ESP32 CPU Core情報の表示の 前に

i さんから、ESP32の入門機種として ESP32-WROOM-32は、もう古い機種になっていて、もっと新しいモジュールで、低価格で、小型のモジュールが 出てますよ。 との指摘がありました。

C3や S3の 文字が付いたモジュールです。

これは、開発ボード上に載せてあるモジュールですよね。 ハンダ付けが面倒ではと思っていましたが…。

Wi-Fiモジュール ESP32-C3-WROOM-02-N4

在庫グレード： AAA

販売コード： 117493
型番： ESP32-C3-WROOM-02-N4
発売日： 2022/12/08
メーカーカテゴリ： Espressif Systems (Shanghai) Pte. Ltd.

よくある質問(Q&A)
商品選定・製作の参考にしてください。

1個

1個 ￥310

在庫

購入数量：

かごに入れ

Wi-Fiモジュール ESP32-S3-WROOM-1-N16R8

在庫グレード： AAA

販売コード： 117256
型番： ESP32-S3-WROOM-1-N16R8
発売日： 2022/05/06
メーカーカテゴリ： Espressif Systems (Shanghai) Pte. Ltd.

履歴

よくある質問(Q&A)
商品選定・製作の参考にしてください。

1個 ￥620

在庫

購入数量：

かごに入れ

i さんは、斬新な ハンダ付けの方法を、実現されてました。 やろうと思えば、お金をかけなくても、比較的 楽にハンダ付けする方法は ある。 という アマチュア精神旺盛な方のようです。 但し 半田付け後に、ハンダ付けした線を引っ張ると、ランドを半分に切ったようなパターンが 剥がれる事が、あるらしいです。
ピン間は ハーフピッチの寸法の ようなのでさほど、難しい半田付けでは なさそうです。

その、半田付けしたリード線を、通常の 2.54 ピッチの基板につければ使いやすくなるという事です。

あと、C3、S3は CP2102のようなUSB-シリアル変換ICが 必要無いです。 ESP32-C3、-S3モジュールは USBの 信号線を 直接接続できます。

ESP32-S3は WROOM-32のCPUコア Xtensa LX6より新しい、 Xtensa LX7 クロック 240MHz が使用されています。 ROM 16MByte、 RAM 8MByte です。 価格 620円です。

ESP32-C3は、CPUコアは RISC-V で クロック 160MHz が、使用されています。
ROM 4MByte、RAM 400KByteで
価格は 310円です。

電源電圧は 3.0 ～ 3.6V なので、USBケーブルの 5Vは 直接接続出来ませんが、新品の単三電池は、1.6Vぐらいの電圧なので、単三電池2本で 動かす事も可能と思います。

この 310円の ESP32-C3でも、十分な高性能と思いますので、何かに使ってみたいですね。

価格は、やや高くなるのですが、XIAO ESP32 C3 と、XIAO ESP32 S3 というモジュールが、ありました。DIPピッチのパッドが付いているので2.54ピッチの ピンヘッダが付けられます。 が、両方とも ピン数が 14ピンというのが、割り切って使わないといけないですね。 それと、この XIAO シリーズは、高性能の アンテナが付いています。 遠隔通信距離最大 100mとなっています。

Seeed Studio XIAO ESP32C3

在庫グレード： AAA

販売コード： 117454
型番： 113991054
発売日： 2022/08/30
メーカーカテゴリ： Seeed Studio(シードスタジオ)

よくある質問(Q&A)
商品選定・製作の参考にしてください。

1個

1個 ￥940

在庫あ

購入数量：

かごに入れ

Seeed Studio XIAO ESP32S3

在庫グレード： AAA

販売コード： 118078
型番： 113991114
発売日： 2023/06/05
メーカーカテゴリ： Seeed Studio(シードスタジオ)

よくある質問(Q&A)
商品選定・製作の参考にしてください。

1個

1個 ￥1,300

在庫あ

購入数量：

かごに入れ

ESP32 CPU Core情報の表示

参照したのが、「ESP32-WROOM-32 チップ・メモリ・MACアドレス情報取得方法」の サイトです。コア情報を読み出すプログラムは、このサイトのソースを そのまま流用してます。

じゃ簡単じゃないか。と思われるでしょうがソースコードは そのままコピーなので簡単ですが、コア情報を読み出すために、ESPというクラスを 用いています。

これは、Arduino IDE上で ESP32を 扱えるようにするため、最初に行う ESP32のボートマネージャのインストールだけでは、プログラムのビルドが出来ません。

