Spectrum Technology

V1.0 2020/10/8

抜粋版

はじめてのAWS_Greengrass開発キット

～ネット環境不要でAWS　IoTを動作させ、GPU搭載のエッジデバイスで
AI，IoTの最大活用～

設定・開発編（Jetson版）

AWS IoT Greengrass

スペクトラム・テクノロジー株式会社

https://spectrum-tech.co.jp

sales@spectrum-tech.co.jp

目次

抜粋版のためページと本文は一致しません

目次

抜粋版のためページと本文は一致しません

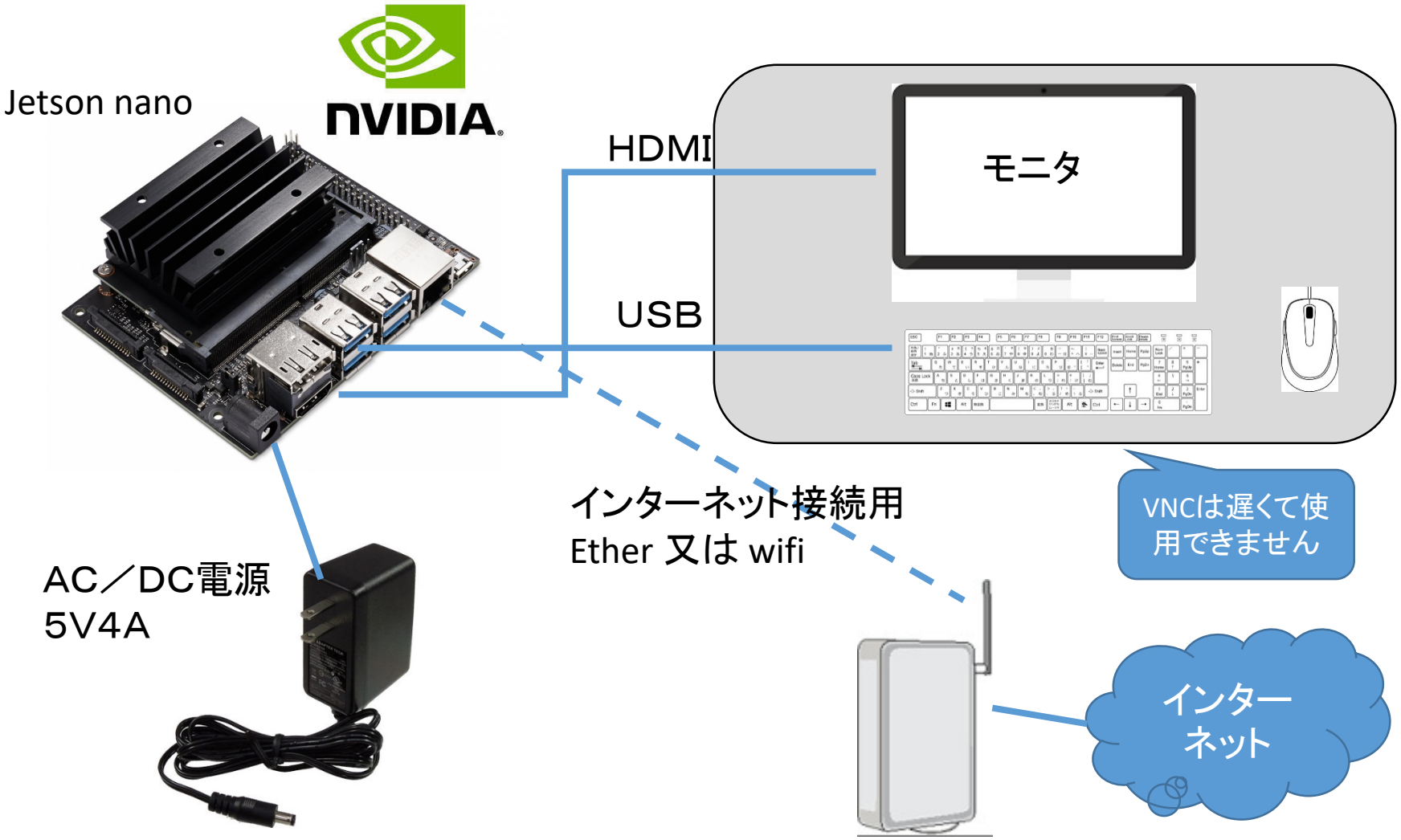
目次

ページ

抜粋版のためページと本文は一致しません

Spectrum Technology

開発キット（Jetson版） 接続構成



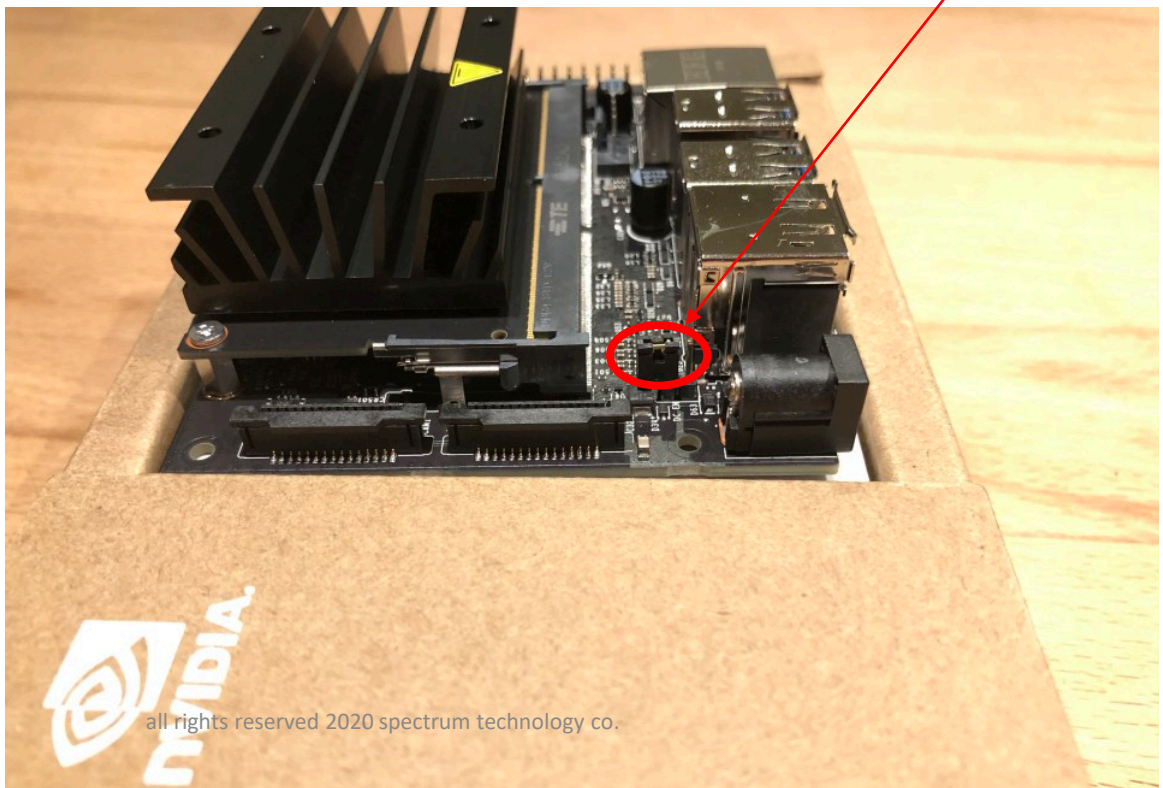
開発キット設定

1. Jetson組立

① 外部電源用ショートピン設定

- 添付の外部電源（5V4A）を使用するために ショートピンを設定します。
- USB電源の場合は、立ち上がらないことがあ りますので、USB電源は使用しないでください。

出荷時は、ショートされて いません



開発キット設定

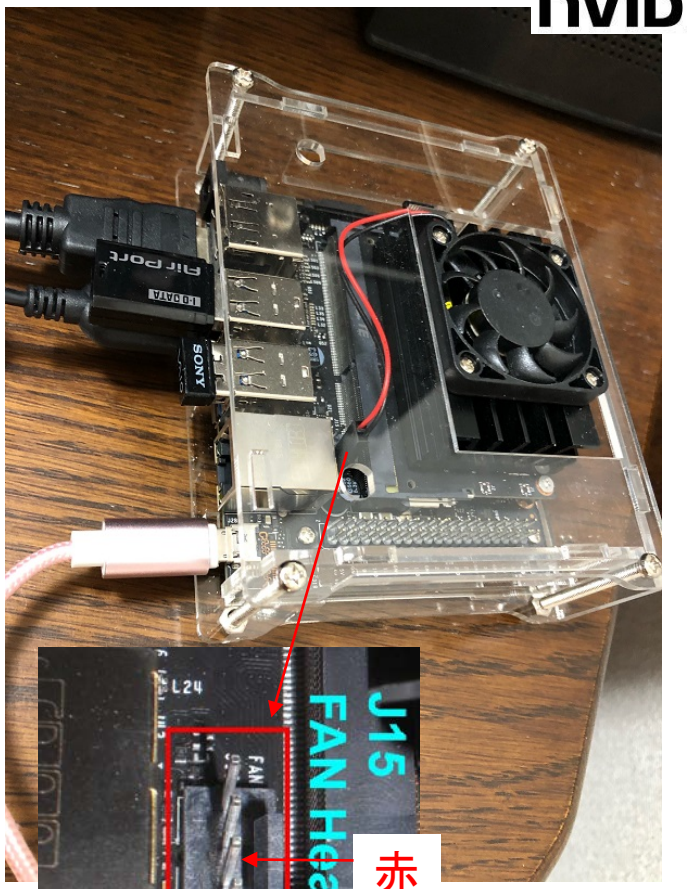
1. Jetson組立

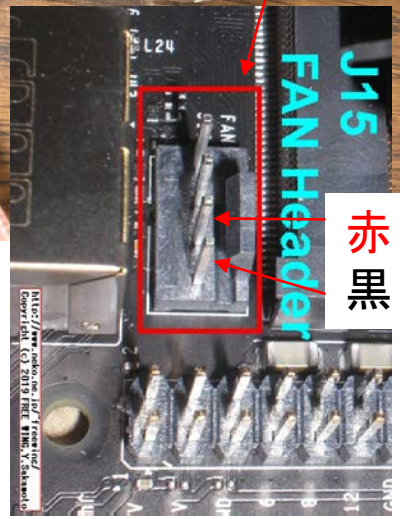
② FAN取付

- 添付のFANをネジ4本で接続
- メス側は、ラジオペンチ等ではさんで、オスのネジを挿入して、ネジ止めします。
- 電源ケーブルを左側から黒、赤で接続します。

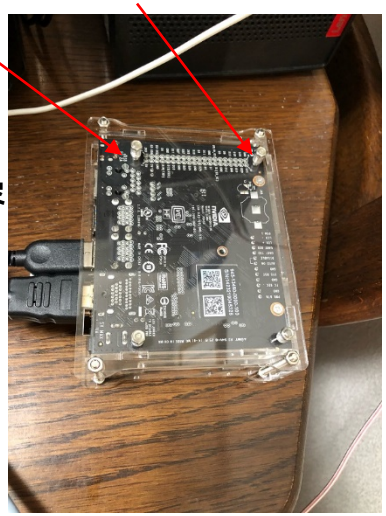
③ ケース組立

- 内側のネジ4本を先に、スペーサを入れて固定
- 内側のネジ4本が足になります。





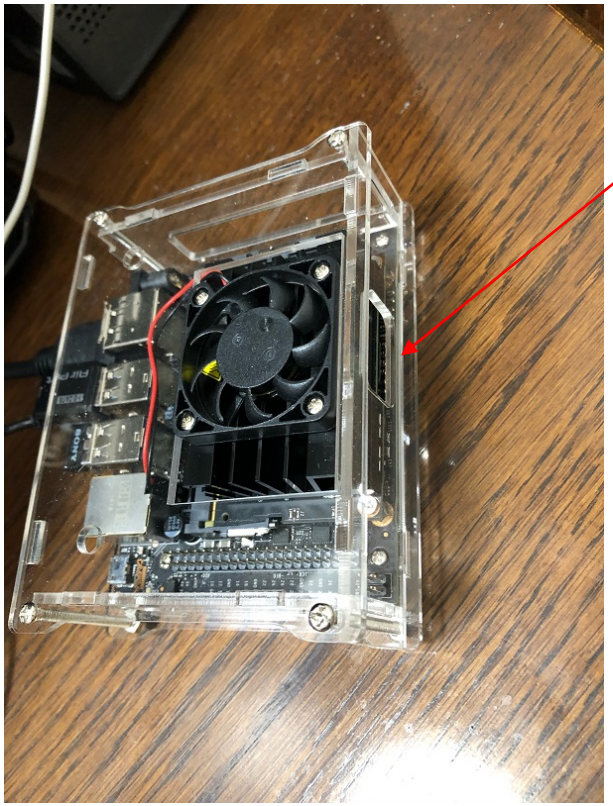
- 周りのケース4個と上のケースをはめ込みネジ止めします。
- 注意：USBなどを収容する側のケースは無理すると割れます。



