください。

#### Linux

項目	仕様
os	Ubuntu 22.04(推奨)
CPU	1GHz デュアルコア以上
メモリ	4GB以上
ストレージ	128GB以上
接続	USB、Wi-Fi

#### Windows

項目	仕様
OS	Windows 11
CPU	1GHz デュアルコア以上
メモリ	4GB以上
ストレージ	128GB以上
接続	USB、Wi-Fi

#### macOS

項目	仕様
os	Monterey 12.6 or Ventura 13.0
CPU	Intel Core i5 1.6GHz以上
メモリ	4GB以上
ストレージ	128GB以上
接続	USB、Wi-Fi



## 組立

Pi:Co Classic3に取り付けられているCPU基板(RX631搭載CPUボード)を取り外して、 本製品に置き換えます。このとき、下記写真のとおり本製品のUSBコネクタが表示用LEDの 方に向くように本製品のピンをソケットに挿してください。取り付け後、ピンが曲がってい ないか、ずれていないか確認してください。



本製品をPi:Co Classic3に正しく取り付けた状態

## 本製品を単体で使用する場合

本製品は、製品裏面にあるJP1パッドをショートすることで、Pi:Co Classic3から取り外 して開発することが可能です。この場合、USBのバスパワーで本製品のCPUを駆動できま す。ただし、JP1パッドをショートした後は、本製品をPi:Co Classic3に取り付けないでく ださい。Pi:Co Classic3の電源回路を破壊する恐れがあります。



JP1パッド

製品出荷時のJP1パッドの状態



## 環境設定

ここでは本製品の開発環境の構築手順について説明します。

## コマンドとArduinoスケッチの表記について

本書では、Ubuntuのターミナルで実行するコマンドを下記のように表記します。"Unset" より下の行がコマンドを表します。ただし、"#"で始まる行はコメントを表すため実行不要 です。

```
Unset
$ echo "コマンド"
# この行はコメントです
```

Arduinoスケッチは下記のように表記します。"C/C++"より下の行がスケッチの内容です。

```
C/C++

void setup(){

// ピンモードを出力に設定します

pinMode(LED0, OUTPUT);

}

void loop(){

// LEDを点滅させます

digitalWrite(LED0, HIGH);

delay(500);

digitalWrite(LED0, LOW);

delay(500);

}
```



## はじめに

本製品はArduino Foundationが提供するArduino IDEを開発環境として使用します。 Arduino IDEはマルチプラットフォームに対応しています。そのため、本製品は3種のOS( Windows、macOS、Linux)で開発できます。

また、本製品はROS 2(Robot Operating System 2)というオープンソースソフトウェア にも対応しています。ROS 2を用いることで他の開発者が提供するソフトウェアを活用で き、3Dビジュアライザや、ゲームパッドによる遠隔操作等の機能をPi:Co Classic3に取り入 れられます。本製品はmicro-ROSというマイコン向けのROS 2ソフトウェアを活用し、 Pi:Co Classic3とPC間でのデータ通信を実現しています。

本章の前半ではArduino IDEのインストール手順を説明し、後半ではROS 2を使用する場合の開発環境の構築手順を説明します。ROS 2を使用する場合はLinux OSの一つである Ubuntuで開発することを推奨します。

## Arduino IDEのインストール

2023年4月時点で最新版のArduino IDE 2.1.0をインストールする手順を示します。Arduino IDEが更新された場合は、本書内のバージョン番号を適宜読み替えてください。

webブラウザで<u>https://www.arduino.cc</u>にアクセスしSOFTWAREのタブをクリックします。



<u>https://www.arduino.cc</u> のトップページ



使用するOSに合うソフトウェアをDOWNLOAD OPTIONSからクリックします。本書で は、Windows MSI installer、macOS Intel, 10.14: "Mojave" or newer, 64 bits、およびLinux AppImage 64 bits (X86-64)を選択した前提でインストール手順を説明します。



## Arduino IDE 2.1.0

The new major release of the Arduino IDE is faster and even more powerful! In addition to a more modern editor and a more responsive interface it features autocompletion, code navigation, and even a live debugger.

For more details, please refer to the **Arduino IDE 2.0** documentation.

Nightly builds with the latest bugfixes are available through the section below.

SOURCE CODE

The Arduino IDE 2.0 is open source and its source code is hosted on **GitHub**.

#### DOWNLOAD OPTIONS

Windows Win 10 and newer, 64 bits Windows MSI installer Windows ZIP file

Linux AppImage 64 bits (X86-64) Linux ZIP file 64 bits (X86-64)

macOS Intel, 10.14: "Mojave" or newer, 64 bits macOS Apple Silicon, 11: "Big Sur" or newer, 64 bits

Release Notes

Arduino IDE 選択ページ

以下のSupport the Arduino IDEページが表示された場合は、寄付をしてソフトウェアをダウンロードする"CONTRIBUTE & DOWNLOAD"と、ダウンロードのみを実施する"JUST DOWNLOAD"の**どちらかを選択します**。

Support the Arduino IDE   Arduin ×	+			× هم رک	- • ×
PROFESSIONAL EDUCATION	STORE		Q Search on Arduine	•.cc	SIGN IN
<b>⊖</b> → Hardwa	RE SOFTWARE CLOUD	DOCUMENTATION -	COMMUNITY <del>-</del> Blog	ABOUT	
	Suppor Since the release 1.x re been downloaded 6 (develo \$3 \$5 \$10 JUST DOWN	t the Arduino ease in March 2015, the ,620,025 times — Impr prent with a donation \$25 \$5( LOAD CONTRI	IDE e Arduino IDE has essive! Help its 0 Other BUTE & DOWNLOAD		⑦ Help

Arduino IDE ダウンロード方法選択ページ



### Windowsの場合

ダウンロードしたMSI installer (例: arduino-ide\_2.1.0\_Windows\_64bit\_msi) をダブルク リックしてインストールします。インストールが終わると下記の画面が消えます。

Arduino	IDE	
<u>.</u>	Please wait while Windows configures Arduino IDE	
		Cancel

インストール中の画面

Arduino IDEが正常にインストールできたことを確認するためArduino IDEを起動します。 Windowsのスタートボタン->検索ボックスに"arduino"というキーワードを入力すると Arduino IDEが検索結果として表示されます。

ቻምット ታベて アプリ ドキュメント	ウェブ 設定 人 フォルダー 🕨 🥋 … 🌔
最も一致する検索結果	
Arduino IDE アプリ	
アプリ	Arduino IDE アプリ