追加のボードマネージャが必要です。
でも、それだけだと うまく行かない可能性があります。

ESP32に関わる アップデートモジュール等があれば、アップデートを 一通りやっておく方がいいです。

以下のEspressif公式の Arduino core for the ESP32 インストールのページ

https://docs.espressif.com/projects/arduino-esp32/en/latest/installing.html

上記URLを ブラウザで アクセスします。

次ページで、画像で示します。

First Things First
Supported SoC's
Arduino Core Reference
Supported Operating Systems
Supported IDEs
Support
Community
Issues Reporting
First Steps
How to Install
Before Installing
Installing using Arduino IDE
Development Boards
Examples

This is the way to install Arduino-ESP32 directly from the Arduino IDE.

Note

For overview of SoC's support, take a look on Supported Soc's table where you can find if the particular chip is under stable or development release.

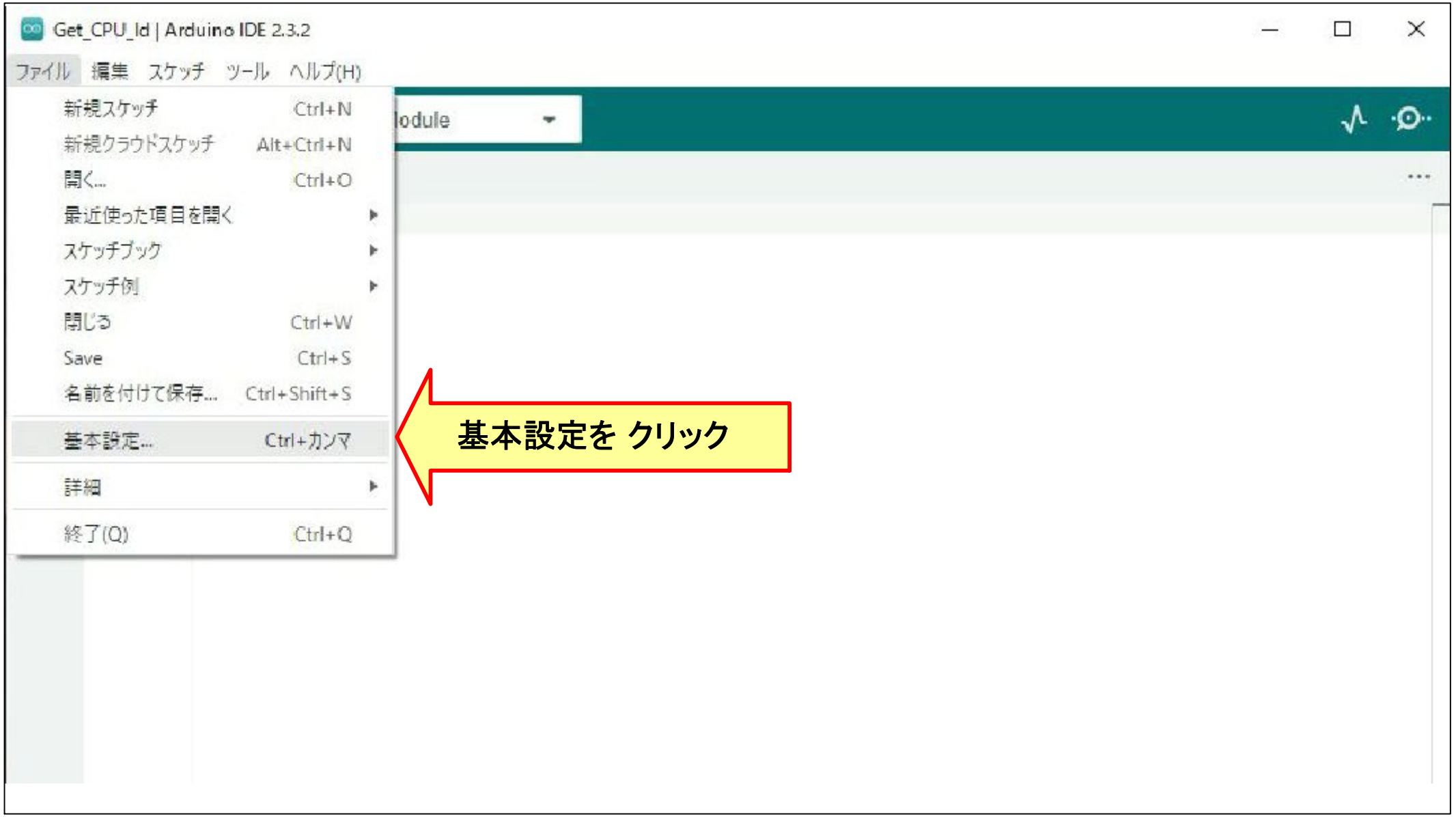
- Stable release link:

