周辺回路の紹介の前に

前回から　百円R8Cマイコンから、ESP32マイコンに 移行しました。 毎回の事ですが、この動画を見てくれる視聴者のレベルを　どの当たりに設定するかが、難しいです。

特に　この　ESP32マイコンは　Arduino IDEでプログラム開発が出来るので、初心者よりのイメージも　あります。

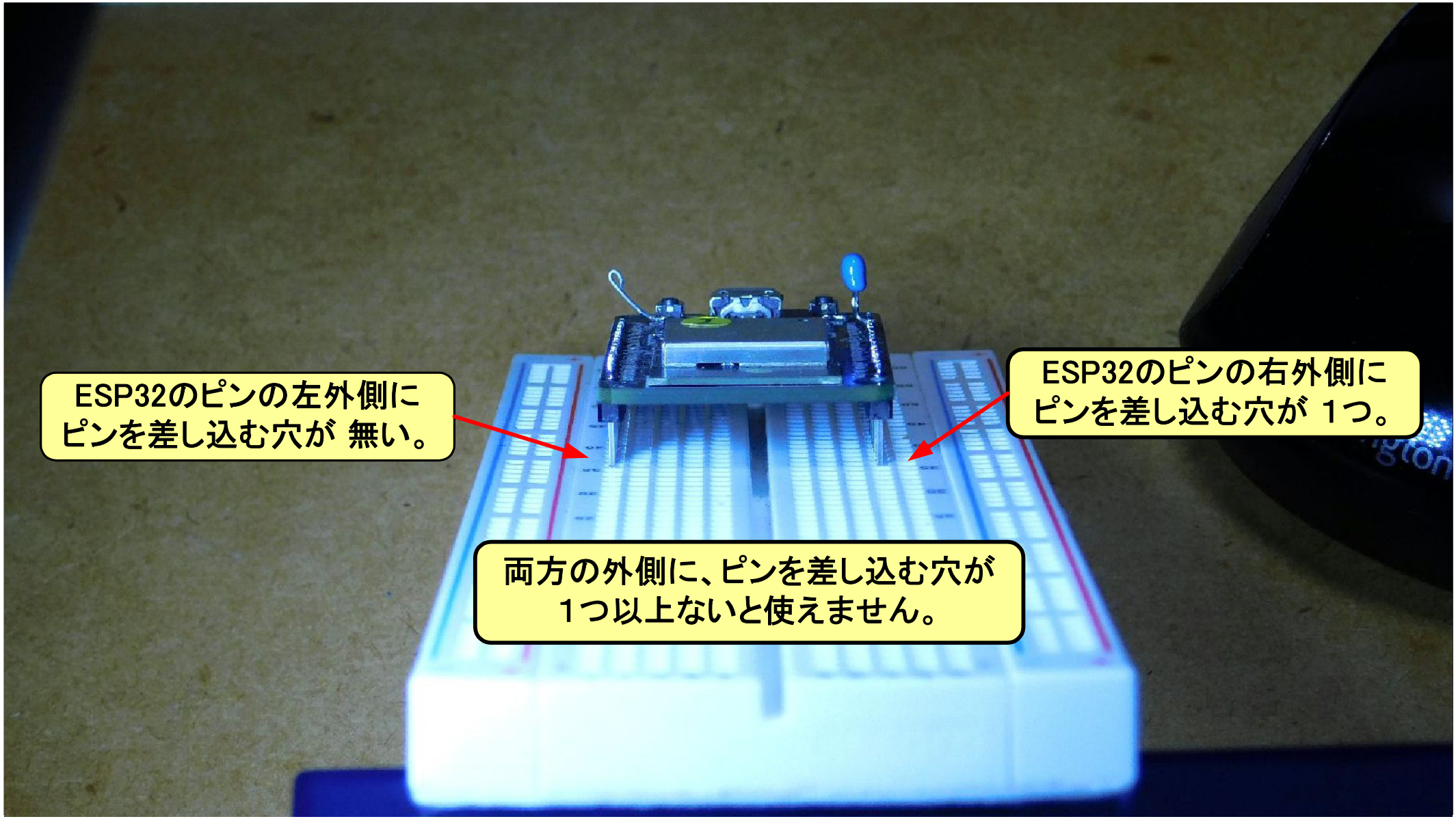
でも、ESP32マイコンは、価格の割に、並はずれた高性能なCPUを　搭載しています。
160MHzまたは 240MHzの 高速CPUクロックで、メモリ容量も大きく、更に　CPUコアを　2つ搭載しています。 ネットでサーチすると、ESP32で　RTOSや　部分的にアセンブラでチューニングする　つわものの 方もいらっしゃいます。