開発キット設定

2.　Jetson起動

①　マイクロSDカードを挿入

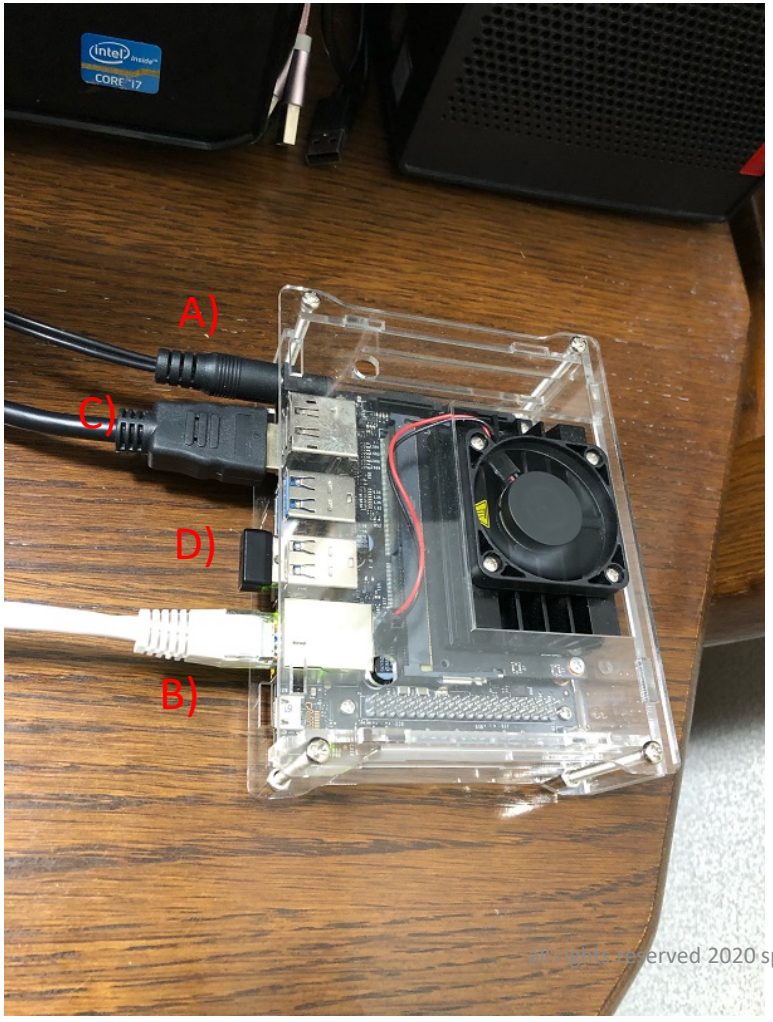


・既にマイクロSDカードは挿入済。
・取り出しは、押すと少し飛び出します。
・挿入時は、金属面を上にして差します。

開発キット設定マニュアル

2. Jetson起動

② 電源、LANケーブル、モニター、キーボード接続



ノートPCの場合は、表示されません

A) AC／DC電源
- 外部電源を接続

B) LANケーブル又はWiFiドングル接続
- 左図は、LAN接続例

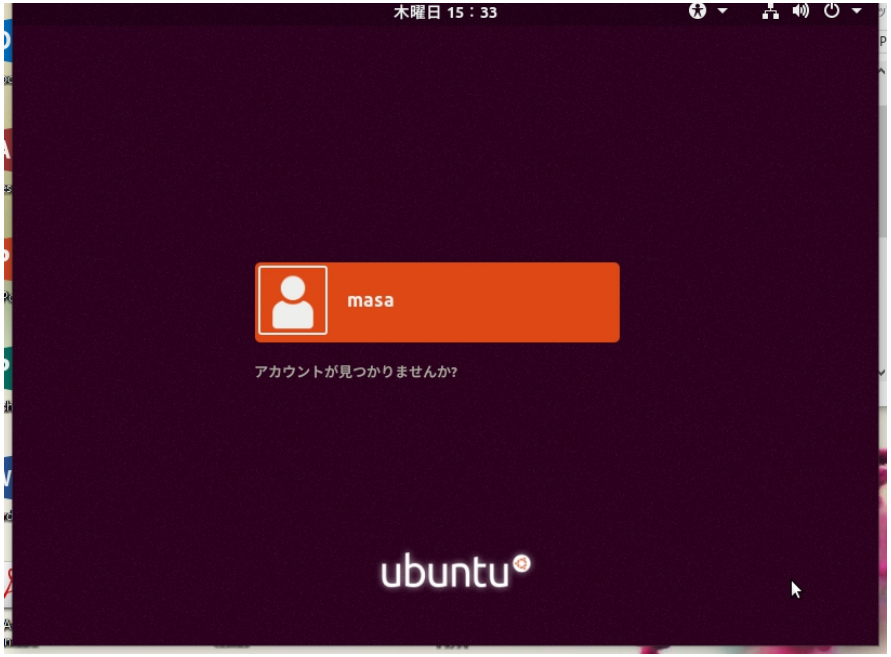
C) モニタ接続
- モニター（TV、PCでHDMI端子のあるもの）を準備します。
- HDMIケーブルにより、Jetsonとモニタを接続します。

D) マウス、キーボード接続
- マウス、キーボードがBluetoothで接続されている場合は、Bluetooth USBと接続します。

開発キット設定

3.　Jetson初期設定

①　Jeston　nano起動

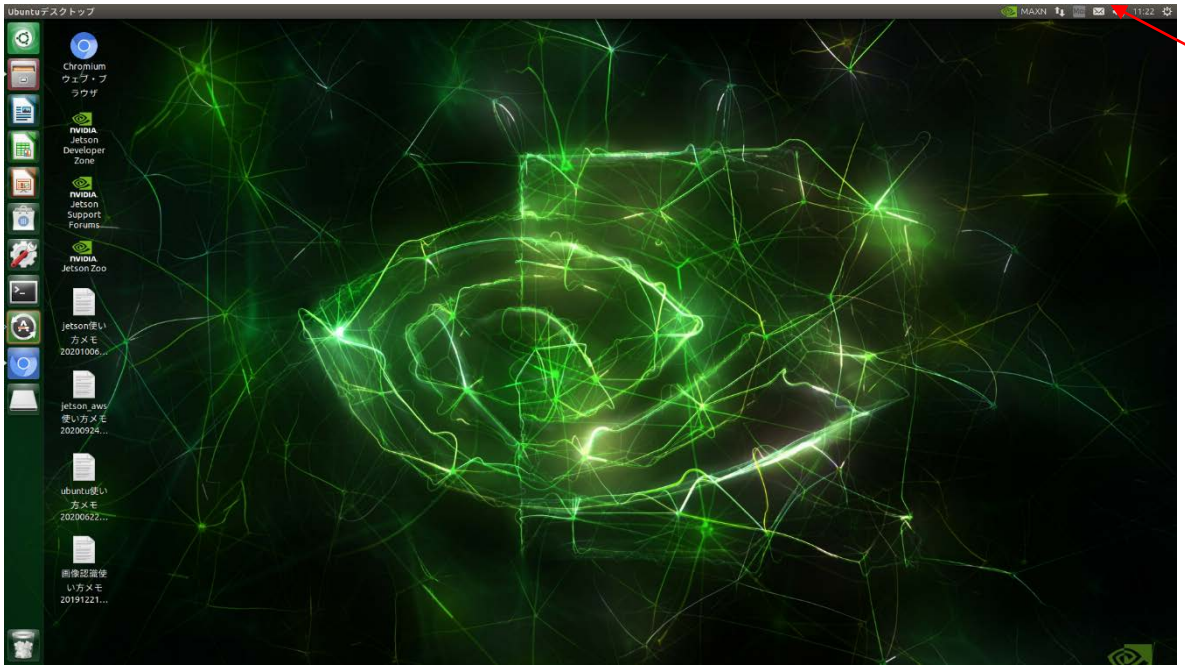


- Ubuntuの画面が立ち上がります。
- masaをクリックしてパスワードを入力
- パスワード：jetson
- 後ほど、パスワードは変更できます。
- アカウントは変更できますが、フォルダ名など変更になるためプログラムにより動作しない場合がありますので、自己責任でお願いします。
- ロック画面の場合は、マウスで画面を下にドラッグします。

開発キット設定

3. Jetson初期設定

② デスクトップ画面



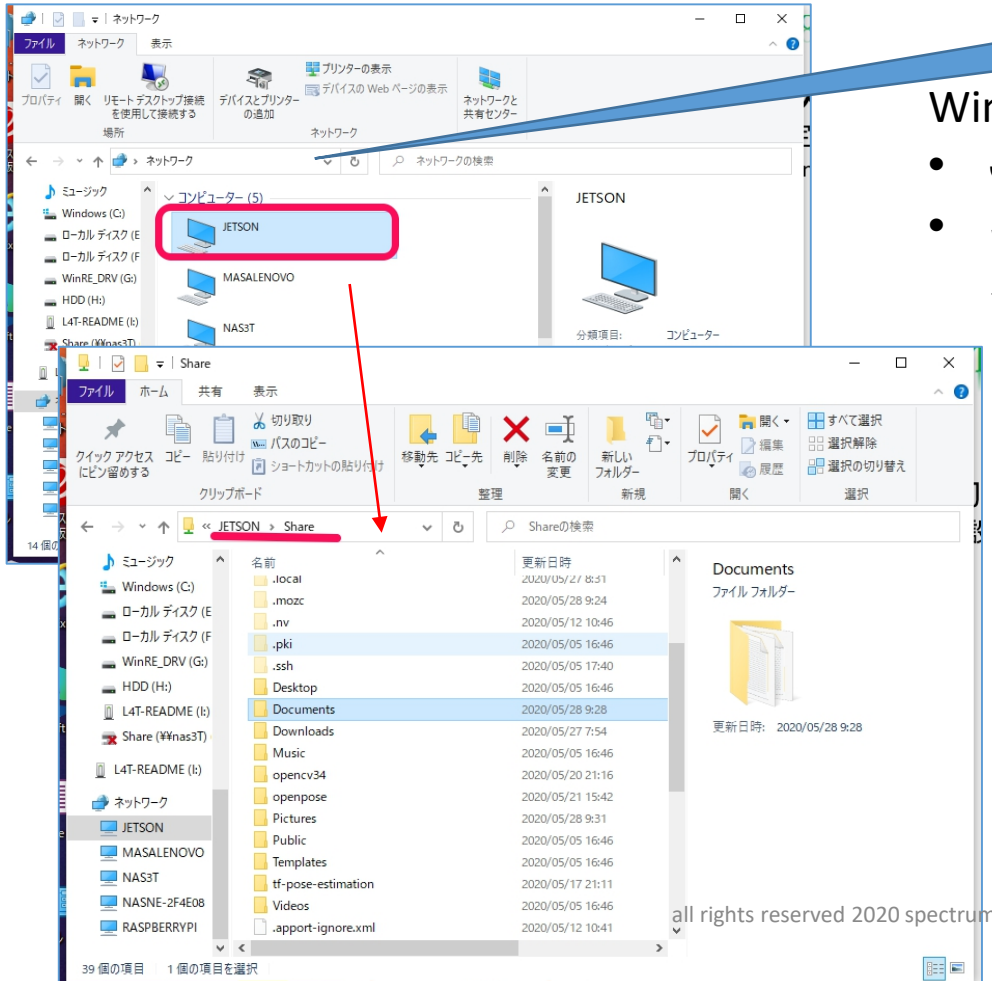
アイコン説明

- 左バー：アクティビティ　設定、ターミナルなど
- 右上：回線接続、日本語入力、スピーカ、時計、電源

開発キット設定

3. Jetson初期設定

⑤ Samba設定（Windowsから） Windowsネットワークとフォルダを共有します

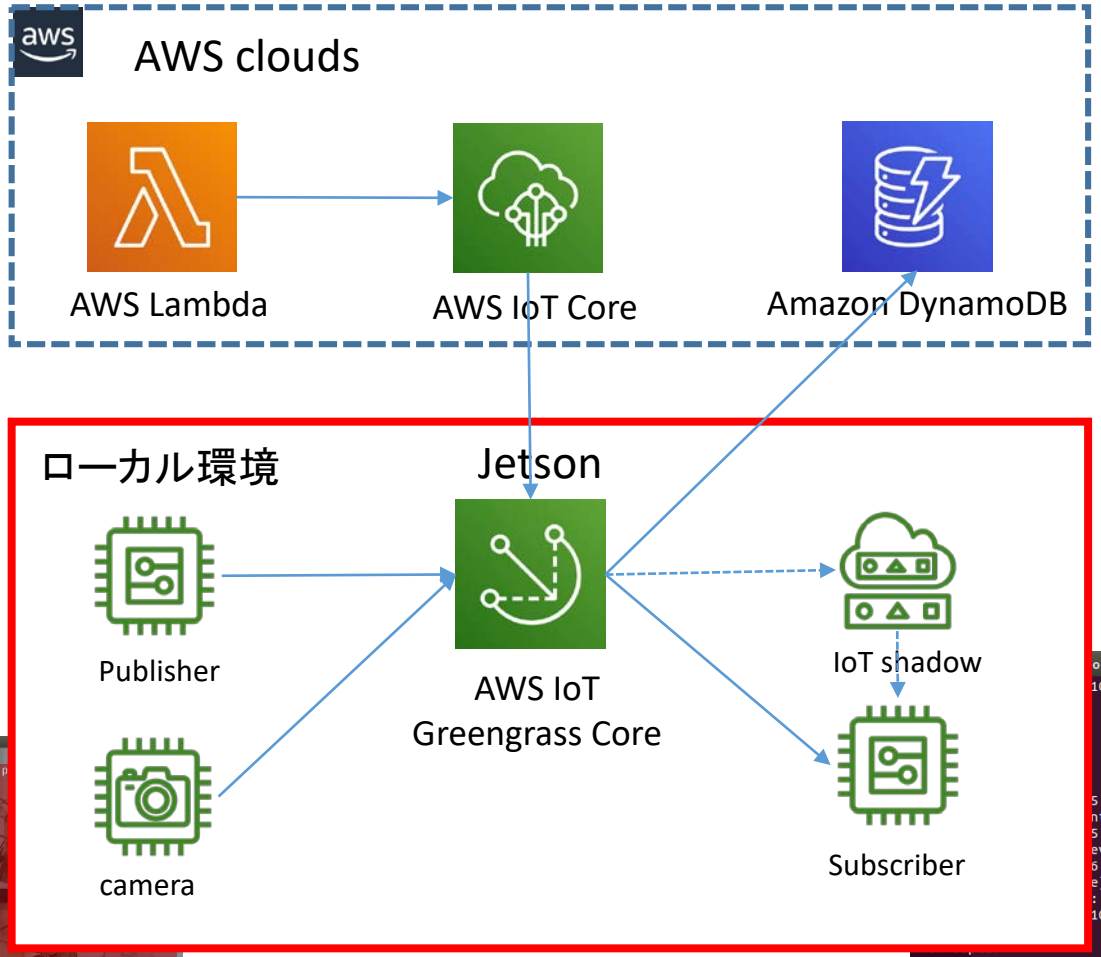


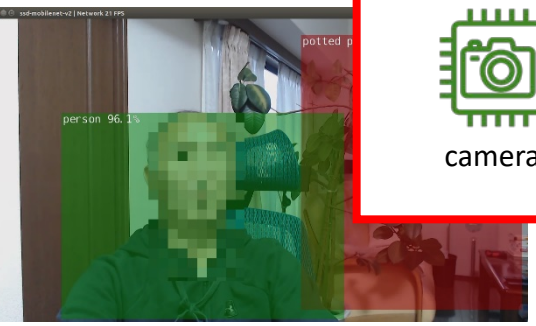
Jetsonがでない場合は、ipアドレスを直接入力
¥¥192.168.1.xx