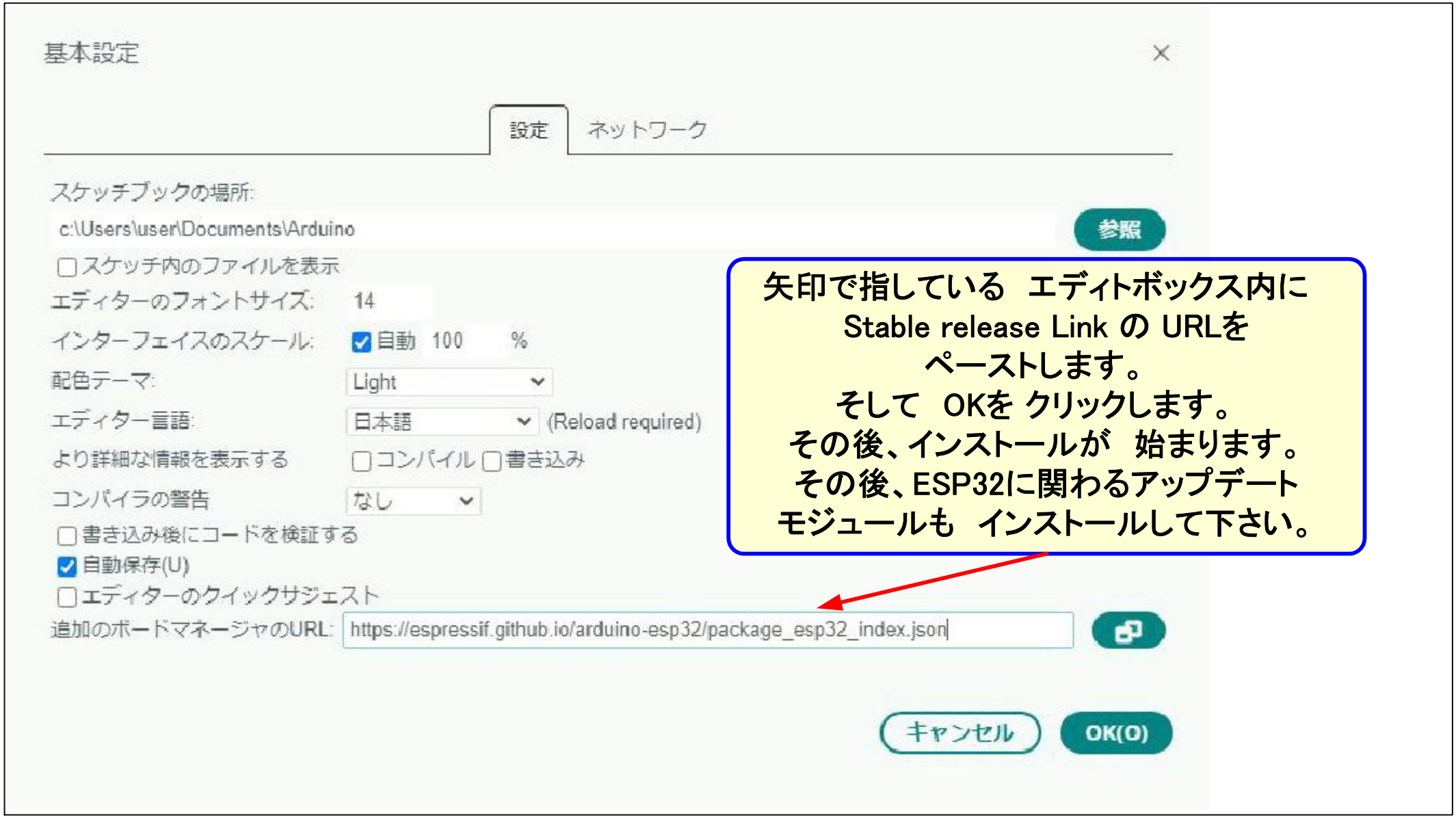
```
https://espressif.github.io/arduino-esp32/package_esp32_index.json
```