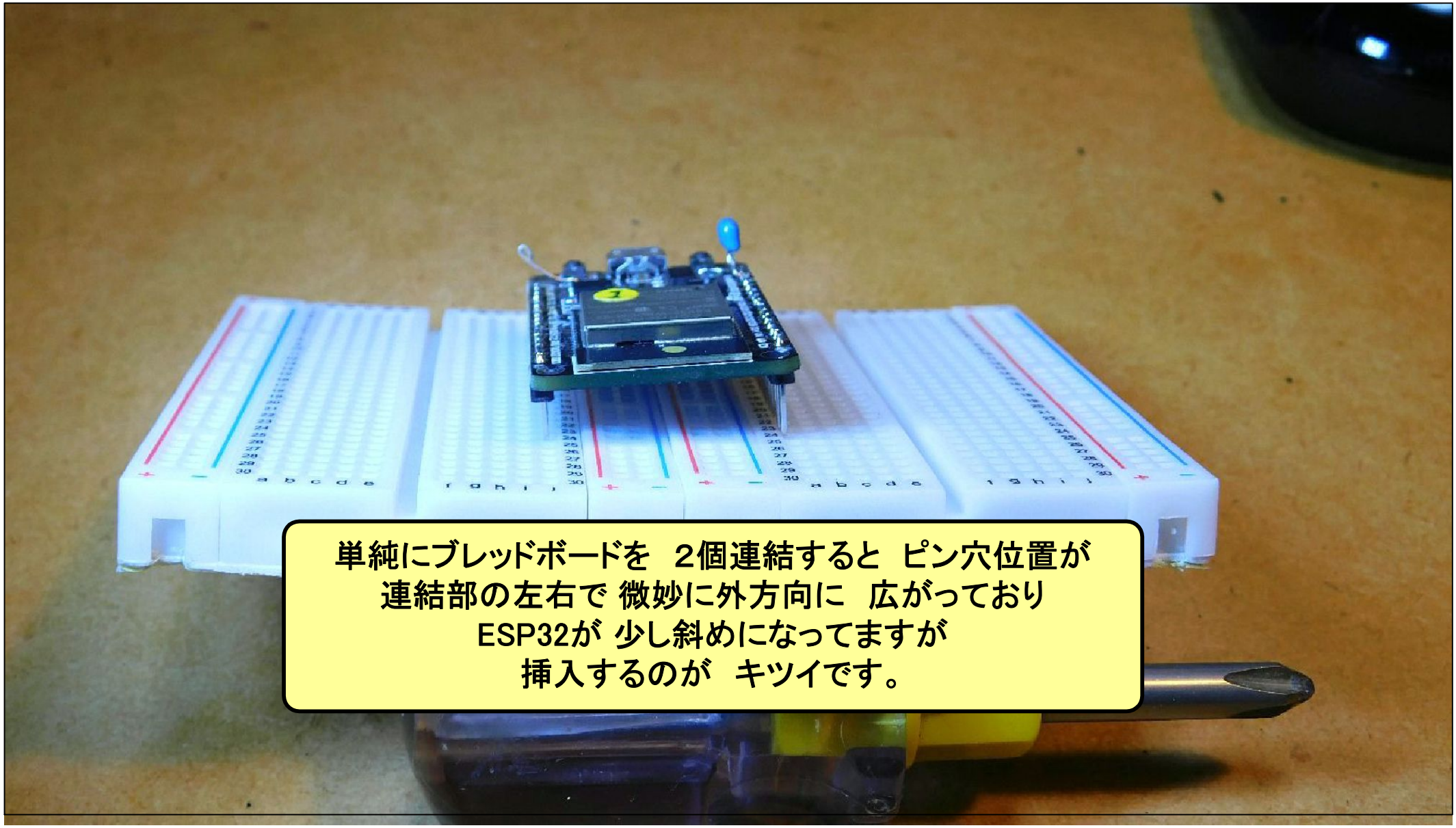
よって、ESP32に興味を 持つ方の技術的レンジが　広いのです。
で　私としては、深い内容を期待する方には申し訳ないですが、しばらくの間は、初心者 寄りの路線で　行こうと思います。