検索ボックスに"arduino"を入力した状態



表示されたArduino IDEをクリックし、Arduino IDEが起動すると以下のような画面が表示 されます。以上でインストール完了です。

💿 ske	tch_nov3a   A	Arduino IDE 2.0.1	_		×
File Ed	dit Sketch	Tools Help			
	<b>→</b> 🔛	Select Board		$\checkmark$	·Q··
Ph	sketch_no	ov3a.ino			
	1	void setup() {			
包		// put your setup code here, to run once:			
		}			
山	5	void loop() {			
		// put your main code here, to run repeatedly:			
~⇒		3			
		,			
	Output				≣ 6
	Downlo	pading packages			
	arduin	10:avr-Bccm/.5.8-atmei3.5.1-arduino/			
		Ln 1, Col 1 UTF-8 × No	board selec	ted 🗘	

起動したArduino IDE

Windowsセキュリティの重要な警告メッセージが表示されたときは、アクセスを許可し てください。アクセスをブロックしてしまうとArduino IDEのバージョンアップや追加ライ ブラリのダウンロードができなくなります。

im Windows ชระบุร	ティの重要な警告		×
このアプ! ています	リの機能のいく	つかが Windows Defender ファイアウォールでブロックされ	
すべてのパブリック ネッ の機能のいくつかがブロ	トワークとプライベー Iックされています。	トネットワークで、Windows Defender ファイアウォールにより Arduino IDE	
	名前(N):	Arduino IDE	
	発行元(P):	Arduino SA	
	パス(H):	C:¥users¥maaok¥appdata¥local¥programs¥arduino-ide¥arduino ide.exe	
Arduino IDE にこれらの	のネットワーク上で(	D通信を許可する:	
□ プライベート ネッ	›Իワ−ク (ホ−ム ネ	ットワークや社内ネットワークなど)(R)	
✓ パブリック ネット (このようなネット)	、ワーク(空港、 喫? トワークは多くの場	茶店など) (非推奨)(U) 合、 セキュリティが低いかセキュリティが設定されていません)	
アプリにファイアウォールの	の経由を許可する	ことの危険性の詳細	
		♥ アクセスを許可する(A) キャンセル	

Windowsセキュリティの重要な警告メッセージ



### macOSの場合

ダウンロードしたファイル(例:arduino-ide\_2.1.0\_macOS\_64bit.dmg)をダブルクリッ クすると以下の画面が表示されます。画面左にあるArduino IDEのアイコンを右のフォルダ に移動するとインストールが完了します。インストール後のArduino IDEはLaunchpadから 起動できます。



dmgファイルを開いた状態



## Linux(Ubuntu)の場合

ダウンロードしたファイル(例:arduino-ide\_2.1.0\_Linux\_64bit.AppImage)を右クリッ クし、メニューからプロパティを選択します。プロパティ内のアクセス権タブを開き、"プ ログラムとして実行可能"にチェックを入れます。

タウ	ンロード ×	
Ç		
	開く	Return
arduino-ide_2. 64bit.App	別のアプリケーションで開く( <u>A</u> )	
	切り取り( <u>T</u> )	Ctrl+X
	コピー( <u>C</u> )	Ctrl+C
	指定先へ移動	
	指定先にコピー	
	ゴミ箱へ移動する(⊻)	Delete
	名前を変更( <u>M</u> )	F2
	压縮( <u>O</u> )	
	星を付ける	
	プロパティ ( <u>R</u> )	Ctrl+I

AppImageファイルを右クリックした状態

arduino-ide_264bit.AppImage のプロパティ 🛛 🗙				
基本	アクセス権			
所有者(O)	自分			
アクセス	読み書き ~			
グループ(G)	ubuntu ~			
アクセス	読み書き ~			
その他				
アクセス	読み取り専用 〜			
実行	✔ プログラムとして実行可能(E)			
セキュリティコンテキスト	不明			

プロパティのアクセス権タブを開いた状態



その後、AppImageファイルをダブルクリックします。Terms of Serviceが表示された場合 は、中身を確認し"Disagree/Agree" を選択してください。以上でインストール完了です。



Terms of Service画面

AppImageファイルをダブルクリックしてもArduino IDEが実行されない場合は、 AppImageファイルのコンテンツを展開し実行するツール(FUSE)がインストールされて いない可能性があります。<u>https://github.com/AppImage/AppImageKit/wiki/FUSE</u>参照し、 FUSEをインストールしてください。



## Arduino IDEの初期設定

ここではArduino IDEの言語や外観を設定します。また、本製品に搭載されている ESP32-S3用のスケッチファイルをビルドするために、ライブラリやコンパイラを追加しま す。

## 言語設定

Arduino IDEの言語を日本語に設定します。メニューバーにあるFile->Preferences...を選択します。

💿 sketch_apr21a   Ar	🥯 sketch_apr21a   Arduino IDE 2.1.0						
File Edit Sketch To	ools Help						
New Sketch	Ctrl+N		Module 🗸				
New Cloud Sketch	Alt+Ctrl+N						
Open	Ctrl+O						
Open Recent		►					
Sketchbook		►	setup code here, to run once:				
Examples		►					
Close	Ctrl+W						
Save	Ctrl+S		main code here to nun nenestedly				
Save As	Ctrl+Shift+S		main code here, to run repeatedly				
Preferences	Ctrl+カンマ						
Advanced		►					
Quit	Ctrl+Q						

メニューバーのFile->Preferences...を選択した状態



Languageを"English"から"日本語"に変更して右下にある "OK" をクリックします。

🔤 sketch_apr26a   Arduino IDE 2.1.0	-		$\times$
File Edit Sketch Tools Help			
Contraction Contra		٦,	.@
Preferences		×	
Settings Network			
Sketchbook location:			
c:\Users\maaok\OneDrive\Documents\Arduino	BROW	/SE	
Show files inside Sketches			
Editor font size: 14			
Interface scale: ✓ Automatic 100 %			
Theme: Dark ~			
Language: 日本語 (Reload required)			
Show verbose output during 🛛 Compile 🗖 upload			
Compiler warnings None V			
✓ Verify code after upload			
✓ Auto save			
Additional boards manager URI s:		-0	
CANCEL		ok	
Lin 1 Col 1 - FSP32S3 Day Mod	ule (not co	nnectedl	$\cap$
		miceteuj	

Languageを日本語に設定した状態

"OK"をクリックしたら自動的にArduino IDEが再起動します。再起動後、メニューバーの 左端にあるボタンが"File"から"ファイル"になっていれば言語設定完了です。