Windowsネットワーク

- JETSONを選択
- Shareの配下にJetsonのフォルダが見えます。

開発キット設定
4. 全体構成





Spectrum Technology

開発キット設定
5. ハードウエア概要

品名	項目	内容	備考
Jetson nano	CPU	Quad-core ARM A57 @ 1.43 GHz	
	GPU	128-core Maxwell	
	メモリ	4GB RAM	
	OS	Ubuntu 18.04 LTS	
	インターフェース	1000base-T, USB 3.0x4, 2.0x1, HDMIx2, microSDカード, 40 GPIO pin,I2C, SPI,UART	WiFiは別ドングルが要
	電源／消費電力	Micro USB typeB　2A 5V（起動失敗することがあります、添付の5V4Aの電源を利用ください）	
	サイズ	115x90x50mm	ケース収納時
付属品		**内容**	**備考**
ケース		透明のみ	
ファン		小型ファン、速度調整はありません。	
5V4A AC/DC電源		AC100Vからの電源になります。	
microSD 64GB		Ubuntu OS及び必要なプログラム Python3.6, 関連pip 、opencv、tensorRT	
マニュアル		設定・開発編	

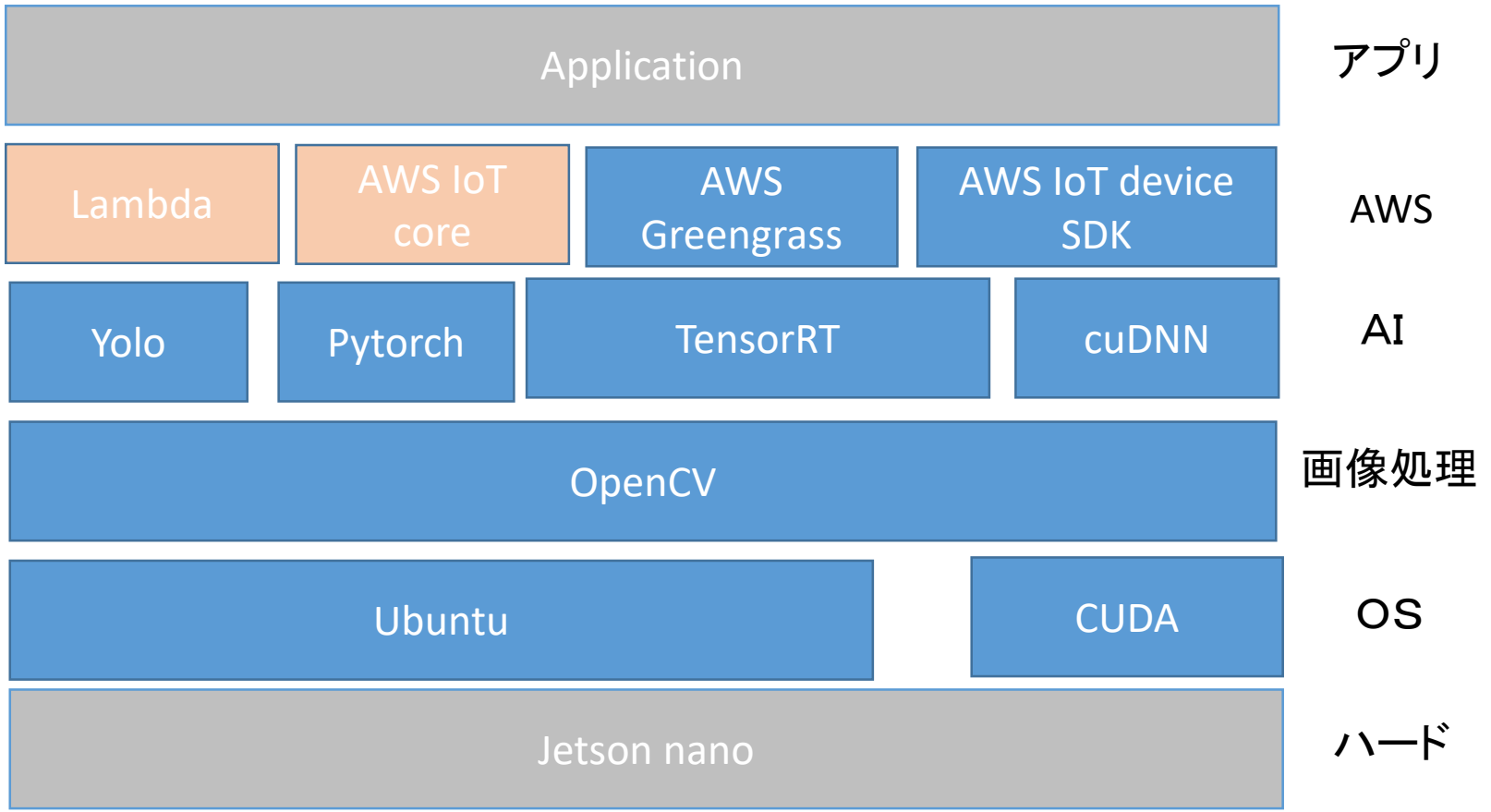
HDMIケーブルは付属しておりません。別途オプション品等を購入ください

開発キット設定
6. ソフトウエアの概要

区分	ソフト名	バージョン	備考
OS	ubuntu	18.04 LTS	
AWS	AWS greengrass	1.11	
	AWS IoT device SDK for python	1.4	
画像	Opencv	3.4.10	
プログラム	python3	3.6.9	AWS 使用時は、python3.7
	C++	7.5	C++は使用できますが、解説はpythonのみになります。
AI	TensorRT	7.1.3	Jetpack4.4使用
	cuDNN	8.0+cuda10.2	
	yolo	Yolov4他	
	Pytorch	1.6.0	
pip	pillow, matplotlib, numpy、scipy,keras, sklearn, requests, h5py,git,pip他		必要なモジュール

開発キット設定
7. プロトコルスタック

Python3：プログラム言語
C++:本資料では未説明



Ubuntu運用

1. Ubuntuについて

   Linuxの中でも一番シェアの高いOSです。2004年にDebian系から派生。

2. Linux基本コマンド

   ① システム関係

   - 起動：電源を入れると自動で起動します。
   - 再起動：$ reboot

     又は、左上のメニューの「ゲストを再起動
   - 終了： $ shutdown

     又は、左上のメニューの「ゲストをシャットダウン」
   - ログアウト $ exit

     ルートからログアウトします
   - 日本語／英語の入力切替：半角／全角のボタン（ESCボタンの下）

Ubuntu運用

2. Linux基本コマンド

② ディレクトリ操作、コピー、移動、削除

masa@jetson:~$cd /home/masa/Documents ディレクトリの切り替え
masa@jetson :/home/masa/Documents$ ls ファイルとディレクトリの表示（表示したら操作したいファイルを右クリックでコピペして操作します
masa@jetson:~$ cp ファイル名　ディレクトリ 配下のディレクトリのファイルを別のディレクトリへコピー
masa@jetson:~$ mv ファイル名　ディレクトリ 配下のディレクトリのファイルを別のディレクトリへ移動
masa@jetson:~$ rm ファイル名 ファイルの削除
便利な機能　　rm –help コマンドのオプションが分からない場合は、ヘルプで問い合わせる。すべてのコマンド共通(マイナスを2個とhelp)

③ ユーザ権限、プロセス他

masa@jetson :~ $ su - スーパーユーザ（root)に切り替え、パスワードを入力
masa@jetson:~$ sudo ルート権限で各種コマンドを実施します。
masa@jetson:~$ ps a 現状の動いているプロセスを表示
masa@jetson:~$ kill 特定のプロセスを強制終了
masa@jetson:~$ apt-get install pkg パッケージのインストールなどに使用
masa@jetson:~$ date 日付、時間の設定を行います。
masa@jetson:~$ leafpad /etc/network/interfaces インタフェースに記述してい内容を変更します。Viよりも使いやすいです。

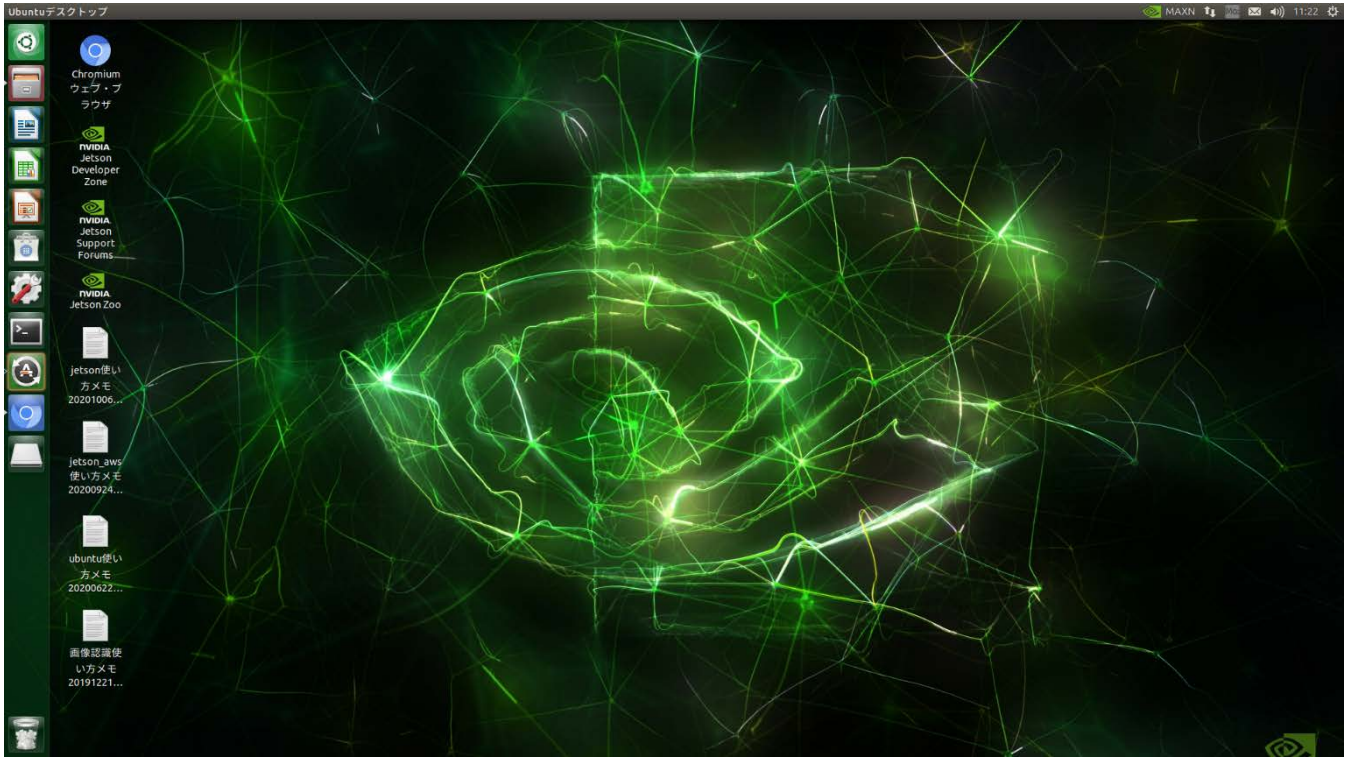
④ モジュール、usb、メモリ、HDDなどの表示

masa@jetson:~$ lsmod linuxのモジュールリスト表示
masa@jetson:~$ lsusb usbのデバイス表示
masa@jetson:~$ free –mt メモリ使用状態表示
masa@jetson:~$ df HDD（マイクロSD）の使用状態表示

Ubuntu運用

3.　基本操作

①　表示画面と内容



主に使用するもの
・ブラウザ：Chrome
・フォルダ：Documents内に必要なファイルがあります。
・コマンド：コマンド画面を立ち上げて、python3のプログラムを動作させます。

Jetson　AI開発

１．概要

- Jetsonは、Nvidiaが提供する、GPUを搭載し、高速化された並列処理用のビジュアルコンピューティングプラットフォームで画像認識、機械学習や自動運転等を想定した組込み用として開発された。CUDAに対応し、Edgeコンピューティング代表格である 。
- また、NVIDIA Jetson Nano は、Jetsonの中でも最小のモデルで、小型で低コスト、かつ低消費電力で、AI デバイス開発を可能にします。エントリレベルのネットワーク ビデオ レコーダー (NVR)、ホーム ロボット、分析機能をフルに搭載したインテリジェント ゲートウェイなど、組み込み IoT アプリケーションの新しい世界を開きます。
- Jetsonプロジェクト：
  https://developer.nvidia.com/embedded/community/jetson-projects

Jetson AI開発
2. 開発メニュー

① Hello AI world
- 画像分類
- 物体認識
- ライブカメラ物体認識
- 物体認識（色分け）
- ライブカメラ物体認識（色分け）
- 転移学習（Pytorch）
- 再学習（TensorRT）
  - 犬・猫編
  - 植物編

② AWS設定

③ AWS Greengrass

A) AWS Greengrassとは
B) 用途
C) AWS Greengrass設定
D) AWS Greengrass Core設定、起動
E) Hello world事例
F) デバイス間通信事例
G) デバイス・シャドー通信事例
H) クラウド連携（DynamoDB連携）事例
I) ライブカメラ物体認識事例

④ Yolo（Darknet）：写真、動画 物体認識

Jetson AI開発
3. 開発内容
① Hello AI world

コマンド
$ cd /home/masa/Documents/jetson-inference/build/aarch64/bin
$ python3 imagenet-console.py --network=googlenet images/orange_0.jpg output_0.jpg

デスクトップ上のよく使うコマンドからコピペしてください。

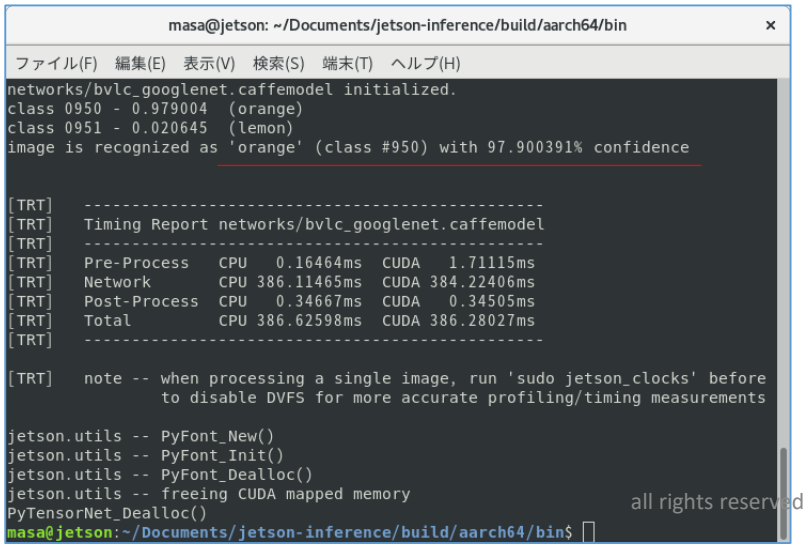
- https://github.com/dusty-nv/jetson-inference#hello-ai-world
- 画像分類 その1 python3
  - モデル：googlenet, ResNet-18を使って画像を分類し、確率を算出します。TensorRT使用
  - 以下のフォルダに移動し、プログラムを動作させます。