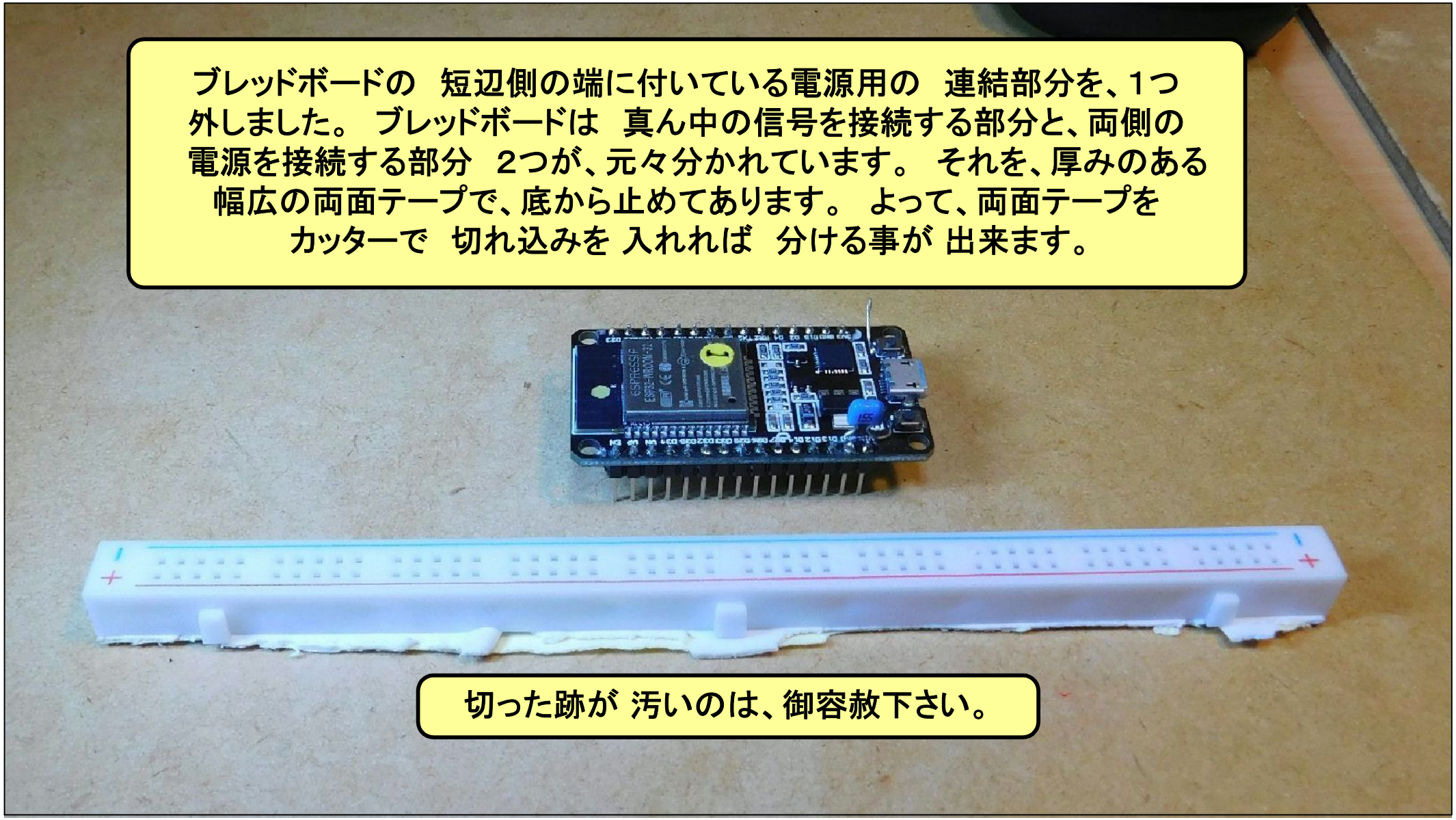
といいつつ、ついつい難しい内容を 織り込んでしまう悪い癖があるので、御容赦下さい。