### 外観設定

OSの設定によっては、Arduino IDEの外観が暗い色調で表示されます。コントラストの弱いPC画面では明るい色調に設定するとスケッチ編集画面のカーソルをはっきり表示できる場合があります。下記手順に従ってお好みの外観を設定してください。

メニューバーのファイル->基本設定…を選択します。"配色テーマ"を"Dark"から"Light"に変更し、"OK"をクリックします。これによりArduino IDEの外観が明るい色調に変更されます。以降のページで登場するArduino IDE画面は"Light"テーマを設定したものです。

🔤 sketch_apr26a   Arduino IDE 2.1.0		- 0 ×
ファイル(F) 編集 スケッチ ツール ヘルプ		
ESP32S3 Dev Mod	ule 👻	<u>, ۸o</u>
<b>国</b> 基本設定		×
£	設定 ネットワーク	
スケッチブックの場所:		
c:\Users\maaok\OneDrive\Docu	ments\Arduino	参照
- ロスケッチ内のファイルを表示		_
エディターのフォントサイズ:	14	
インターフェイスのスケール:	☑ 自動 100 %	
配色テーマ:	Light ~	
エディター言語:	日本語	
より詳細な情報を表示する	□ コンパイル □ 書き込み	
コンパイラの警告	なし 、	
✔ 書き込み後にコードを検証す	3	
✓ 自動保存(U)	7 1	
<ul> <li>□ エテイターのクイックワシエ</li> <li>追加のボードマネージャのUPL</li> </ul>		
Œ		(キャンセル) OK
🗘 indexing: 1/48		行 1、列 1 ESP32S3 Dev Module [未接続]  🗘

配色テーマでLightを選択した状態



#### ESP32ボード情報の追加

スケッチを本製品に書き込めるように、ESP32のボード情報をArduino IDEに追加しま す。メニューバーのファイル->基本設定…を選択します。画面下部の"追加のボードマネー ジャのURL"に

https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package\_esp32\_ind ex.json

を追加し"OK"をクリックします。

URLを追加したら、ESP32ライブラリやツールをダウンロードしてインストールします。メ ニューバーのツール->ボード->ボードマネージャ...を選択します。

应 sket	ch_apr26b	Arduir	no IDE 2.1.0				
ファイル(F	う 編集 ス	ケッチ	ツール ヘルプ				
		オ	自動整形	Ctrl+T			
			スケッチをアーカイブする				
	sketch_a	pr26b.	ライブラリを管理	Ctrl+Shift+I			
	1	void	シリアルモニタ	Ctrl+Shift+M			
ē.	2		シリアルプロッタ				
	4	}					
mb	5						
ШИ	6	void	SSLルート証明書を書き込み				
	7		ボード	Þ	ボードマネージャ	Ctrl+Shift+B	
÷>	° 9	2	ポート	•			
	10	í	ボード情報を取得		Arduino AVR Boards		
Q					esp32		<u> </u>
			ノートローダを書き込む		Seeed SAMD Boards		
					STM32 MCU based boards		

ボードマネージャ…を選択した状態

左側にあるボードマネージャの検索欄に"esp32"と入力すると"esp32 by Espressif Systems"が 表示されます。表示されているバージョンが**2.0.2**以上であることを確認してください。この後 インストールするmicro-ROS for Arduinoライブラリがバージョン2.0.2に対応しているためで す。

バージョンが2.0.2以上であることを確認したら"インストール"をクリックします。2023年4月 時点では2.0.8が最新です。



ボードマネージャ画面の検索欄に"esp32"を入力した状態



Arduino IDEはESP32以外のマイコンにも対応しているため、スケッチを書き込む際はESP32 のボードを選択します。メニューバーのツール->ボード->esp32->ESP32S3 Dev Moduleを選 択します。"ボード"と"ポート"が同じような文字で上下にあるため間違えないように選択し てください。

💿 sket	ch_apr26b	Arduin	o IDE 2.1.0				ESP32S3 Dev Module
ファイル(F	) 編集 フ	くケッチ	ツール ヘルプ				ESP32C3 Dev Module
		) 7	自動整形				ESP32S2 Dev Module
			スケッチをアーカイブする				ESP32 Dev Module
	sketch_a	pr26b.	ライブラリを管理	Ctrl+Shift+I			ESP32-WROOM-DA Module
	1	void	シリアルモニタ	Ctrl+Shift+M			ESP32 Wrover Module
<b>1</b>	2		シリアルプロッタ				ESP32 PICO-D4
	4	}	M//E:101 / M//E:NINAフォールウェア・フップデータ				ESP32-S3-Box
Ith	5		ccill_LL 転用またまきの み				ESP32-S3-USB-OTG
	6	void					ESP32S3 CAM LCD
Б	8	11	ボード		ボードマネージャ		ESP32S2 Native USB
*	9	}	ポート		Arduino AVR Boards	1	ESP32 Wrover Kit (all versions)
$\sim$	10		ボード情報を取得		esp32		UM TinyPICO
Q			ブートローダを書き込む		Seeed SAMD Boards	1	UM FeatherS2
					STM32 MCU based board	ls I	UM FeatherS2 Neo
							UM TinyS2
							UM RMP

ツール->ボード->esp32->ESP32S3 Dev Modulesを選択

以上で、ESP32-S3のビルド環境設定完了です。



## Arduino用のmicro-ROSのライブラリをインストール

ここでは、Arduino向けのmicro-ROSライブラリである**micro-ROS for Arduino**のインストール方法を説明します。

webブラウザで<u>https://github.com/micro-ROS/micro\_ros\_arduino/releases</u> にアクセスし、 最新のライブラリを選択します。2023年4月時点ではv2.0.5-humbleが最新です。Assetsの中 からsource code(zip)をダウンロードします。ダウンロードしたzipファイルは**展開不要で す**。