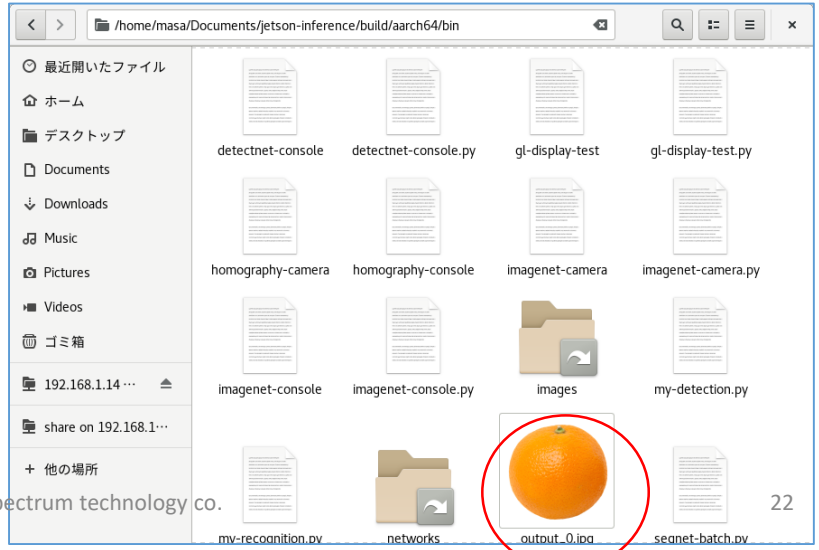
本キットでは、python3のみ利用可能です。

$ cd /home/masa/Documents/jetson-inference/build/aarch64/bin

$ python3 imagenet-console.py --network=googlenet images/orange_0.jpg output_0.jpg

  - --network=モデルを指定、images/orange_0.jpg：同じフォルダ内のimageのorange_0.jpgの写真を使い、確率を算出
  - Orangeで97%の確率で出力。1回目のモデル使用には、5分位学習にかかります。





Spectrum Technology

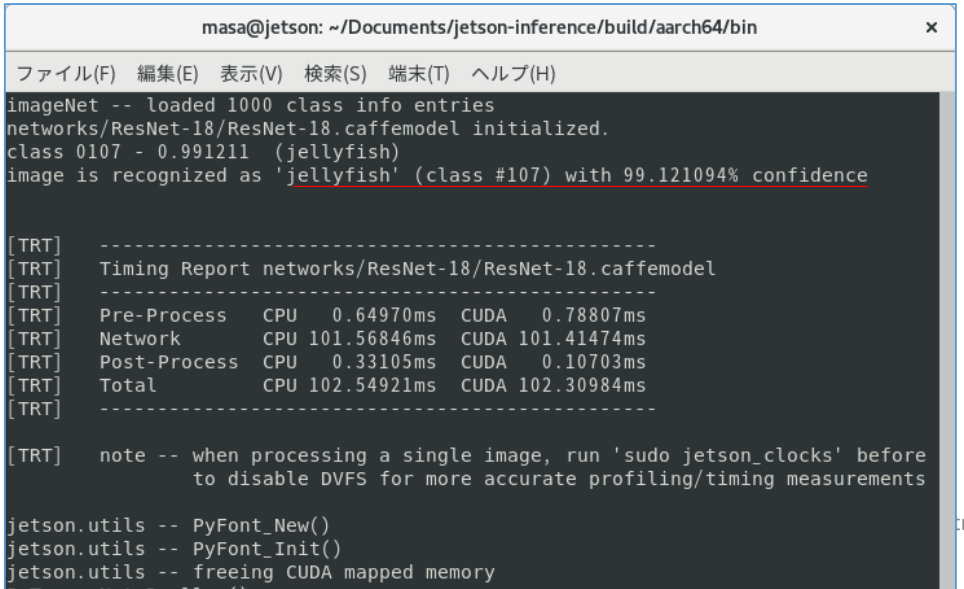
Jetson AI開発
3. 開発内容
① Hello AI world

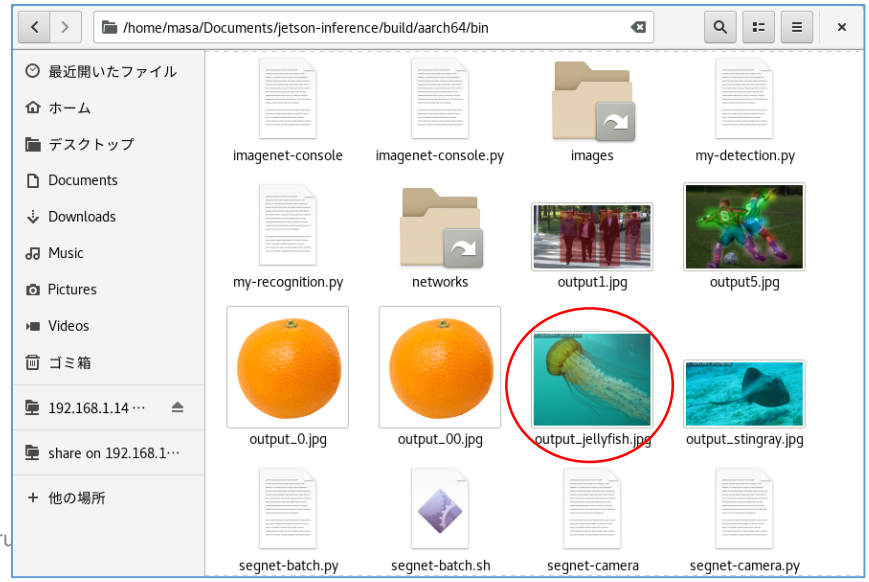
コマンド
$ cd /home/masa/Documents/jetson-inference/build/aarch64/bin
$ python3 imagenet-console.py --network=resnet-18 images/jellyfish.jpg output_jellyfish.jpg

- https://github.com/dusty-nv/jetson-inference#hello-ai-world
- 画像分類　その２ python3、resnet－１８
  - モデル：googlenet, ResNet-18を使って画像を分類し、確率を算出します。TensorRT使用
  - 以下のフォルダに移動し、プログラムを動作させます。

  $ cd /home/masa/Documents/jetson-inference/build/aarch64/bin

  $ python3 imagenet-console.py --network=resnet-18 images/jellyfish.jpg output_jellyfish.jpg
  - --network=モデルを指定、同じフォルダ内のimageのjellyfish.jpgの写真を使い、確率を算出
  - Jellyfishで９９%の確率で出力。１回目のモデル使用には、５分位学習にかかります。





Spectrum Technology

Jetson AI開発
3. 開発内容
① Hello AI world

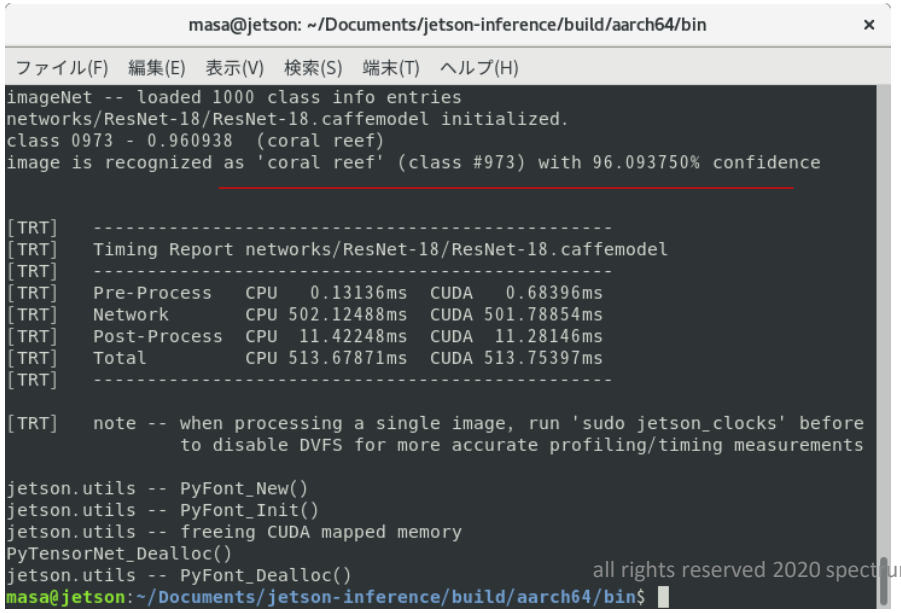
コマンド
$ cd /home/masa/Documents/jetson-inference/build/aarch64/bin
$ python3 imagenet-console.py --network=resnet-18 images/coral.jpg output_coral.jpg

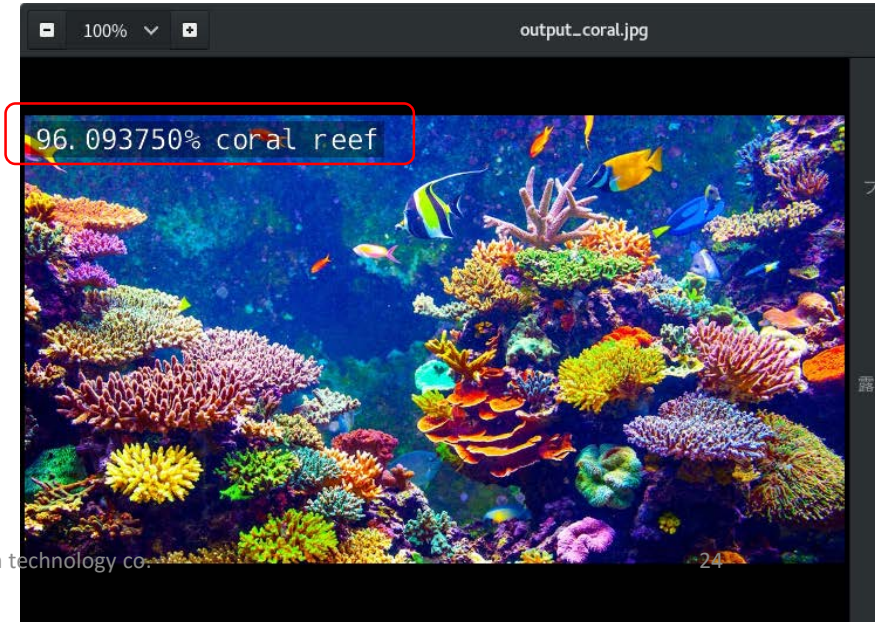
- https://github.com/dusty-nv/jetson-inference#hello-ai-world
- 画像分類 その3 python3、resnet－18
  - モデル：googlenet, ResNet-18を使って画像を分類し、確率を算出します。TensorRT使用
  - 以下のフォルダに移動し、プログラムを動作させます。

$ cd /home/masa/Documents/jetson-inference/build/aarch64/bin

$ python3 imagenet-console.py --network=resnet-18 images/coral.jpg output_coral.jpg

  - --network=モデルを指定、同じフォルダ内のimageのcoral.jpgの写真を使い、確率を算出
  - Coral reefで96%の確率で出力。1回目のモデル使用には、5分位学習にかかります。





Jetson AI開発
3. 開発内容
① Hello AI world

コマンド
$ cd /home/masa/Documents/jetson-inference/build/aarch64/bin
$ python3 imagenet-camera.py --camera=/dev/video0 --width=640 --height=480

- https://github.com/dusty-nv/jetson-inference#hello-ai-world
- ライブカメラ画像分類 python3
  - モデル:googlenet, ResNet-18を使って、ライブカメラで画像を分類し、確率を算出します。TensorRT使用
  - 以下のフォルダに移動し、プログラムを動作させます。

$ cd /home/masa/Documents/jetson-inference/build/aarch64/bin

$ python3 imagenet-camera.py --camera=/dev/video0 --width=640 --height=480

  - --network=指定しない場合は、googlenet, camera= =/dev/video0:webカメラのデバイス、--width, --heightは画面サイズです。
  - フレームの調整が上手くゆきませんでした。フレーム全体を左にドラッグすると左半分になります。

コマンド
$ cd /home/masa/Documents/jetson-inference/build/aarch64/bin
$ python3 detectnet-console.py --network=ssd-mobilenet-v2 images/peds_0.jpg output1.jpg

Jetson AI開発
3. 開発内容
① Hello AI world

- https://github.com/dusty-nv/jetson-inference/blob/master/docs/detectnet-console-2.md
- 物体認識 その1 歩行者 python3
  - モデル：ssd-mobilenet-v2（91種類の物体を認識、詳細はこちら）を使って、物体を認識し、確率を算出します。TensorRT使用
  - 以下のフォルダに移動し、プログラムを動作させます。

$ cd /home/masa/Documents/jetson-inference/build/aarch64/bin

$ python3 detectnet-console.py --network=ssd-mobilenet-v2 images/peds_0.jpg output1.jpg

  - 90%以上の確率で人物を認識。

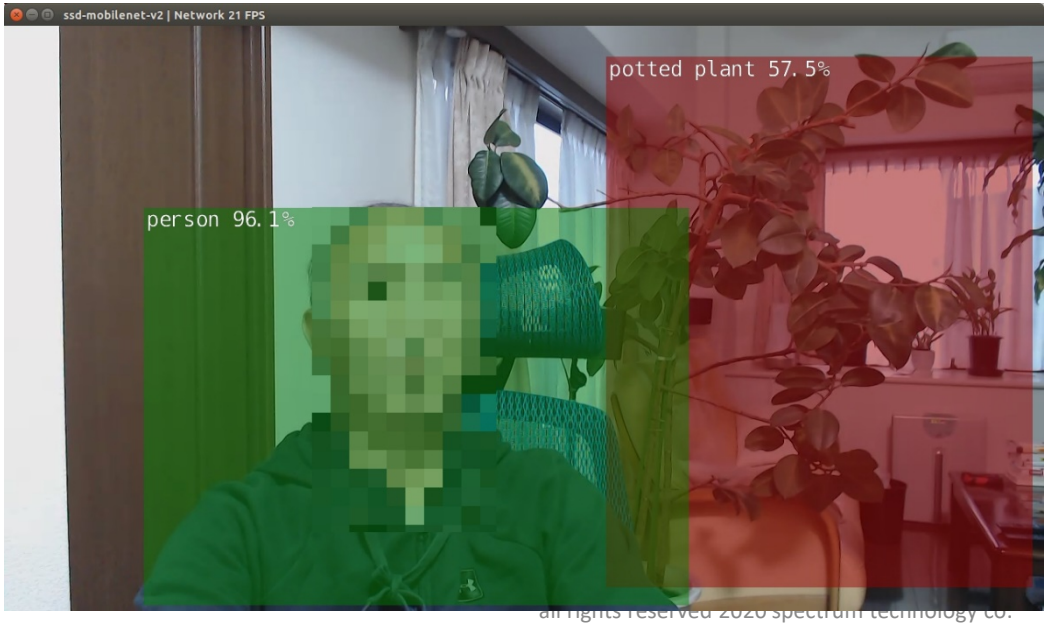
Jetson AI開発
3. 開発内容
① Hello AI world

コマンド
$ cd /home/masa/Documents/jetson-inference/build/aarch64/bin
$ python3 detectnet.py /dev/video0