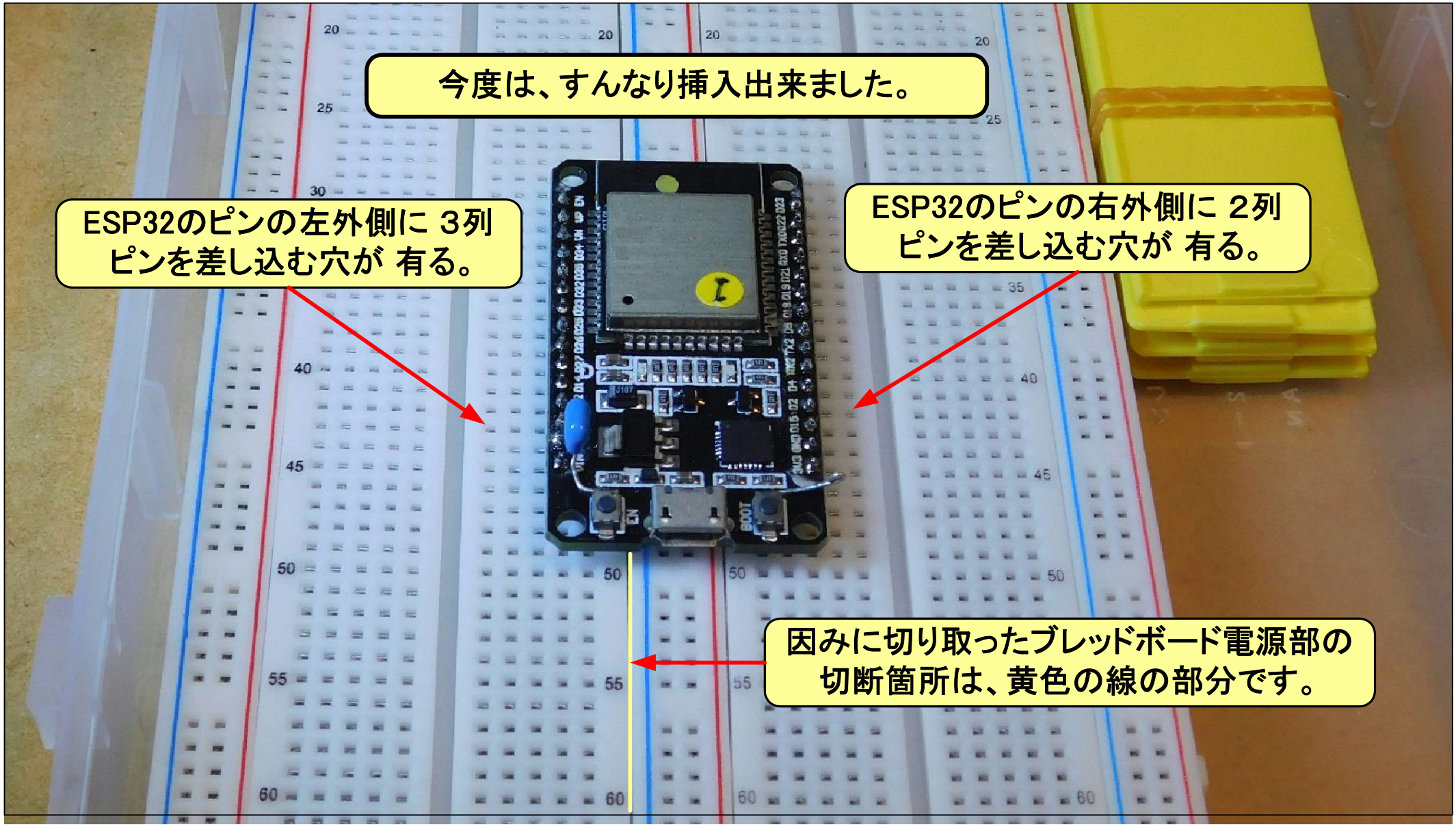
自分としては、難しい内容の事は、細かい事は分からなくても、大雑把にイメージで捉えてもらえれば結構です。 後々 分かってくる場合もあると思います。