▼ Assets 2	
Bource code (zip)	Mar 23
Source code (tar.gz)	Mar 23

Source code (zip)をダウンロード

Arduino IDEを起動し、メニューバーのスケッチ->ライブラリをインクルード->".ZIP形式 のライブラリをインストール…"を選択します。

💿 sket	🥯 sketch_apr26a   Arduino IDE 2.1.0						
ファイル(F	-) 編集	スケッチ ツール ヘルプ		ライブラリを管理 Ctrl+Shift+I			
		検証・コンパイル	Ctrl+R				
		書き込み	Ctrl+U	.ZIP形式のフイノフリをインストール			
	sketch	構成と書き込み		Arduinoライブラリ			
	1	書き込み装置を使って書き込む	Ctrl+Shift+U	Arduino_BuiltIn			
17_1	2	コンパイル済みバイナリをエクスポート	Alt+Ctrl+S	ArduinoOTA			
	4	デバッグに最適化		BluetoothSerial			
M	5	スケッチフォルダを表示	Alt+Ctrl+K	DNSServer			
	5	ライブラリをインクルード	•	EEPROM			
Б	8	ファイルを追加		ESP Insights			
æ	9	ノノ T/V ととが用		ESP RainMaker			
	10			ESP32			
Q				ESP32 Async UDP			
				ESP32 BLE Arduino			

スケッチ->ライブラリをインクルード->.ZIP形式のライブラリをインストール...を選択



先ほどダウンロードしたファイルを選択し、"開く"をクリックします。

💿 追加したいライブラリの入っ	たZIPファイルを選択				×
$\leftarrow \rightarrow \checkmark \uparrow$	<u>↓</u> > ダウンロ−ド >		~ (	ダウンロードの検索	م
整理 ▼ 新しいフォルダ-					
		名前		│ 更新日時	種類
🛄 デスクトップ	*	✓ 今日			
<u>↓</u> ダウンロード	*	📒 micro_ros_arduino-2.0.5-humble.zip		2023/04/26 9:41	圧縮 (zi
■ 1 <sup>6</sup> ナー パンパ	•				
ファイル	v名(N): micro_ros_arc	luino-2.0.5-humble.zip		✓ ライブラリ (*.zip)	~
				開<(O)	キャンセル

ダウンロードしたzipファイルを選択

インストールしたライブラリはESP32には対応していますがESP32-S3のビルド環境に対応していません。そのため、ライブラリ内のファイルを加工して対応します。

先ほどインストールしたライブラリは"スケッチブックの場所"

¥libraries¥micro\_ros\_arduino"にあります。スケッチブックの場所はメニューバーのファイル->基本設定…で確認できます。

"スケッチブックの場所"¥libraries¥micro\_ros\_arduino¥srcにある"esp32"フォルダをコピーし、フォルダ名を"esp32s3"に変更します。"src"フォルダ内に"esp32"と"esp32s3"の2つのフォルダがあれば作業完了です。

ー・PC > ドキュメント > Arduino > libraries > micro_ros_arduino > src · C	♀ srcの検索
名前	更新日時
cortex-m3	2023/05/10 12:59
Cortex-m4	2023/05/10 12:59
Cortex-m7	2023/05/10 12:59
aliagnostic_msgs	2023/05/10 12:59
esp32	2023/05/10 12:59
Esp32s3 esp32フォルタをコピー	2023/05/10 13:02
example_interfaces	2023/05/10 12:59
eometry_msgs	2023/05/10 12:59

esp32フォルダをesp32s3としてコピー



## サンプルスケッチのビルドと書き込み

webブラウザで<u>https://github.com/rt-net/pico\_micro\_ros\_arduino\_examples</u> にアクセス し、micro-ROS用のPi:Co Classic3のサンプルスケッチファイルをダウンロードします。 ページ内の"Code"から"Download ZIP"をクリックすることでダウンロードできます。ダウ

ンロードしたzipファイルは**展開してください**。

A rt-net / pico_micro_ros_ex	amples Private				• Watch 2
<> Code 💿 Issues ীী Pull requ	iests 🕞 Actions	Projects	🕮 Wiki 🤇	③ Security	🗠 Insights
ਿ main 👻 ਿ 1 branch 💿 0 tag	gs		Go to file	Add file <del>-</del>	<> Code -
ShotaAk Add ci.yaml to build sket	tch files on Github Wor	Lo	ocal	Code	espaces
.github/workflows	Add ci.yaml to build s	▶ Clone			?
micro_ros_arduino-2.0.5-foxy	Add examples	HTTPS	SSH GitHub	CLI	
micro_ros_arduino-2.0.5-hum	Add examples	https://g	jithub.com/rt-	net/pico_micro	o_ros
uROS_STEP10_twistMsg	Add examples	Use Git or che	eckout with SVN u	ising the web URL	
uROS_STEP11_tfMsg	Add examples	🕁 Open wi	ith GitHub Desl	ktop	
uROS_STEP12_SensorMsg	Add examples				
uROS_STEP13_micromouse	Add ci.yaml to build s	Downloa	ad ZIP		
uROS_STEP1_LED	Add examples				2 weeks ago

Code->Download ZIPを選択してサンプルスケッチをダウンロード

ここではPi:Co Classic3のLEDを点灯させるサンプルスケッチ(uROS\_STEP1\_LED)を 例にとってスケッチのビルドと書き込み方法を説明します。

本製品にはUSBを使ってスケッチを書き込みます。デフォルトではUSBからの書き込みが できないため、書き込みができるようにArduino IDEのオプションを2箇所変更します。



まず、メニューバーのツール->ボード->esp32->ESP32S3 Dev Moduleを選択します。次 に、メニューバーのツール->USB CDC On Bootを開き、"Enabled"にします。

io IDE	2.1.0			
ツール	· ヘルプ		_	
	自動整形	Ctrl+T		
	スケッチをアーカイブする			
	ライブラリを管理	Ctrl+Shift+I		
	シリアルモニタ	Ctrl+Shift+M		
	シリアルプロッタ			
	WiFi101 / WiFiNINAファームウェア・アップデータ			
	SSLルート証明書を書き込み			
	ボード: "ESP32S3 Dev Module"	۲		
	ポ−ト	►		
	ボード情報を取得			
	USB CDC On Boot: "Enabled"			Disabled
	CPU Frequency: "240MHz (WiFi)"	۲	~	Enabled
	Core Debug Level: "None"	•		
	USB DFU On Boot: "Disabled"	Þ		