- https://github.com/dusty-nv/jetson-inference#hello-ai-world
- ライブカメラ物体識別その２ python3
  - モデル：ssd-mobilenet-v2（９１種類の物体を認識、詳細はこちら）により、ライブカメラを使って物体を識別し、確率を算出します。TensorRT使用
  - 以下のフォルダに移動し、プログラムを動作させます。前ページよりも簡単

$ cd /home/masa/Documents/jetson-inference/build/aarch64/bin

$ python3 detectnet.py /dev/video0



Jetson AI開発
3. 開発内容
① Hello AI world

コマンド
$ cd /home/masa/Documents/jetson-inference/build/aarch64/bin
$ python3 segnet-console.py --network=fcn-resnet18-cityscapes images/city_0.jpg output4.jpg

- https://github.com/dusty-nv/jetson-inference#hello-ai-world
- 物体色分け 市街地景色 python3
  - モデル: fcn-resnet18-cityscapes（市街地の景色）を使って物体を識別し、色分けします。TensorRT使用
  - 以下のフォルダに移動し、プログラムを動作させます。

$ cd /home/masa/Documents/jetson-inference/build/aarch64/bin

$ python3 segnet-console.py --network=fcn-resnet18-cityscapes images/city_0.jpg output4.jpg

  - 右のクラスで画像を色分けします。自動運転などに利用します。

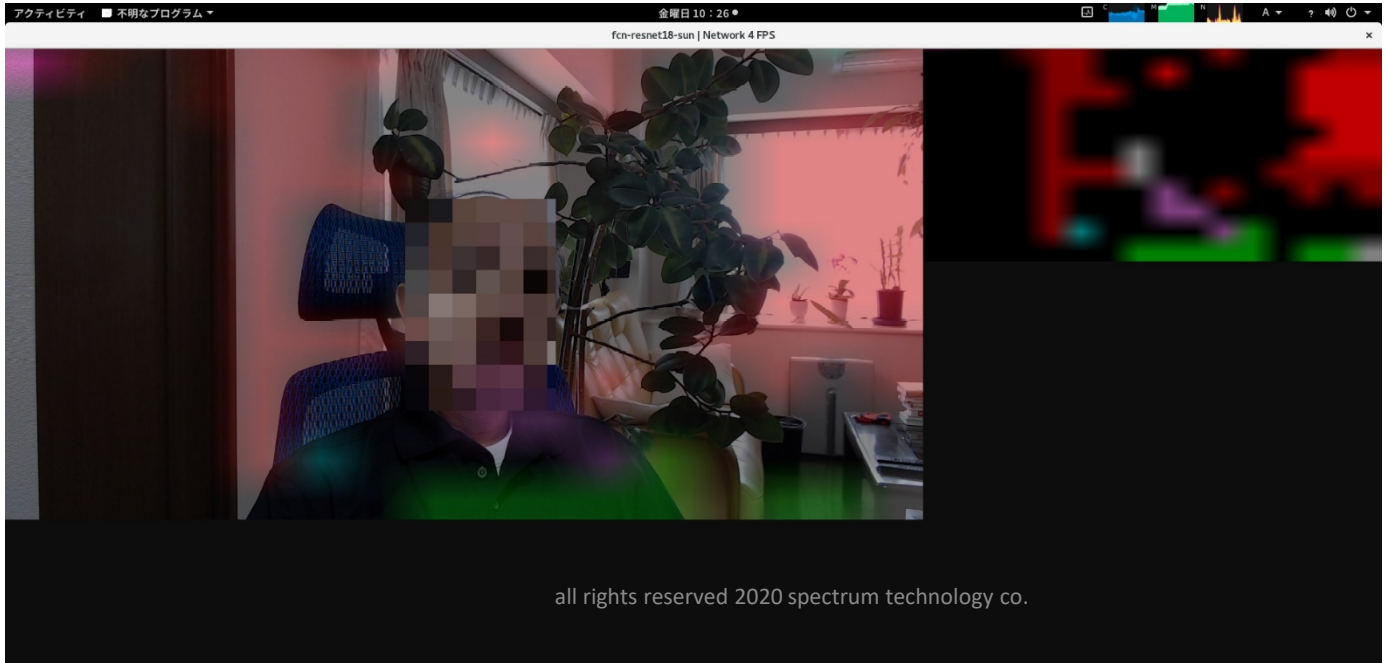
Jetson AI開発
3. 開発内容
① Hello AI world

コマンド
$ cd /home/masa/Documents/jetson-inference/build/aarch64/bin
$ python3 segnet-camera.py --network=fcn-resnet18-sun --camera=/dev/video0

- https://github.com/dusty-nv/jetson-inference#hello-ai-world
- ライブカメラ物体色分け　屋内の家具、壁　python3
  - モデル：fcn-resnet18-sun（屋内の家具、壁）を使って物体を識別し、色分けします。TensorRT使用
  - 以下のフォルダに移動し、プログラムを動作させます。

$ cd /home/masa/Documents/jetson-inference/build/aarch64/bin

$ python3 segnet-camera.py --network=fcn-resnet18-sun --camera=/dev/video0



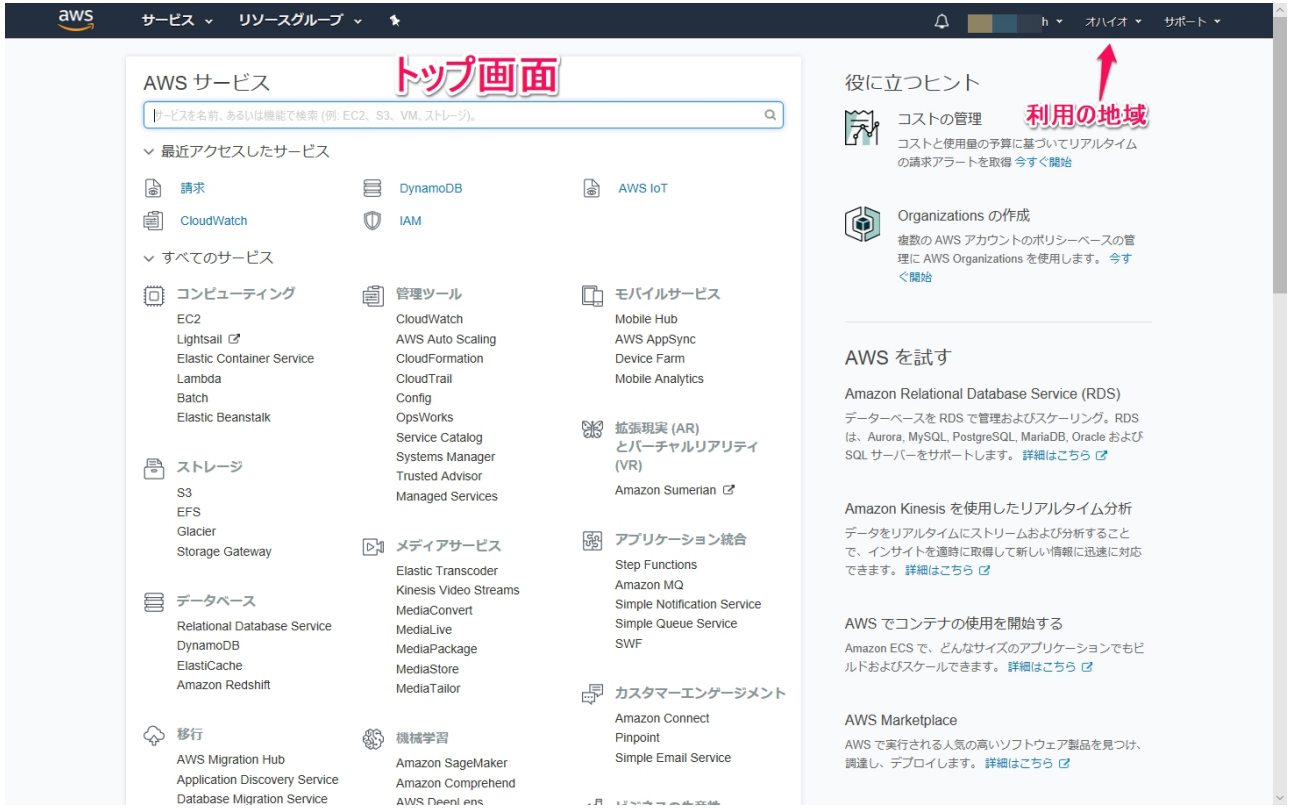
Jetson　AI開発
3. 開発内容
②　AWS設定

- 手順1：AWSアカウントの作成
  - https://aws.amazon.com/jp/register-flow/
  - 必要なメールアドレス、パスワードなどを入力します。

Spectrum Technology

Jetson　AI開発
3. 開発内容
②　AWS設定

- AWSトップ画面
  - 利用する場合に、地域を意識して設定してください。地域毎に料金が変わったり、利用できるサービスが限定されている場合があります。

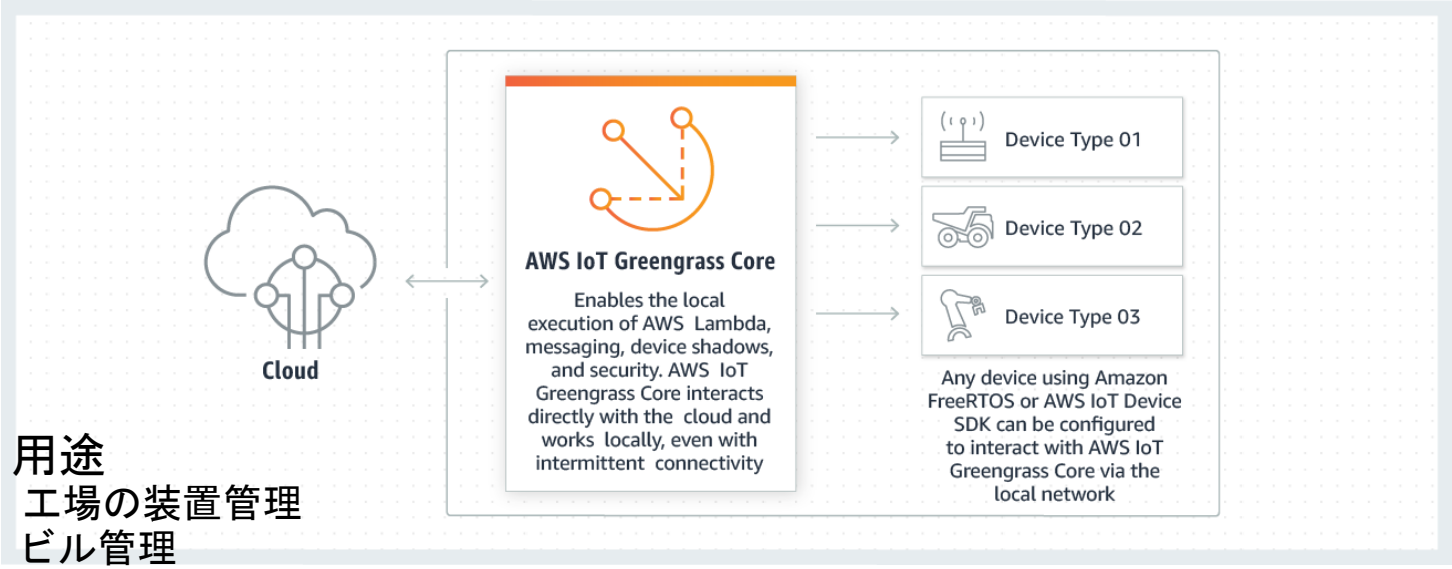


Jetson AI開発
3. 開発内容
③ AWS Greengrass

- https://docs.aws.amazon.com/ja_jp/greengrass/latest/developerguide/what-is-gg.html

A) AWS Greengrassとは

- ネット環境がなくても、ローカル側でAWS IoT環境を実現します。また必要によりクラウド側のAWS IoT、DynamoDBなどとも通信ができます。
- ローカル側のデバイスとMQTTで通信し、クラウドと同じ動きをします。
- ローカル側での通信は無料です。
- ローカル側のみで機械語学習の推論（AI）が利用できます。



B) 用途

- 工場の装置管理
- ビル管理
- 農場の環境管理
- 監視カメラの情報共有

Jetson AI開発
3. 開発内容
③ AWS Greengrass

- https://docs.aws.amazon.com/ja_jp/greengrass/latest/developerguide/gg-config.html

C） AWS Greengrass設定

- AWS IoTのコンソールにログイン
- AWSのトップ画面から、AWS IoT core又は IoT Greengrassを選択
- AWS Greengrassのグループを作成して、証明書などをダウンロードします。