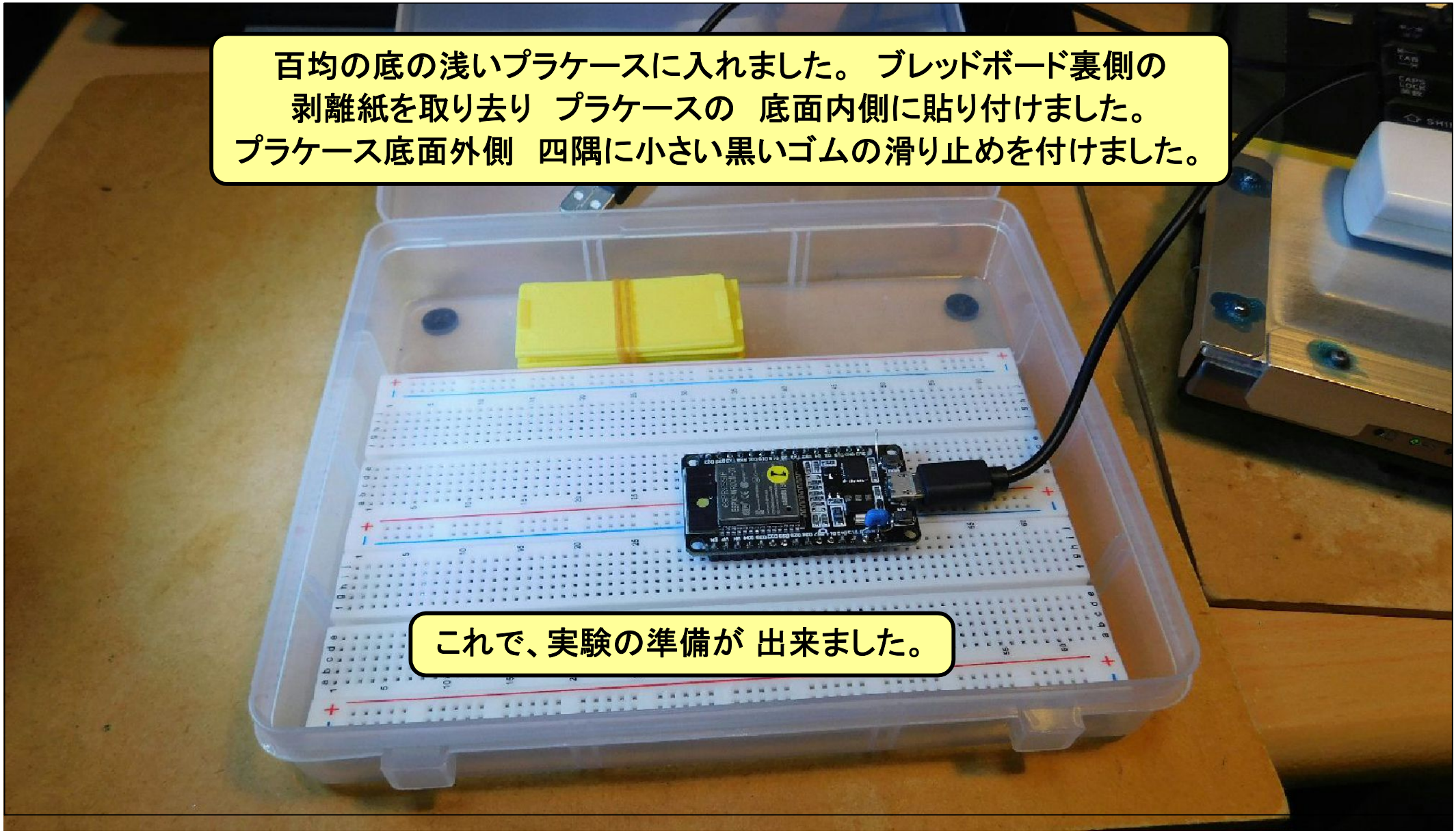
それと、ESP32で　実験を行う際に、ブレッドボードを使うつもりでいましたが、実際に　マウントすると、具合の悪い所に　気付きました。
画像で、お見せします。


ESP32のピンの左外側に
ピンを差し込む穴が 無い。
ESP32のピンの右外側に
ピンを差し込む穴が 1つ。
両方の外側に、ピンを差し込む穴が
1つ以上ないと使えません。


単純にブレッドボードを　２個連結すると　ピン穴位置が
連結部の左右で 微妙に外方向に　広がっており
ESP32が 少し斜めになってますが
挿入するのが　キツイです。


ブレッドボードの　短辺側の端に付いている電源用の　連結部分を、1つ
外しました。 ブレッドボードは　真ん中の信号を接続する部分と、両側の
電源を接続する部分　2つが、元々分かれています。 それを、厚みのある
幅広の両面テープで、底から止めてあります。 よって、両面テープを
カッターで　切れ込みを 入れれば　分ける事が 出来ます。
切った跡が 汚いのは、御容赦下さい。