USB CDC On BootをEnabledに設定

メニューバーのツール->USB Modeを開き、"Hardware CDC and JTAG"にします。

ノール	ヘルプ			
	自動整形	Ctrl+T		
	スケッチをアーカイブする			
	ライブラリを管理	Ctrl+Shift+I		
	シリアルモニタ	Ctrl+Shift+M		
	シリアルプロッタ			
	WiFi101 / WiFiNINAファームウェア・アップデータ			
	SSLルート証明書を書き込み			
	ボード, "ECD2252 Day Madula"			
	ポート. ESFSZSS Dev Module			
	ボード			
	USB CDC On Boot: "Enabled"			
	CPU Frequency: "240MHz (WiFi)"			
	Core Debug Level: "None"			
	USB DFU On Boot: "Disabled"			
	Erase All Flash Before Sketch Upload: "Disabled"			
	Events Run On: "Core 1"			
	Flash Mode: "QIO 80MHz"			
	Flash Size: "4MB (32Mb)"			
	JTAG Adapter: "Disabled"			
	Arduino Runs On: "Core 1"			
	USB Firmware MSC On Boot: "Disabled"			
	Partition Scheme: "Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS)"			
	PSRAM: "Disabled"			
	Upload Mode: "UART0 / Hardware CDC"			
	Upload Speed: "921600"			
	USB Mode: "Hardware CDC and JTAG"		Hardware CDC and JTAG	
	書き込み装置		USB-OTG (TinyUSB)	1
	ゴートローダたまさい			

USB ModeをHardware CDC and JTAGに設定



ダウンロードしたサンプルスケッチ内のuROS\_STEP1/uROS\_STEP1.inoファイルをダブ ルクリックで開きます。または、Arduino IDEを起動し、メニューバーのファイル->開く…を クリックしてuROS STEP1.inoを開きます。

💿 sketch_apr26b   Ard	luino IDE 2.1.0	
ファイル(F) 編集 スケッ	チッール ヘル	Ĵ
新規スケッチ	Ctrl+N	odule 👻
新規クラウドスケッチ	Alt+Ctrl+N	
開く	Ctrl+O	
最近使った項目を開く		>
スケッチブック		<pre>etup code nere, to run once:</pre>
スケッチ例		•
閉じる	Ctrl+W	
保存	Ctrl+S	ain code hone to nun nonest
名前を付けて保存…	Ctrl+Shift+S	ain code nere, to run repeate
基本設定	Ctrl+カンマ	
詳細		•
終了	Ctrl+Q	

ファイル->開く...を選択する

スケッチファイルを開く時に以下のメッセージが出た場合は、スケッチファイル名とフォ ルダ名が一致していない、またはフォルダが無いため新しくフォルダを作って良いか、とい うことを確認しています。Arduino IDEの仕様上、メインとなる"ファイル名"と"フォルダ名" が同じでないときにこのメッセージが表示されます。ファイル名やフォルダ名を変更する場 合はそれぞれ同じ名前にしてください。

ファイ	ľ
ル"uROS_STEP1.ino"は、"uROS_	
STEP1"という名前のスケッチフォルタ	ł
の中にあることが必要です。	ł
このフォルタを作成し、ファイルを移	
動させ、継続しますか?	
OK キャンセル	

ファイル名とフォルダ名が一致しない場合の確認画面



サンプルスケッチを開くと以下のような画面が表示されます。



uROS\_STEP1\_LED.inoを開いた状態



ボードが"ESP32S3 Dev Module"になっていないときは、メニューバーのツール->ボード ->esp32->ESP32S3 dev Moduleを選択してください。

💿 sket	ch_apr26b	Arduin	o IDE 2.1.0				ESP32S3 Dev Module
ファイル(F	う 編集 フ	くケッチ	ツール ヘルプ				ESP32C3 Dev Module
		) 7	自動整形				ESP32S2 Dev Module
			スケッチをアーカイブする				ESP32 Dev Module
	sketch_a	pr26b.	ライブラリを管理				ESP32-WROOM-DA Module
	1	void	シリアルモニタ				ESP32 Wrover Module
<b>行</b> ]	2		シリアルプロッタ				ESP32 PICO-D4
	4	}	M(F:101 / M(F:NUNAフォールウェス・マップ	₩			ESP32-S3-Box
Ith	5		wiFitut / wiFitutA/デームウェア・アック				ESP32-S3-USB-OTG
	6	void	SSLIV-Filling者を音さ込め				ESP32S3 CAM LCD
	8		ボード		ボードマネージャ		ESP32S2 Native USB
8	9	}	ポート		Arduino AVR Boards		ESP32 Wrover Kit (all versions)
	10		ポ−ド情報を取得		esp32		UM TinyPICO
Q			ブートローダを書き込む		Seeed SAMD Boards		UM FeatherS2
			, 10, 70, 20, 20, 20, 20, 20, 20, 20, 20, 20, 2		STM32 MCU based board	s	UM FeatherS2 Neo
							UM TinyS2
							UM RMP

ツール->ボード->esp32->ESP32S3 dev Moduleを選択

左上にあるアイコンで"検証"を実行しスケッチをビルドします。スケッチの記述に 問題がなければ以下のようなメッセージが表示されます。

出力	≣× 6
最大1310720バイトのフラッシュメモリのうち、スケッチが227649バイト(17%)を使っています。 最大327680バイトのRAMのうち、グローバル変数が13296バイト(4%)を使っていて、ローカル変数	数で314384/

検証に問題が無いときのメッセージ

スケッチの記述にエラーがあった場合は、赤文字でエラー箇所が表示されます。

出力	≣× 6
<pre>pinMode(LED4,OUTPUT);</pre>	
<pre>/Users/aokimasatake/Documents/Arduino/uROS_STEP1/uROS_STEP1.ino:12:11: note: pinMode(LED4,0UTPUT);</pre>	suggeste
exit status 1	
Compilation error: 'LED4' was not declared in this scope	

#### 検証エラー時のメッセージ

Arduino IDEの右下にあるアイコン をクリックすればメッセージを非表示にできま す。もう一度クリックすると表示されます。画面を大きくしてスケッチ全体を少しでも多く 見たい時や、このメッセージ画面が不要なとき等に活用してください。



Ubuntuの場合、ModuleNotFoundError: No module named 'serial'のメッセージが出力され たときはビルドができていません。下記のコマンドをターミナルで実行し、ビルドに必要な アプリをインストールしてください。

Unset

- \$ sudo apt-get install python-pip
- \$ pip install pyserial
- \$ pip install esptool

スケッチにエラーが無いことを確認したら、PCと本製品をUSBケーブルで接続し、本製品のモード切替スイッチをRUN1と書れていない無い方向(USBコネクタ側でない方向)にスライドし、Pi:Co Classic3の電源を入れます。Pi:Co Classic3の電源を入れた状態でPCと接続した場合は、本製品のリセットボタンを押します。



モード切替スイッチを操作し、USBケーブルを接続した状態

本製品をPCと接続した後、以下の手順でボードとポートを設定します。Pi:Co Classic3の 電源をOFFした後や、本製品のリセットボタンを押した後にも再度設定してください。 メニューバーの下に表示されているボード名(ESP32S3 Dev Module)をクリックし、一 番下の"他のボードとポートを選択…"を開きます。