- Development release link:

```
https://espressif.github.io/arduino-esp32/package_esp32_dev_index.json
```

矢印で指している
Stable release Link 側の
URLを コピーします。


Get_CPU_Id | Arduino IDE 2.3.2
ファイル 編集 スケッチ ツール ヘルプ(H)
新規スケッチ Ctrl+N
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
スケッチ例
閉じる Ctrl+W
Save Ctrl+S
名前を付けて保存... Ctrl+Shift+S
基本設定... Ctrl+カンマ
詳細
終了(Q) Ctrl+Q
基本設定を クリック


基本設定
設定
ネットワーク
スケッチブックの場所:
c:\Users\user\Documents\Arduino
参照
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 (Reload required)
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る
自動保存(U)
エディターのクイックサジェスト
追加のボードマネージャのURL: https://espressif.github.io/arduino-esp32/package_esp32_index.json
キャンセル
OK(O)
矢印で指している エディトボックス内に
Stable release Link の URLを
ペーストします。
そして OKを クリックします。
その後、インストールが 始まります。
その後、ESP32に関わるアップデート
モジュールも インストールして下さい。

今回のプログラムソースの表示

今回のソースプログラムを get_cpu_id.ino と名前を付けましたが
setup 関数内で、全てを表示するようになっています。 loop 関数内は 空です。
殆どが、Serial.printf 関数で、表示文字列を 作成し、Arduino IDE側の シリアルモニタに
文字列を 転送する処理です。 青色のクラス及び関数は、コア情報の読み出し関数です。

```
void setup( void ) {
  Serial.begin( 115200 );  // 通信開始  伝送速度 115200 bps
  Serial.println("¥r¥n-----------------------------");

  uint64_t  chipid;     //  chipid 変数宣言
  chipid=ESP.getEfuseMac();  //The chip ID is essentially its MAC address(length: 6 bytes).
  Serial.printf("ESP32 Chip ID = %04X¥r¥n",(uint16_t)(chipid >> 32));  //print High 2 bytes
  Serial.printf("Chip Revision %d¥r¥n", ESP.getChipRevision() );

  esp_chip_info_t chip_info;  //  chip_info 変数宣言
  esp_chip_info( &chip_info );  // チップインフォ情報取り出し
  Serial.printf("Number of Core: %d¥r¥n", chip_info.cores );
  Serial.printf("CPU Frequency: %d MHz¥r¥n", ESP.getCpuFreqMHz());
  Serial.printf("Flash Chip Size = %d byte¥r¥n", ESP.getFlashChipSize());
  Serial.printf("Flash Frequency = %d Hz¥r¥n", ESP.getFlashChipSpeed());
```

```
Serial.printf("ESP-IDF version = %s¥r¥n", esp_get_idf_version());
//利用可能なヒープのサイズを取得
Serial.printf("Available Heap Size= %d¥r¥n", esp_get_free_heap_size());
//利用可能な内部ヒープのサイズを取得
Serial.printf("Available Internal Heap Size = %d¥r¥n", esp_get_free_internal_heap_size());
//これまでに利用可能だった最小ヒープを取得します
Serial.printf("Minimum Free Heap Ever Available Size = %d¥r¥n",
esp_get_minimum_free_heap_size());
Serial.println();  // 改行

uint8_t  mac0[6];  // MACアドレス格納用 変数宣言
esp_efuse_mac_get_default( mac0 );  // MACアドレス取り出し
Serial.printf("Default Mac Address = %02X:%02X:%02X:%02X:%02X:%02X¥r¥n",
mac0[0], mac0[1], mac0[2], mac0[3], mac0[4], mac0[5] );

uint8_t  mac3[6];  // MACアドレス格納用 変数宣言
esp_read_mac( mac3, ESP_MAC_WIFI_STA );  // MACアドレス取り出し
Serial.printf("[Wi-Fi Station] Mac Address = %02X:%02X:%02X:%02X:%02X:%02X¥r¥n",
mac3[0], mac3[1], mac3[2], mac3[3], mac3[4], mac3[5] );
```

```
  uint8_t  mac4[7];  // MACアドレス格納用 変数宣言
  esp_read_mac( mac4, ESP_MAC_WIFI_SOFTAP ) ;  // MACアドレス取り出し
  Serial.printf("[Wi-Fi SoftAP] Mac Address = %02X:%02X:%02X:%02X:%02X:%02X¥r¥n",
  mac4[0], mac4[1], mac4[2], mac4[3], mac4[4], mac4[5] );

  uint8_t  mac5[6];  // MACアドレス格納用 変数宣言
  esp_read_mac( mac5, ESP_MAC_BT );  // MACアドレス取り出し
  Serial.printf("[Bluetooth] Mac Address = %02X:%02X:%02X:%02X:%02X:%02X¥r¥n",
  mac5[0], mac5[1], mac5[2], mac5[3], mac5[4], mac5[5] );

  uint8_t  mac6[6];  // MACアドレス格納用 変数宣言
  esp_read_mac( mac6, ESP_MAC_ETH );  // MACアドレス取り出し
  Serial.printf("[Ethernet] Mac Address = %02X:%02X:%02X:%02X:%02X:%02X¥r¥n",
  mac6[0], mac6[1], mac6[2], mac6[3], mac6[4], mac6[5] );
}

void loop() {
}
```