今度は、すんなり挿入出来ました。
ESP32のピンの左外側に 3列
ピンを差し込む穴が 有る。
ESP32のピンの右外側に 2列
ピンを差し込む穴が 有る。
因みに切り取ったブレッドボード電源部の
切断箇所は、黄色の線の部分です。


百均の底の浅いプラケースに入れました。 ブレッドボード裏側の
剥離紙を取り去り プラケースの 底面内側に貼り付けました。
プラケース底面外側 四隅に小さい黒いゴムの滑り止めを付けました。
これで、実験の準備が 出来ました。

周辺回路の紹介の前に　CPUの紹介

実は、ESP32の CPUコアは、上海の Espressif Systems ではなくて、アメリカの Tensilica 社の Xtensa LX6 マイクロプロセッサです。 現在は アメリカの Cadence社に 買収されています。 Xtensa CPUコアは、Cadence 社で、継続的に扱われています。

Tensilica社は 半導体メーカーではなくて、IPメーカーです。CPUコアの設計図を OEM販売している会社のようです。 同様の会社にはイギリスのアーム社が、あります。

Tensilicaの 製品概要の序文を紹介します。

Xtensaプロセッサ・コア

コンフィギュラブルで拡張性が高く合成可能なテンシリカの Xtensa プロセッサは、エンベディッドのシステム・オン・チップ（SOC）に対応すべく特別に設計された、初めてのマイクロプロセッサ・アーキテクチャです。 設計者がターゲットとする SOCのアプリケーション要求に合わせて、個々にインプリメンテーションが出来るような コンフィギュラブルな アーキテクチャとして、Xtensaは 最初から設計されています。

と、書いてありました。

プロセッサアーキテクチャ：

5段パイプライン　高機能 32bit　RISC

命令セット：

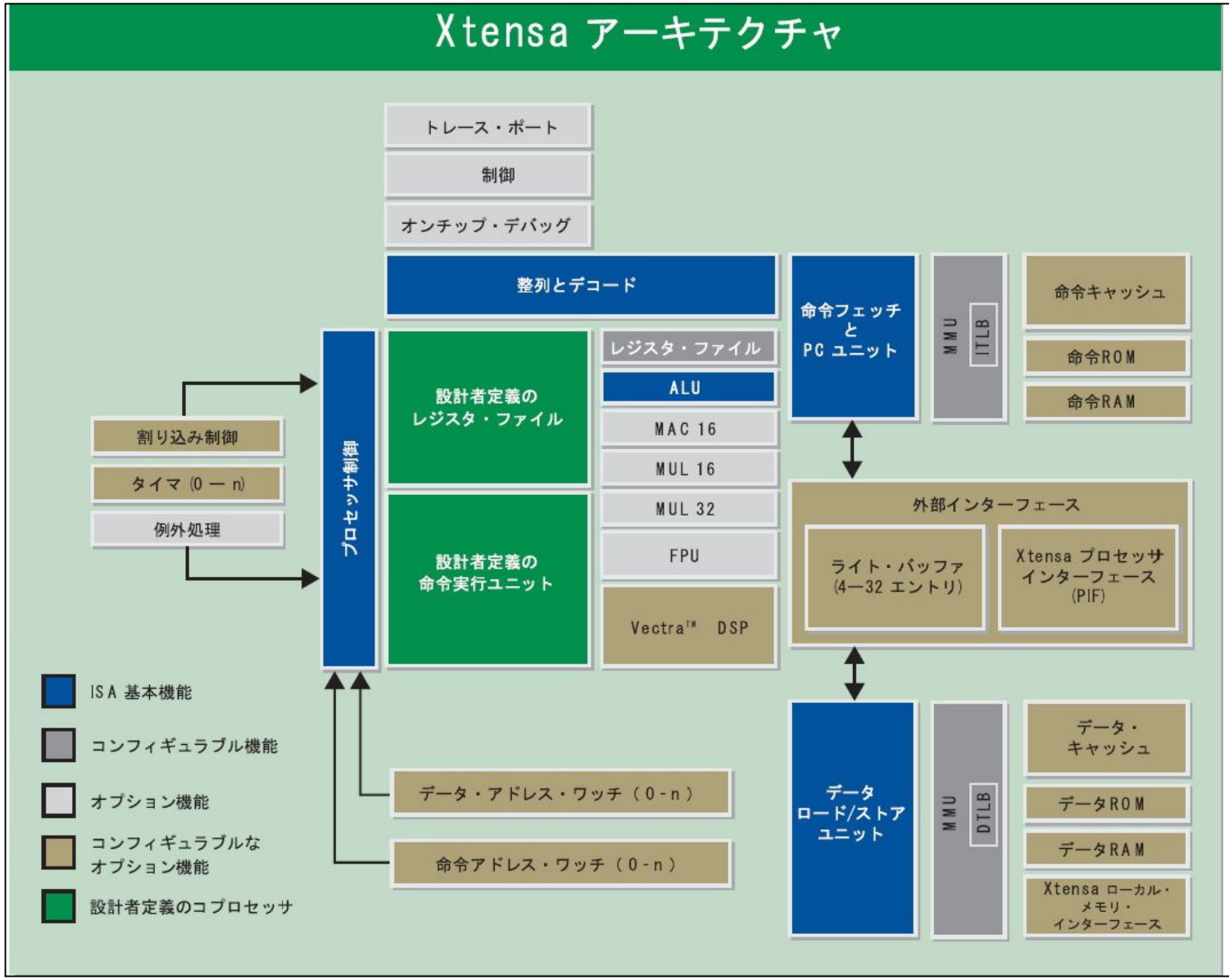
コンパクトな 16bit、24bitエンコーディング方式の Xtensa ISA（ モード切替なしで実現 ）