••	•		uROS_S1	ΓΕΡ1	I_LED   Arduir
	$\rightarrow$	ESI	P32S3 Dev Module	•	
	uROS_STE	i ţ	AirM2M_CORE_ESP32C3 /dev/cu.usbmodem13401		
	2 / 3 /	Į Į	ESP32S3 Dev Module /dev/cu.Bluetooth-Incoming-Por	rt	ise, Version
	5 / 6 / 7 /	, ₽	不明 /dev/cu.D-SP8		.cense at ises/LICENSE-
⊳ \$	8 / 9 / 10 /	, , .	不明 /dev/cu.RN4678-6612		.aw or agreed .s distribut€
Q	11 / 12 / 13 /	Ŷ	不明 /dev/cu.wlan-debug		IS OF ANY KIN .c language <u>c</u>
	14 15 <del>4</del>	他の	Dボードとポートを選択		
	16 # 17 #	⁺defir ⁺defir	ne LED1 2 ne LED2 42		

ボード名->他のボードとポートを選択…を開く

ボードの検索欄に"dev"を入力し、"ESP32S3 Dev Module"にチェックが入っていることを 確認します。チェックが入っていない場合は、ESP32S3 Dev Moduleをクリックします。 ポート選択欄では本製品を接続したUSBポートを選択します。Windowsの場合はCOMX、 macOSの場合はusbmodemXXXXX Serial Port(USB)、Ubuntuの場合は/dev/ttyUSBXという 名称です。選択したら"OK"をクリックします。

他のボードとポートを選択			×
スケッチを書き込みたい場合には、ボードとポ ボードのみを選択した場合、コンパイルはでき	ートのī ますが、	両方を選択してください。 スケッチの書き込みはできません。	
ボード		<b>ポート</b>	
dev	Q		
ESP32C3 Dev Module ESP32S2 Dev Module ESP32S3 Dev Module Hornbill ESP32 Dev	*	/dev/cu.Bluetooth-Incoming-Port Serial Port         /dev/cu.D-SP8 Serial Port         /dev/cu.RN4678-6612 Serial Port         /dev/cu.usbmodem13401 Serial Port (USB)         /dev/cu.wlan-debug Serial Port	~
LOILDR Dev Board		□全てのポートを表示 <b>キャンセル</b> OK	

ボードとポートの設定





ポート設定後、 をクリックしてスケッチを書き込みます。下記のように表示された ら書き込み完了です。



書き込み完了時の画面表示

Ubuntu PCで書き込みに失敗する時は、書き込み前にターミナルで次のコマンドを実行し、USBポートの使用権限を変更します。

Unset

\$ sudo chmod 777 /dev/ttyACM0

永続的にUSBポートの使用権限を変更する場合は、次のコマンドを実行し、PCを再起動 します。

Unset

- # 下記を実行した後にPCを再起動すること
- \$ sudo usermod -aG dialout \$USER

スケッチの書き込み完了後、本製品のモード切替スイッチをRUN1と書かれた方向にスラ イドしてリセットを押すか、電源をOFF->ONするとPi:Co Classic3前方のLEDが0.5秒点 滅、0.5秒消灯を繰り返します。



### micro-ROSの環境設定

ここではmicro-ROSのインストール方法と、Pi:Co Classic3とPC間で通信する方法につい て説明します。PCのOSはUbuntuを推奨します。また、PCにはあらかじめROS 2 Humbleを インストールしてください。インストール方法は

<u>https://docs.ros.org/en/humble/Installation/Ubuntu-Install-Debians.html</u>を参照してください。

ROS 2 Humbleインストール後に下記のコマンドを実行し、ROS 2パッケージをビルドす るために必要なソフトウェアをインストールしてください。

Unset

- \$ sudo apt install python3-colcon-common-extensions
- \$ sudo apt install python3-rosdep2
- \$ rosdep update

#### micro-ROS-Agentのインストール

Ubuntu PCにmicro-ROS-Agentのソースコードをダウンロードしてビルドします。次のコマンドを実行します。ROS 2のワークスペースはros2\_wsとしています。

Unset

```
# ワークスペースを作成します
$ source /opt/ros/humble/setup.bash
$ mkdir -p ~/ros2_ws/src
$ cd ~/ros2_ws/src
# micro-ROSセットアップパッケージをダウンロードしてビルドします
$ git clone -b humble
https://github.com/micro-ROS/micro_ros_setup
src/micro-ros-build
$ sudo apt update && rosdep update
$ cd ~/ros2 ws
$ rosdep install --from-paths src --ignore-src -y
$ colcon build
$ source install/local_setup.bash
# micro-ROS-Agentパッケージをダウンロードします
$ ros2 run micro_ros_setup create_agent_ws.sh
# micro-ROS-Agnetパッケージをビルドしてインストールします
$ colcon build
$ source install/local_setup.bash
```

RT CORPORATION

インストールができたら次のコマンドを実行しmicro-ROS-Agentを起動します。

Unset

\$ ros2 run micro\_ros\_agent micro\_ros\_agent udp4 --port 8888

micro-ROS-Agentを起動すると次のように表示されます。終了時はCtrl+Cを入力します。

ubuntu@ubuntu-MacBookPro:~\$	source /opt/ros/	humble/setup.bash					
ubuntu@ubuntu-MacBookPro:-\$ cd ros2_ws/							
ubuntu@ubuntu-MacBookPro:~/	<pre>ros2_ws\$ source i</pre>	nstall/local_setup.bash					
ubuntu@ubuntu-MacBookPro:~/	ros2_ws\$ ros2 run	<pre>micro_ros_agent micro_ros_ag</pre>	ent udp4port 8888				
		ux.cpp   init		port: 8888			
[1676975414.202702] info		<pre>set_verbose_level</pre>		verbose_level: 4			

micro-ROS-Agentの実行画面



#### micro-ROSライブラリにカスタムメッセージをインストールする

Arduino用のmicro-ROSライブラリ(micro\_ros\_arduino)には、ロボットを操作する時に 使用するtfやtwistなどの基本的なROSメッセージが用意されています。本製品のサンプルス ケッチでもそれらのメッセージを使用しています。

ROSメッセージはユーザが自由に作成できます(これをカスタムメッセージと呼びま す)。サンプルスケッチでは、Pi:Co Classic3のセンサ値を表現するカスタムメッセージを 使用しています。micro\_ros\_arduinoはバイナリファイルで提供されているため、カスタム メッセージを追加するにはmicro ros arduinoをビルドします。