ソースは、以上です。

①番基板　ESP32-WROOM-32（ 新 ）

```
----------------------------------------
ESP32 Chip ID = A008
Chip Revision 1
Number of Core: 2
CPU Frequency: 160 MHz
Flash Chip Size = 4194304 byte
Flash Frequency = 80000000 Hz
ESP-IDF version = v4.4.7-dirty
Available Heap Size= 279164
Available Internal Heap Size = 279164
Minimum Free Heap Ever Available Size = 273684

Default Mac Address = 84:0D:8E:1B:08:A0
[Wi-Fi Station] Mac Address = 84:0D:8E:1B:08:A0
[Wi-Fi SoftAP] Mac Address = 84:0D:8E:1B:08:A1
[Bluetooth] Mac Address = 84:0D:8E:1B:08:A2
[Ethernet] Mac Address = 84:0D:8E:1B:08:A3
```

②番基板　ESP32-WROOM-32（ 新 ）

```
----------------------------------------
ESP32 Chip ID = 14CF
Chip Revision 1
Number of Core: 2
CPU Frequency: 160 MHz
Flash Chip Size = 4194304 byte
Flash Frequency = 80000000 Hz
ESP-IDF version = v4.4.7-dirty
Available Heap Size= 279164
Available Internal Heap Size = 279164
Minimum Free Heap Ever Available Size = 273684

Default Mac Address = 80:7D:3A:80:CF:14
[Wi-Fi Station] Mac Address = 80:7D:3A:80:CF:14
[Wi-Fi SoftAP] Mac Address = 80:7D:3A:80:CF:15
[Bluetooth] Mac Address = 80:7D:3A:80:CF:16
[Ethernet] Mac Address = 80:7D:3A:80:CF:17
```

```
★  ESP32 WROVER Module
-------------------------------------
ESP32 Chip ID = 0CB2
Chip Revision 3
Number of Core: 2
CPU Frequency: 240 MHz
Flash Chip Size = 8388608 byte
Flash Frequency = 80000000 Hz
ESP-IDF version = v4.4.7-dirty
Available Heap Size= 4471163
Available Internal Heap Size = 279040
Minimum Free Heap Ever Available Size = 4465683

Default Mac Address = 48:E7:29:A3:B2:0C
[Wi-Fi Station] Mac Address = 48:E7:29:A3:B2:0C
[Wi-Fi SoftAP] Mac Address = 48:E7:29:A3:B2:0D
[Bluetooth] Mac Address = 48:E7:29:A3:B2:0E
[Ethernet] Mac Address = 48:E7:29:A3:B2:0F
```

前ページの ①と ②が ESP32-WROOM-32 の 2枚の基板の コア情報です。
このページの ★は、最近買った ESP32-WROVER-Eの コア情報です。

①と ②は 共に Chip Revision 1 で ★が Chip Revision 3 です。

CPUコア数は、①、②、★ 共に 2 です。

CPUクロック周波数は、①、②が 160MHz で ★が 240MHz です。

ROMサイズが、①、②が 4.194Mbyte で ★が 8.388Mbyte です。

RAM空きエリアが ①、②が 279Kbyte で ★が 4.471Mbyte です。