A) グループ作成を選択

コマンド
$ cd /home/masa/Documents/helloworld

Jetson AI開発
3. 開発内容
③ AWS Greengrass

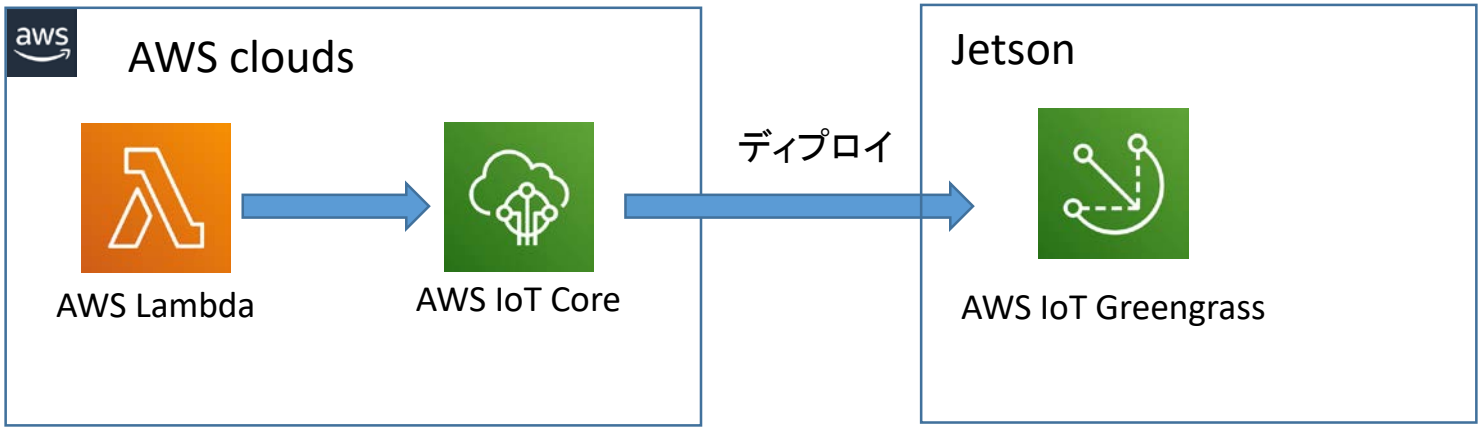
- https://docs.aws.amazon.com/ja_jp/greengrass/latest/developerguide/module3-I.html

E） Hello world事例

- 最も基本的な、Hello worldの事例です。
- 添付のzipファイルを使ってLambda関数を作成し、AWS IotコンソールからJetsonにディプロイします。下図に流れを記載します。

$ cd /home/masa/Documents/helloworld

hello_world_python_lambda.zip　　Jetsonのフォルダにあるzipファイルを PC側に取り出します

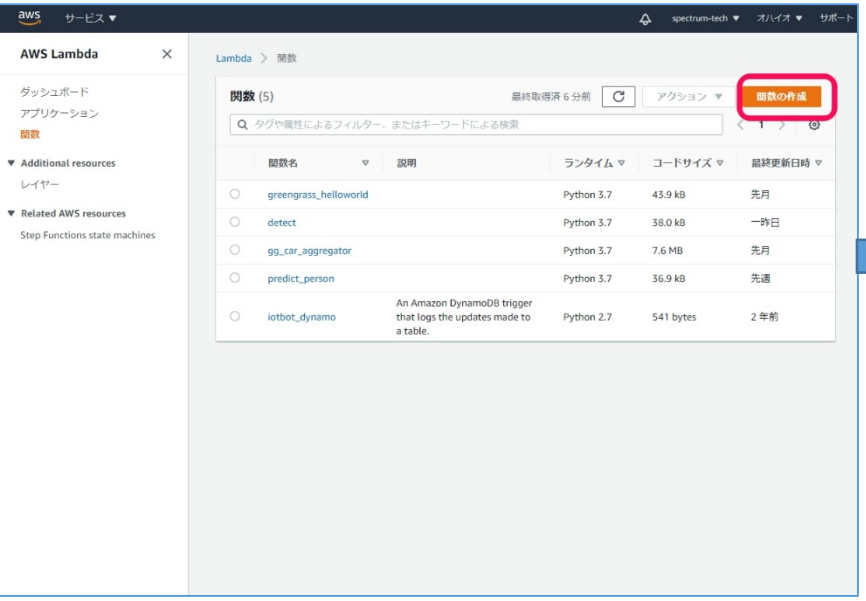


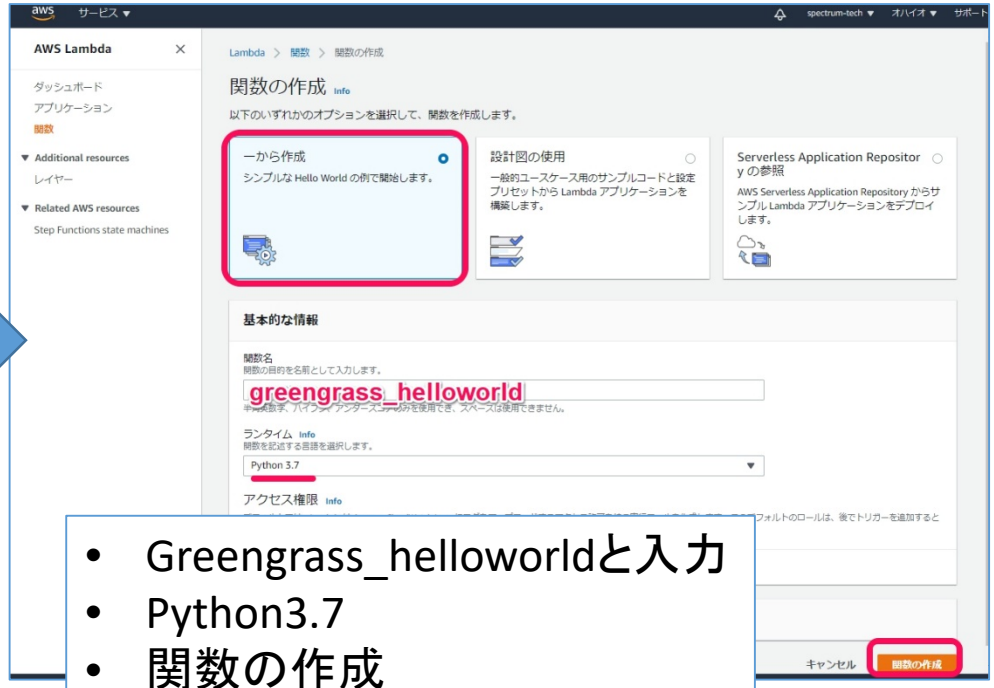
Jetson AI開発
3. 開発内容
③ AWS Greengrass

- https://docs.aws.amazon.com/ja_jp/greengrass/latest/developerguide/module3-I.html

E） Hello world事例

- 最も基本的な、Hello worldの事例です。
- 添付のzipファイルを使ってLambda関数を作成し、AWS IotコンソールからJetsonにディプロイします。
- Lambda関数の作成

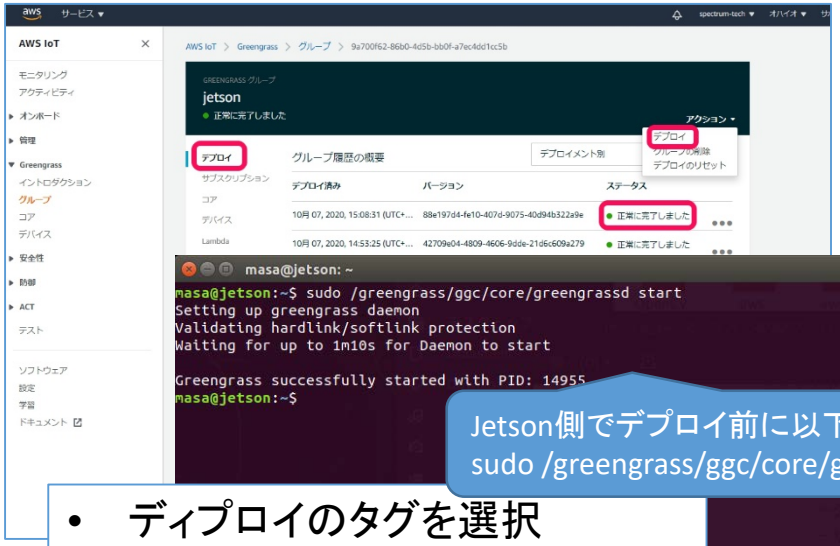




- Greengrass_helloworldと入力
- Python3.7
- 関数の作成

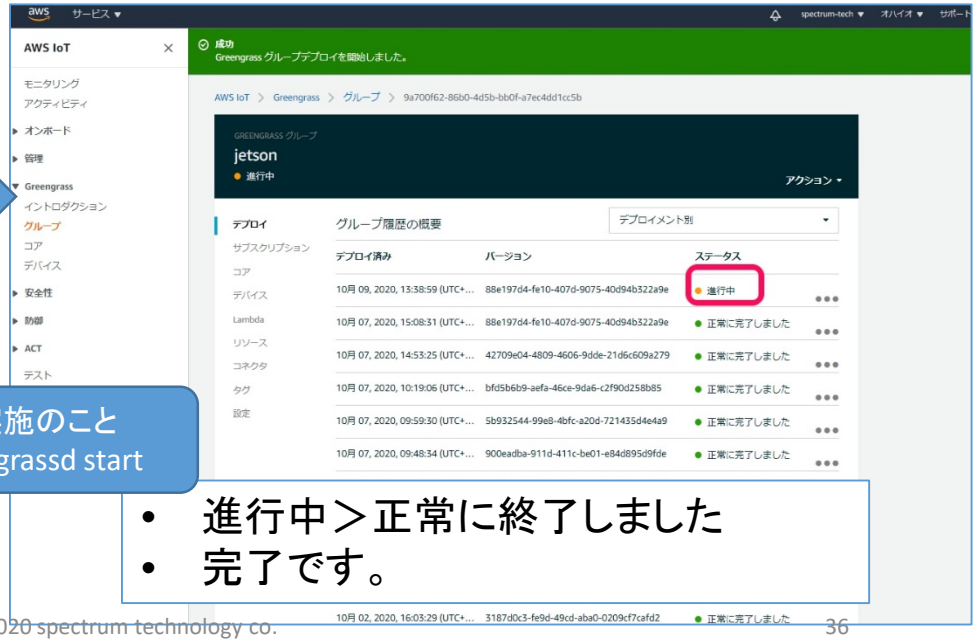
Jetson AI開発
3. 開発内容
③ AWS Greengrass

- https://docs.aws.amazon.com/ja_jp/greengrass/latest/developerguide/module3-I.html
- E） Hello world事例
  - 最も基本的な、Hello worldの事例です。
  - 添付のzipファイルを使ってLambda関数を作成し、AWS Iotコンソールから Jetsonにディプロイします。
  - AWS IoTコンソールで、ディプロイデータを作成
  - 既に作成した、グループのjetsonにlambda関数、サブスクリプションを追加します。



Jetson側でデプロイ前に以下を実施のこと
sudo /greengrass/ggc/core/greengrassd start

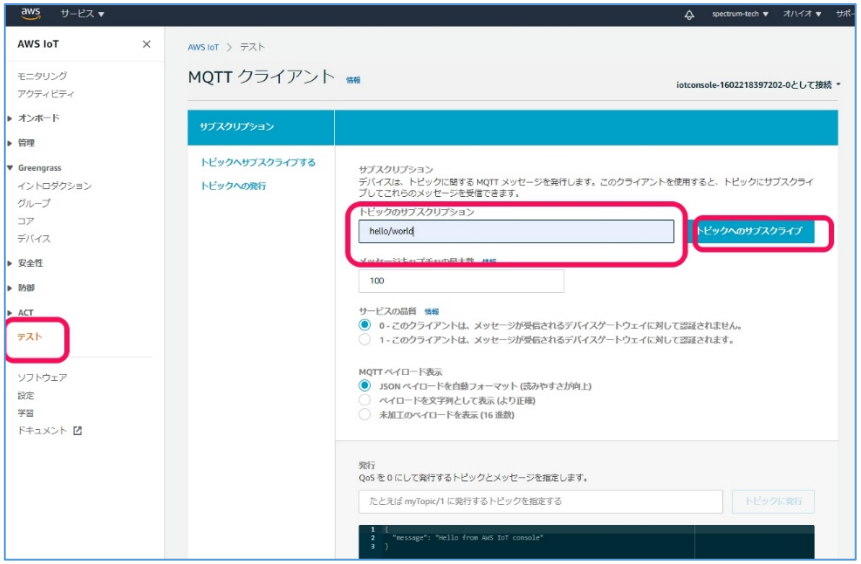
- ディプロイのタグを選択
- アクション＞ディプロイの実施

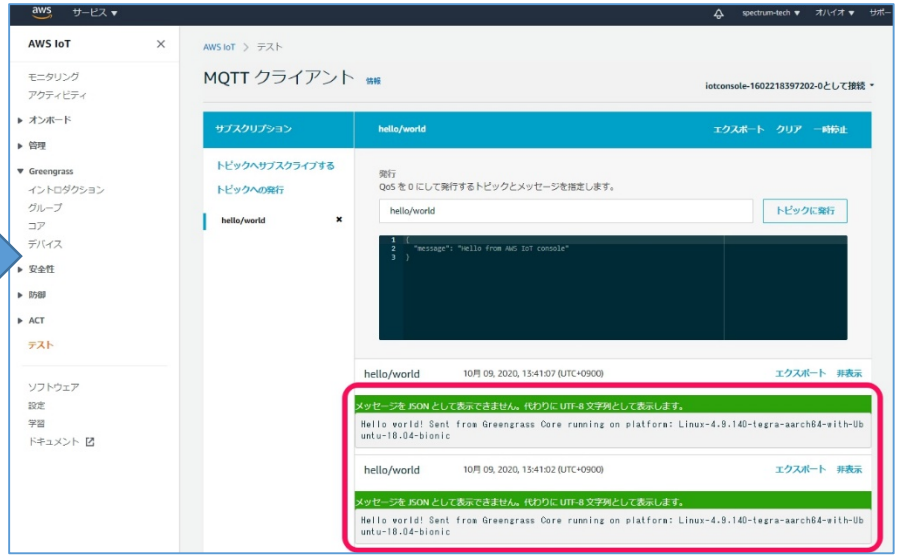


- 進行中＞正常に終了しました
- 完了です。

Jetson AI開発
3. 開発内容
③ AWS Greengrass

- https://docs.aws.amazon.com/ja_jp/greengrass/latest/developerguide/module3-I.html
- E） Hello world事例
  - 最も基本的な、Hello worldの事例です。
  - ディプロイ後、AWS IoTコンソールでサブスクリプションのテストを実施。