ここでは、micro\_ros\_arduinoのソースファイルをダウンロードし、カスタムメッセージ を追加してビルドする方法を説明します。追加するカスタムメッセージは <u>http://github.com/rt-net/pico\_msgs</u> に公開されています。

micro\_ros\_arduinoを作成する際にDockerを使用します。Dockerをインストールしていない場合はhttps://docs.docker.com/engine/install/ubuntu/を参照してインストールしてください。下記に示すコマンドは、先ほどのリンクにあるインストール方法の一つです。

```
Unset
$ sudo apt-get update
$ sudo apt-get install ca-certificates curl gnupg
$ sudo install -m 0755 -d /etc/apt/keyrings
$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | \
sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
$ sudo chmod a+r /etc/apt/keyrings/docker.gpg
$ echo "deb [arch="$(dpkg --print-architecture)"
signed-by=/etc/apt/keyrings/docker.gpg]
https://download.docker.com/linux/ubuntu \
"$(. /etc/os-release && echo "$VERSION_CODENAME")" stable" | \
sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
$ sudo apt-get update
$ sudo apt-get install docker-ce docker-ce-cli containerd.io
docker-buildx-plugin docker-compose-plugin
```

次にmicro\_ros\_arduinoのソースファイルをダウンロードします。

```
Unset
$ cd ~
$ git clone http://github.com/micro-ROS/micro_ros_arduino
```



ダウンロードしたmicro\_ros\_arduinoフォルダにある extras/library\_generation/extra\_packages/extra\_packages.reposにPi:Co Classic3のセンサの パッケージのURLを追加します。

```
Unset
repositories:
control_msgs:
type: git
url: <u>https://github.com/ros-controls/control_msgs</u>
version: galactic-devel
pico_msgs: # 追加
type: git # 追加
url: http://github.com/rt-net/pico_msgs # 追加
```

次にビルド用のdockerイメージでmicro\_ros\_arduinoをビルドします。オプションなしで コマンドを実行すると、micro\_ros\_arduinoがサポートしているすべてのマイコンに対して ライブラリをビルドするため時間がかかります。ここでは、ESP32のみビルドするようにオ プション(-p esp32)をつけます。

Unset

```
$ cd ~/micro_ros_arduino
$ docker pull microros/micro_ros_static_library_builder:humble
$ docker run -it --rm -v $(pwd):/project --env \
MICROROS_LIBRARY_FOLDER=extras \
microros/micro_ros_static_library_builder:humble -p esp32
```

ビルドが成功するとmicro\_ros\_arduino/srcにpico\_msgsフォルダができます。この pico\_msgsフォルダとesp32/libmicroros.aファイルをArduino IDEの環境にコピーします。 pico\_msgsフォルダは、Arduinoライブラリがある

libraries/micro\_ros/ros\_arduino-2.0.5-humble/srcにコピーし、 libmicroros.aはlibraries/micro\_ros/ros\_arduino-2.0.5-humble/src/esp32s3にコピーします。

Unset

```
# ~/Arduino/librariesにライブラリがある場合
$ cd ~/micro_ros_arduino/src
$ cp -r pico_msgs
~/Arduino/libraries/micro_ros_arduino-2.0.5-humble/src
$ cp esp32/libmicroros.a
```

~/Arduino/libraries/micro\_ros\_arduino-2.0.5-humble/src/esp32s3

以上でカスタムメッセージのインストール完了です。

### PCのROS 2環境にカスタムメッセージをインストールする

Pi:Co Classic3が送信するカスタムメッセージを受け取るために、PCのROS 2環境ヘカス タムメッセージをインストールします。次のコマンドを実行してください。

Unset

```
$ cd ~/ros2_ws/src
```

```
$ git clone https://github.com/rt-net/pico_msgs.git
```

\$ cd ~/ros2\_ws

```
$ rosdep install --from-paths src --ignore-src -y
```

- \$ colcon build
- \$ source install/local\_setup.bash

以上でインストール完了です。



### PCとPi:Co Classic3間の通信確認

ここでは、micro\_ros\_arduinoに追加したカスタムメッセージを使用し、Pi:Co Classic3からPCへメッセージを送信できることを確認します。

#### Pi:Co Classic3側

カスタムメッセージを使ったサンプルスケッチ(uROS\_STEP11\_SensorMsg)を使用し ます。PCとPi:Co Classic3との間で通信を行うためWi-Fiの設定が必要です。まず、PCを Wi-Fiに接続してPCのIPアドレスを取得します(例:192.168.1.89)。その後、スケッチの 下記の行を使用するWi-FiのAP(アクセスポイント)名、パスワード、取得したPCのIPアド レス、ポート番号(デフォルトは8888)に書き換えてください。また、注意事項として、 PCとPi:Co Classic3は、同じWi-FiのAPに接続してください。同じWi-FiのAPに接続しない と通信できません。

C/C++

```
set_microros_wifi_transports("使用するWiFiのAP名", "Wi-Fiのパス
ワード", "PCのIPアドレス", 8888);
```

変更したらスケッチをビルドして書き込みます。

#### PC側

ターミナルを2つ起動し、以下のコマンドを実行します。 1つ目のターミナルではmicro-ROS-Agentを起動します。

Unset

```
$ source ~/ros2_ws/install/setup.bash
```

\$ ros2 run micro\_ros\_agent micro\_ros\_agent udp4 --port 8888

2つ目のターミナルではrqt\_plotを起動します。

Unset

```
$ source ~/ros2_ws/install/setup.bash
```

```
$ ros2 run rqt_plot rqt_plot topics /pico_sensor/forward_r
```

```
/pico_sensor/forward_l /pico_sensor/right /pico_sensor/left
```



通信に成功すると、rqt\_plotにPi:Co Classic3の壁センサ値が描画されます。壁センサに手をかざすと値が変化します。



rqt\_plotでPi:Co Classic3の壁センサ値を表示



## 使用しているツール、OSSのバージョン

本書で使用するツールやOSSのバージョンは以下のとおりです。

#### 本書で使用するツールやOSSのバージョン情報

ツールやOSS	バージョン	URL	
Arduino IDE	2.1.0	https://www.arduino.cc/en/software	
Arduino core for the ESP32	2.0.8	https://github.com/espressif/arduino-esp 32/releases	
micro_ros_arduino	v2.0.5_humble	https://github.com/micro-ROS/micro_ros _arduino/releases	