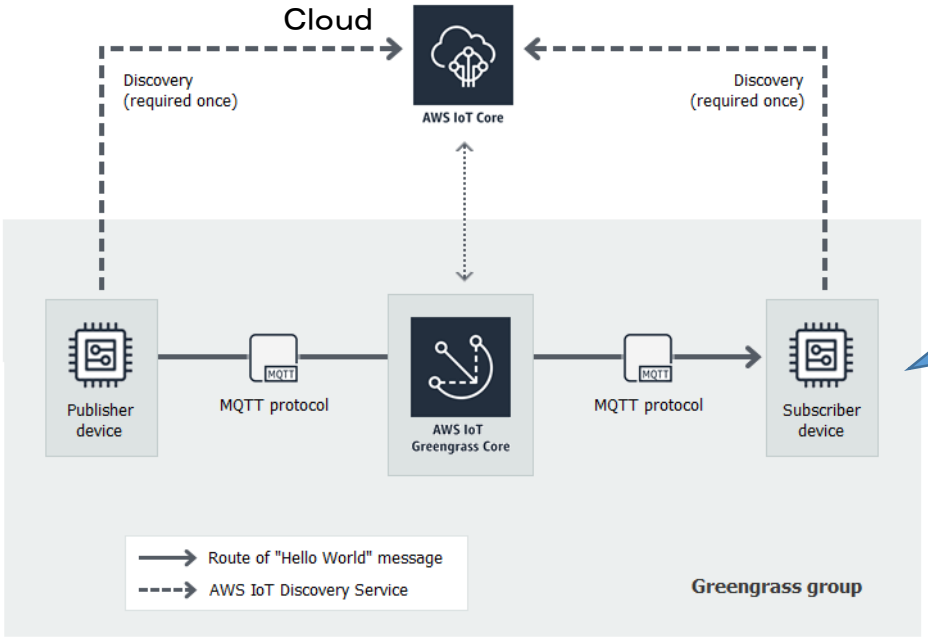


- テストのタグを選択
- トピック：hello/worldと入力
- サブスクリプション

- Jetson側と通信ができていると
- Helloworldと端末バージョンが次々表示されます。

Jetson AI開発
3. 開発内容
③ AWS Greengrass

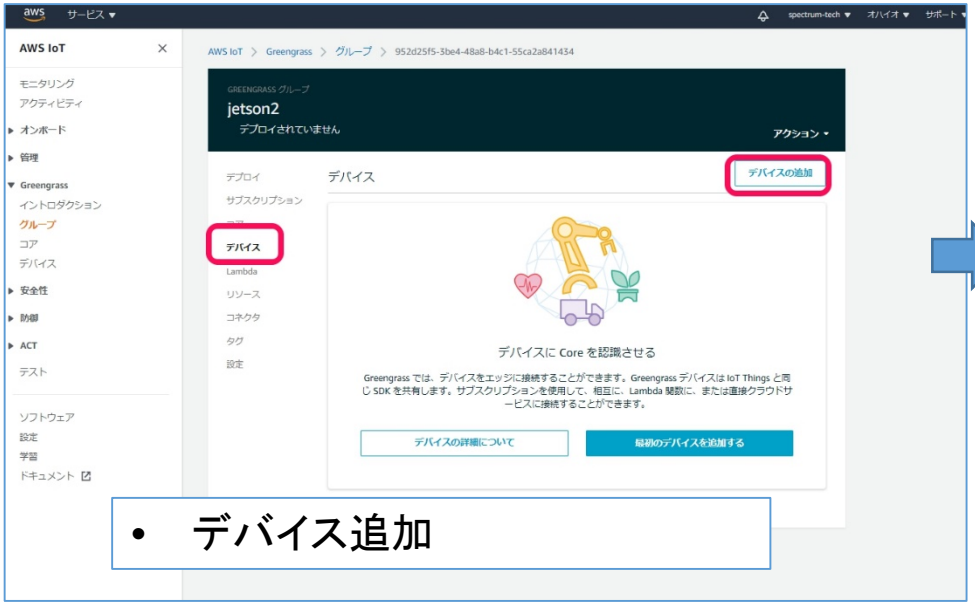
- https://docs.aws.amazon.com/ja_jp/greengrass/latest/developerguide/module4.html
- F） デバイス間通信事例
  - 基本的な、MQTTプロトコルを使い、IoT Greegrassをコアとして、二つのデバイスでPublish，Subscribeで通信を行う事例です。
  - Pub／SubのデバイスをAWS IoTコンソールで作成し、サブスクリプションを設定します。今回はLumbda関数は使用しません。
  - 通信回線が不要でローカル環境で動作します。なお設定時は回線が必要です。



今回は、IoT Greengrass，Pub，Subは同一のJetsonに設定します。

Jetson AI開発
3. 開発内容
③ AWS Greengrass

- https://docs.aws.amazon.com/ja_jp/greengrass/latest/developerguide/module4.html
- F） デバイス間通信事例
  - 基本的な、MQTTプロトコルを使い、IoT Greegrassをコアとして、二つのデバイスでPublish，Subscribeで通信を行う事例です。
  - Pub／SubのデバイスをAWS IoTコンソールで作成し、サブスクリプションを設定します。今回はLumbda関数は使用しません。
  - Pub／Subデバイス設定。



- デバイス追加

- 新しいデバイスの作成。

Spectrum Technology

Jetson AI開発
3. 開発内容
③ AWS Greengrass

コマンド
$ cd /home/masa/Documents/aws-iot-device-sdk-python/samples/greengrass/publish
$ python3 basicPubSub.py --endpoint greengrass-ats.iot.us-east-2.amazonaws.com --rootCA root.ca.pem --publisher.cert.pem --key publisher.private.key --clientId HelloWorld_Publisher --topic 'hello/world/pubsub' --mode publish --message 'Hello, World! Sent from HelloWorld_Publisher'

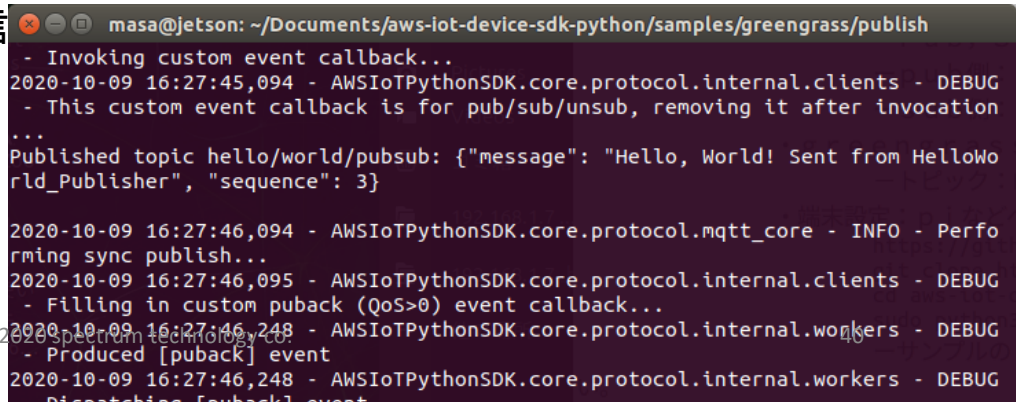
- https://docs.aws.amazon.com/ja_jp/greengrass/latest/developerguide/module4.html
- F） デバイス間通信事例
  - 基本的な、MQTTプロトコルを使い、IoT Greegrassをコアとして、二つのデバイスでPublish，Subscribeで通信を行う事例です。
  - Pub／SubのデバイスをAWS IoTコンソールで作成し、サブスクリプションを設定します。今回はLumbda関数は使用しません。
  - Jetson側で通信テスト。
  - エンドポイント名確認

$ cd /home/masa/Documents/aws-iot-device-sdk-python/samples/greengrass/publish

  - Publish側で使用するダウンロードした証明書を解凍

$ python3 basicPubSub.py --endpoint greengrass-ats.iot.us-east-2.amazonaws.com --rootCA root.ca.pem --cert publisher.cert.pem --key publisher.private.key --clientId HelloWorld_Publisher --topic 'hello/world/pubsub' --mode publish --message 'Hello, World! Sent from HelloWorld_Publisher'

  - 送信するとpublished xxと2秒おきに送信



Jetson AI開発
3. 開発内容
③ AWS Greengrass

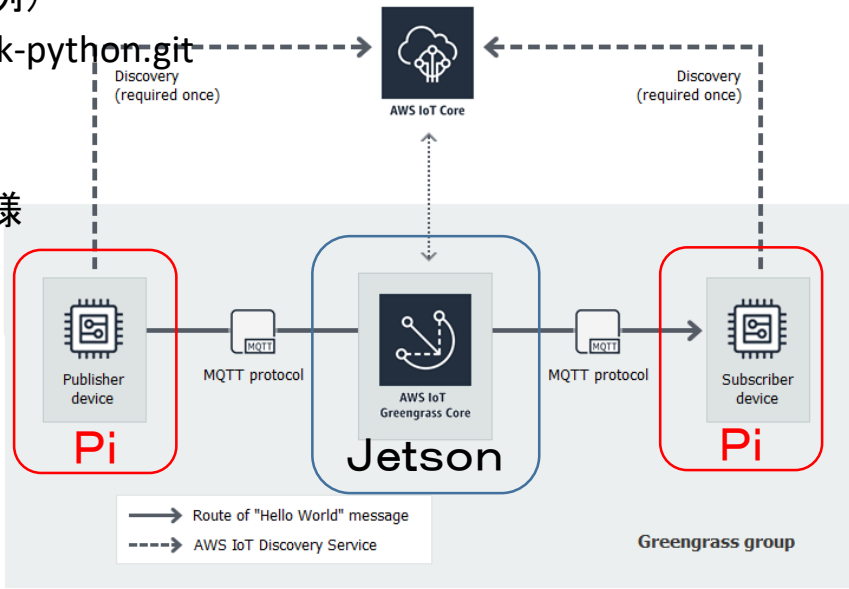
- https://docs.aws.amazon.com/ja_jp/greengrass/latest/developerguide/IoT-SDK.htm
- F） デバイス間通信事例
  - 基本的な、MQTTプロトコルを使い、IoT Greegrassをコアとして、二つのデバイスでPublish，Subscribeで通信を行う事例です。
  - Pub／SubのデバイスをAWS IoTコンソールで作成し、サブスクリプションを設定します。今回はLumbda関数は使用しません。
  - 同一セグメントの別端末からのPub／Sub設定と通信
  - AWS IoT Device SDK for Python をインストール（Pi例）

$ git clone https://github.com/aws/aws-iot-device-sdk-python.git

$ cd aws-iot-device-sdk-python

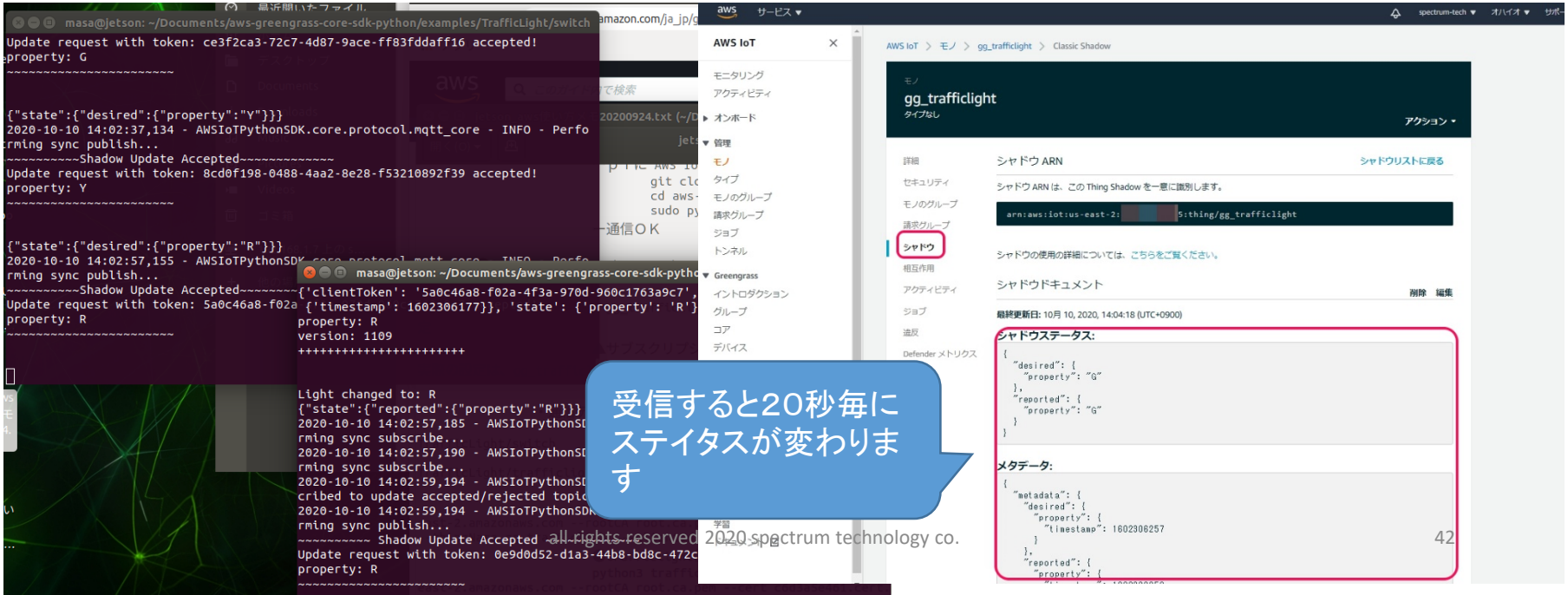
$ sudo python3 setup.py install

  - 通信テストは、Jetsonで実施した、PubSubと同様



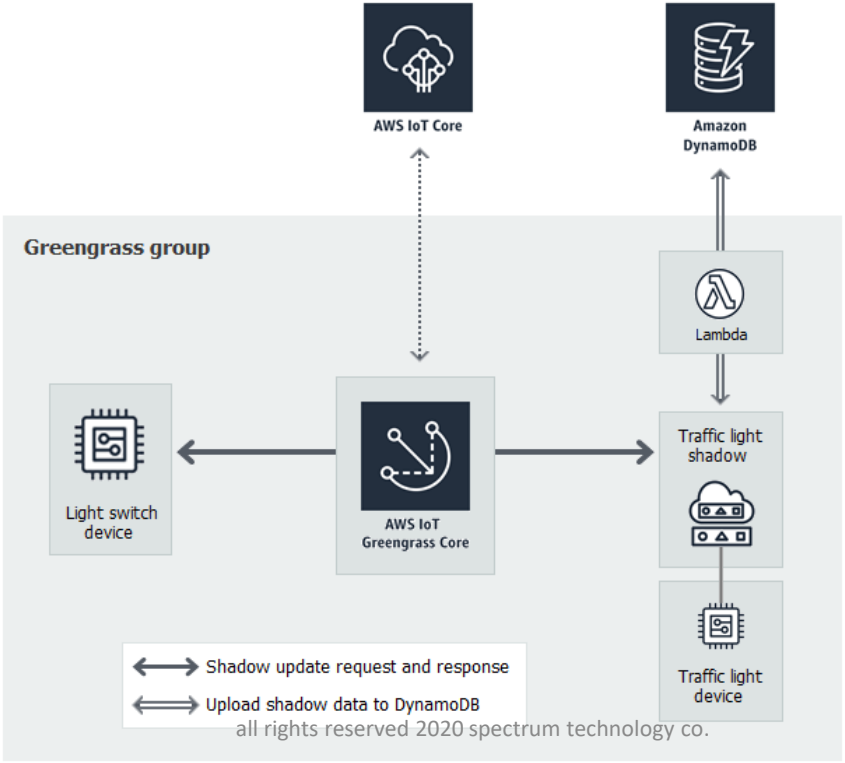
Jetson AI開発
3. 開発内容
③ AWS Greengrass

- https://docs.aws.amazon.com/ja_jp/greengrass/latest/developerguide/module5.html
- G） デバイス・シャドー通信事例
  - 信号機（Traffic light）と信号制御器（Light switch）間の情報をシャドーを通して制御する事例
  - クラウド側と情報を同期することも出来ます。
  - Traffic lightのクラウド同期の開始。
  - Jetsonで switch, trafficlightの通信を開始
  - AWS IoTコンソール：管理＞モノ＞gg_trafficlight>シャドウ＞classic shadowを選択



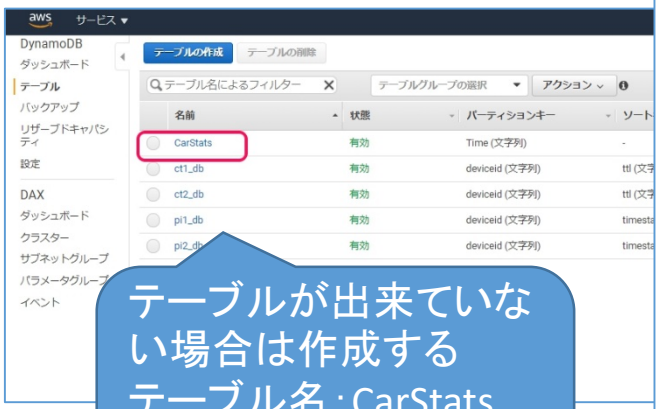
Jetson AI開発
3. 開発内容
③ AWS Greengrass

- https://docs.aws.amazon.com/ja_jp/greengrass/latest/developerguide/module6.html
- H)クラウド連携(DynamoDB連携)事例
  - 信号機(Traffic light)と信号制御器(Light switch)間の情報をシャドーを通して制御する事例に追加してDynamoDBにデータをアップします。
  - グループロール作成、Lambda関数作成・設定、サブスクリプション設定、ディプロイ、通信テスト

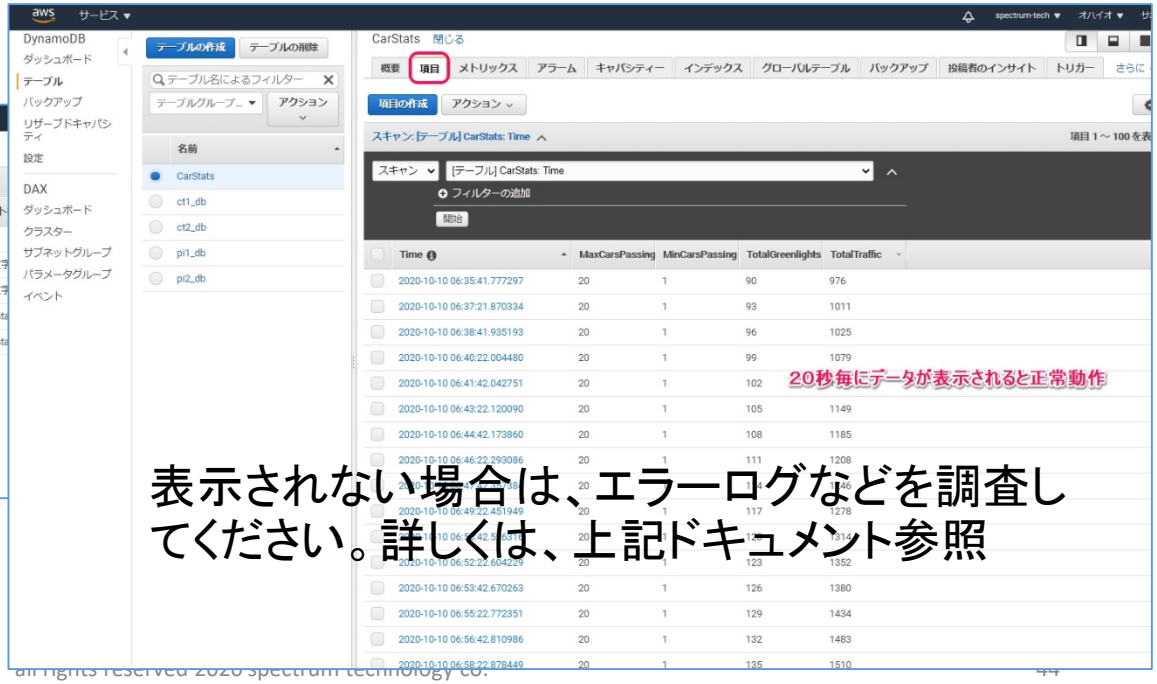


Jetson AI開発
3. 開発内容
③ AWS Greengrass

- https://docs.aws.amazon.com/ja_jp/greengrass/latest/developerguide/config_subs.html
- H)クラウド連携(DynamoDB連携)事例
  - 信号機(Traffic light)と信号制御器(Light switch)間の情報をシャドーを通して制御する事例に追加してDynamoDBにデータをアップします。
  - DynamoDB確認
  - Jetson側でTrafficlight, switchを動作させ、そのデータが約2分毎に入っているかDynamoDBを確認する



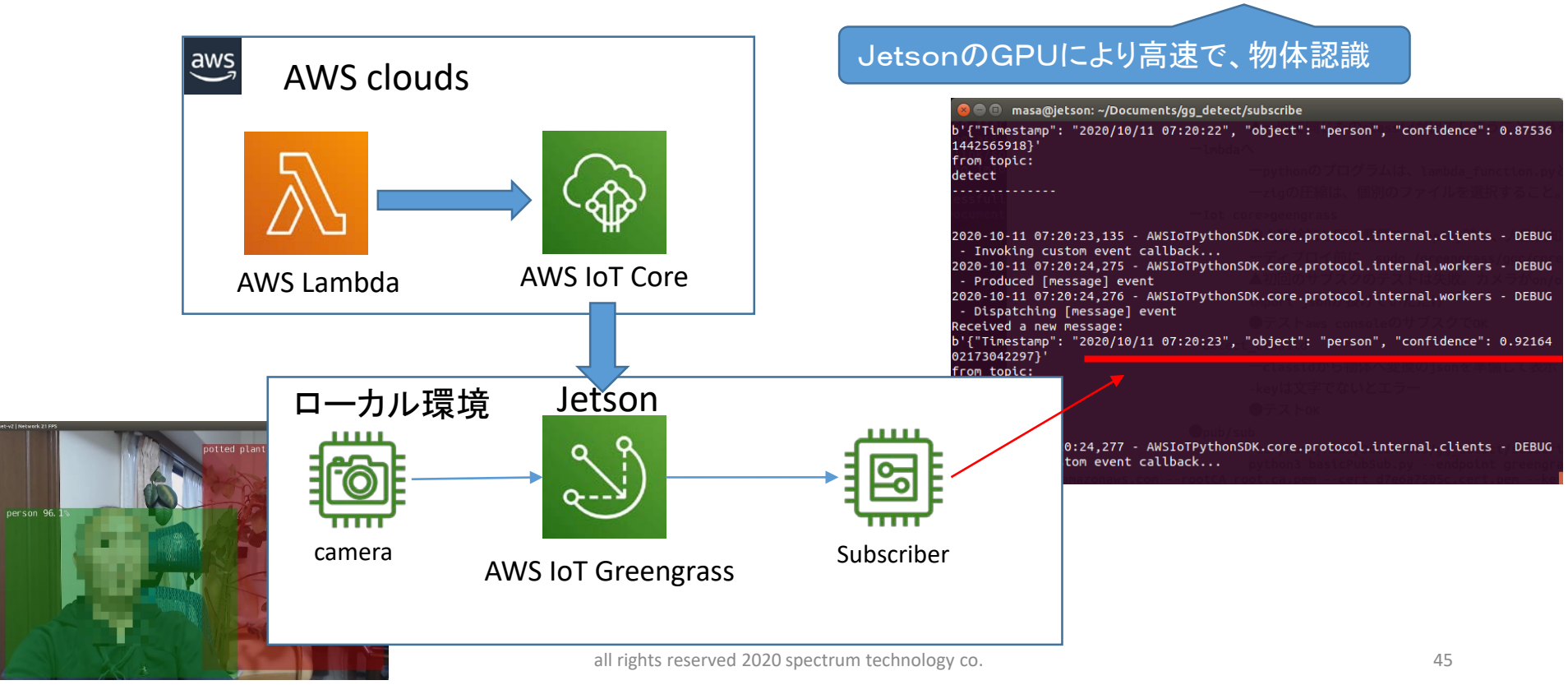
テーブルが出来ていない場合は作成する
テーブル名 : CarStats
パーテーション : Time
で作成



表示されない場合は、エラーログなどを調査してください。詳しくは、上記ドキュメント参照

Jetson AI開発
3. 開発内容
③ AWS Greengrass

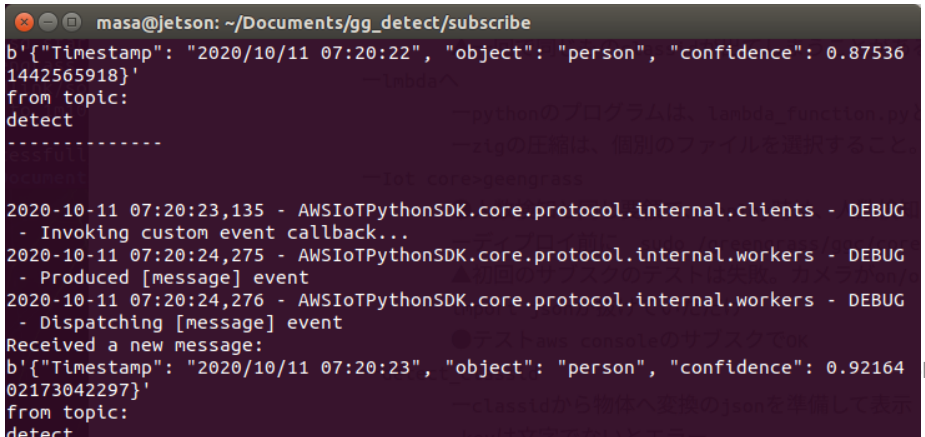
- I） ライブカメラ物体認識事例
  - Hello AI Worldで説明したライブカメラによる物体認識を利用して、そのプログラムをLambda関数を使い、Greengrassに設定。カメラが認識した物体と確率をSubsriberにリアルタイム出力します。



Spectrum Technology

Jetson AI開発
3. 開発内容
③ AWS Greengrass

- https://github.com/dusty-nv/jetson-inference/blob/master/docs/detectnet-camera-2.md
- I） ライブカメラ物体認識事例
  - Hello AI Worldで説明したライブカメラによる物体認識を利用して、そのプログラムをLambda関数を使い、Greengrassに設定。カメラが認識した物体と確率をSubsriberにリアルタイム出力します。
  - 通信テスト
  - カメラが自動で起動します。Subscriber側を動作させ物体認識のデータを確認する。
  - $ cd /home/masa/Documents/gg_detect/subscribe
  - $ python3 basicPubSub.py --endpoint greengrass-ats.iot.us-east-2.amazonaws.com --rootCA root.ca.pem –cert subscriber.cert.pem --key subscriber..private.key --clientId HelloWorld_Subscriber --topic 'detect' --mode subscribe



Spectrum Technology

Jetson　AI開発
3. 開発内容
④Yolo（Darknet）

コマンド
$ cd /home/masa/darknet
$ ./darknet detector test cfg/coco.data cfg/yolov4.cfg yolov4.weights

- 写真　物体認識
  - https://github.com/AlexeyAB/darknet
  - Darknetが提供するYoloは、一番高速で物体認識をするモデルです。TEDで紹介されています。
  - モデル：Yolov4。最新モデルで詳細に認識
  - 以下のフォルダに移動して、プログラムを動作

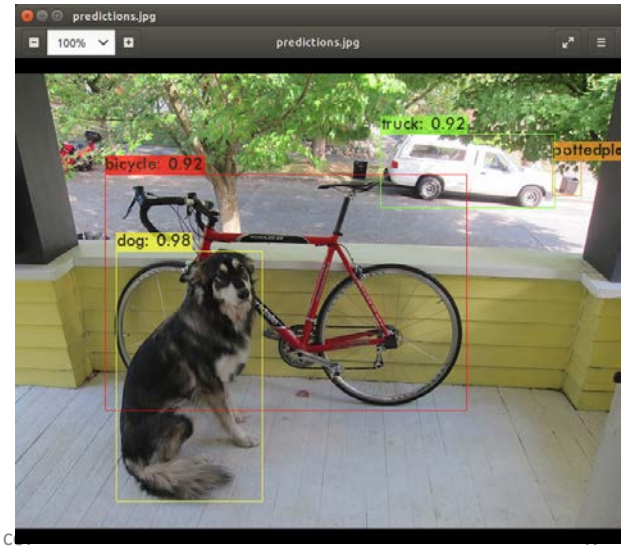
$　cd /home/masa/darknet

  - 写真

$ ./darknet detector test cfg/coco.data cfg/yolov4.cfg yolov4.weights

$ enter imagepath: data/dog.jpgと入力します。

  - 同じ写真で4つの物体を識別



Spectrum Technology

Jetson　AI開発
3. 開発内容
④　Yolo（Darknet）

コマンド
$ cd /home/masa/darknet
$ ./darknet detector demo cfg/coco.data cfg/yolov3-tiny.cfg yolov3-tiny.weights -c 0

- 動画　物体認識
  - https://github.com/AlexeyAB/darknet
  - Darknetが提供するYoloは、一番高速で物体認識をするモデルです。TEDで紹介されています。
  - モデル：Yolov3。前出のJetsonでも使われています
  - 以下のフォルダに移動して、プログラムを動作

$　cd /home/masa/darknet

- ライブカメラ

$ ./darknet detector demo cfg/coco.data cfg/yolov3-tiny.cfg yolov3-tiny.weights -c 0

・まずまず早い