## 製品保証

#### 保証の内容

お客様にお買い上げいただきました本製品につき、株式会社アールティ(以下「弊社」と いいます)の設計あるいは製作上の責任にて故障や不具合が生じた場合、下記に示す保証期 間と条件により、無償で修理等します。

## 保証者の名称、所在地および電話番号

保証者は、本書の「お問い合わせ」に記載のとおりとします。なお、修理等の受付時間 は、「お問い合わせ」に記載の受付時間とします。

## 保証期間

保証期間は、本製品を納入した日から起算して12ヶ月間とし、この期間を経過した場 合、保証は終了とします。

## 保証の適用

- 1. この保証は、日本国内で販売し、使用される本製品に適用されます。海外に設置や 移動した本製品は、この保証の対象となりません。
- 2. この保証は、本製品の本体についてのみ適用します。本製品のセットを構成する付 属品については、この保証の対象外とします。
- 3. 本体の故障または不具合により生じた本製品の本体以外の故障、不具合、破損、滅 失、損害(人的・物的損害、間接損害、特別損害、逸失利益等)については、本保 証の対象外とします。
- 4. この保証は、標準仕様の製品に適用されます。特殊仕様および特記事項を含む特注 仕様の内容は保証範囲外とします。



### 保証の除外事項

次の何れかの事項に該当する場合、保証は適用されません。

- 1. 本製品のマニュアル、装置添付ラベル、取扱説明書群(以下「マニュアル等」とい います)が定める手順、注意事項、安全事項、確認事項、動作方法等を順守しな かったことによる故障または不具合
- マニュアル等に記載された稼働環境条件以外の条件のもとで稼働させたことに起因 する故障または不具合
- マニュアル等に記載された仕様(可搬重量、動作速度等)の限度・範囲を超える使用(お客様によるプログラムの改変、本体の改造等を含みます)に起因する故障または不具合
- 4. 経時変化による劣化・故障・不具合
- 5. 天災地変による故障または不具合
- 6. 結露、異常電圧、衝突、転倒、落下、公害等の事故による故障または不具合
- 7. 弊社または弊社が指定する業者以外による修理・整備に起因する故障または不具合
- 8. 前各号のほか、弊社の責に帰すことのできない事由により生じた故障または不具合

## 保証の態様

保証期間中に、マニュアル等に従い正常な使用状態で本製品が故障し、または不具合を起こし、お客様より弊社にその旨ご連絡をいただいた場合、弊社は、弊社の判断により、以下の何れかの措置を講じます。

(ア) 無償修理

お客様よりお送りいただいた現品につき修理、部品の交換等を行い、正常な状態に回復し ます。修理は、弊社または弊社が委託した業者が行うものとします。部品等にかかる費用 は、弊社の負担とします。なお、修理のために交換された部品または本体の一部について は、お返しできない場合がありますのでご了承ください。

(イ)本製品の無償交換

お客様よりお送りいただいた現品につき修理不能と弊社が判断した場合、同等の製品と交換します。なお、この場合、現品の全部または一部をお返しできない場合がありますのでご 了承ください。同等の製品をお客様へお送りする際の送料は、弊社の負担とします。

(ウ) 返金

弊社は、(ア)項に定める無償修理および(イ)項に定める無償交換に応じることができ ないと判断したとき、本製品の購入価格を上限として返金します。なお、返金の際の銀行振 込等の手数料は、弊社の負担とします。

TRT CORPORATION

## お客様の費用負担

次に掲げる費用は、お客様の負担とします。

- 前条(ア)項に定める無償修理について、修理前の現品をお客様から弊社へお送り いただく際の梱包費用および送料。なお、お客様は、輸送に耐え得る梱包方法にて 梱包するものとします。
- 前条(イ)項に定める無償交換について、交換前の現品をお客様が弊社へお送りいただく際の梱包費用および送料。なお、お客様は、輸送に耐え得る梱包方法にて梱包するものとします。
- 3. 保証の除外事項に該当することが判明した場合または故障・不具合でないことが判明した場合の修理・交換サービス料金および返送の際の送料。弊社の点検・調査により、保証の除外事項に該当することが判明した場合、弊社は、お客様にその旨お伝えし、修理等の要否について確認します。要修理等とのご回答をお客様から得た場合、弊社は、別途お客様と合意した修理・交換サービス料金にて修理等を行います。なお、本項に定める場合の現品の返送にかかる送料は、お客様負担とします。

### 保証を受けるための手続き

弊社は、故障・不具合の原因の究明、修理等の解析を迅速に行うため、お客様に下記のお 手続きをお願いします。なお、修理期間は、現品到着日より約2週間とさせていただきます が、故障状況によってはさらに時間を要する場合がありますのでご了承ください。

- 1. 使用条件をできる限り詳細に明記した書面の提示
- 2. 故障状況をできる限り詳細に明記した書面の提示



## お問い合わせ

本製品に関するお問い合せは、下記窓口までお申し付けください。最新の製品情報、会社 情報等については、弊社ホームページをご覧ください。

〒101-0021 東京都千代田区外神田3-9-2 末広ビル3F 株式会社アールティ

URL https://rt-net.jp/

TEL 03-6666-2566

E-mail support@rt-net.jp (技術サポート)

sales@rt-net.jp (営業サポート)

受付時間 平日11:00-18:00(土日祝、夏季、年末年始はお休みです)

## 改訂履歴

発行日 (YY/MM/DD)	版数	改訂内容	編集者
23/6/26	1.0	初版	Masatake Aoki

## Copyright・知的財産権について

弊社は、本製品および本製品に関連して弊社が制作したソースファイル、ディレクトリ、 実行可能ファイル、データ、開発ツールおよびその他の資料(以下「弊社資料」といいま す)にかかる特許権、実用新案権、意匠権、著作権、ノウハウ、その他の技術および知的財 産に関する一切の権利を留保するものとします。本書は、弊社の商標、商号、役務商標、商 品名、ロゴの使用を許諾するものではありません。ただし、本製品および弊社資料の説明ま たは記述に合理的に必要な範囲において当該商標等を使用する場合は、この限りでないもの とします。なお、本製品および弊社資料に付された商品識別番号、商標、登録商標、コピー ライト、その他の注意事項は除去しないようお願いします。

All the company and product names in this document are trademarks or registered trademarks of their respective companies.

All the documents, photos, and illustrations are copyrighted and protected by the copyright law of Japan and overseas. All the contents in this document are not allowed to be uploaded to any public or local area networks such as the Internet without permission from RT Corporation.