クロックスピード：

350MHz　@0.13μプロセス

200MHz　@0.18μプロセス

サイズ：　約　25,000ゲート（ 基本構成 ）



Xtensaアーキテクチャの図です。この図を見たとき近未来的なイメージで カルチャーショックを受けましたね。 因みに、アセンブラを扱う者として、気になるのは命令体系とアドレッシングモード、そしてレジスタ構成ですが、演算に使うレジスタが どれか分かりますか。？

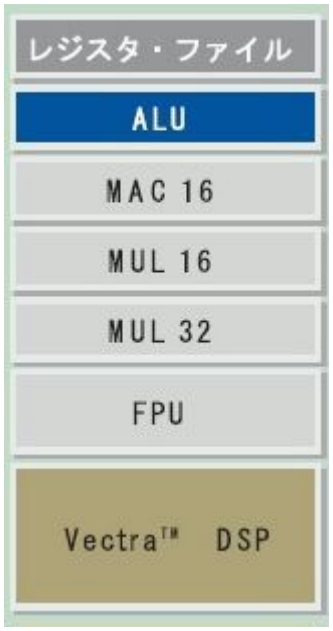
レジスタファイルと書かれた部分ですが、汎用レジスタは、ALU上の灰色のレジスタファイル部分です。

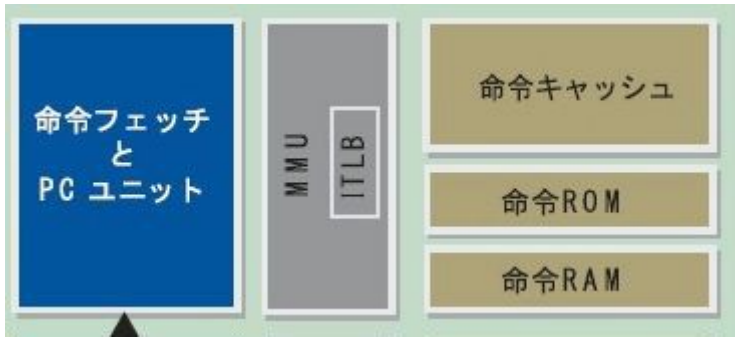
この図では、示されていませんが ESP32の汎用レジスタは レジスタファイルのブロックに アドレスレジスタという名前で、32bitのレジスタを 16本 持っています。

MAC 16は 演算用のアキュムレータと思われます。 MUL 16と MUL32は 高速の乗算器と思います。

その下の FPUは 浮動小数点演算ユニットです。

Vectra tm DSPは デジタルシグナルプロセッサです。
デジタルオーディオや、電子楽器の リアルタイムの信号処理に使用されます。





やはり、ありましたね。 命令のキャッシュ機能。命令フェッチブロックの前段に MMU(メモリ マネジメント ユニット)が、入ってます。このMMUが、ハード的に 命令キャッシュ最適化の制御を行っていると思います。
前ページの右下側に データのキャッシュ機能もありますが、ESP32では、データのキャッシュ機能は 使用してないと思われます。
だいたい実行時に ROMに データを ランダムに 書き込むこと自体、出来ません。

そして　周辺回路の紹介

周辺インタフェース:

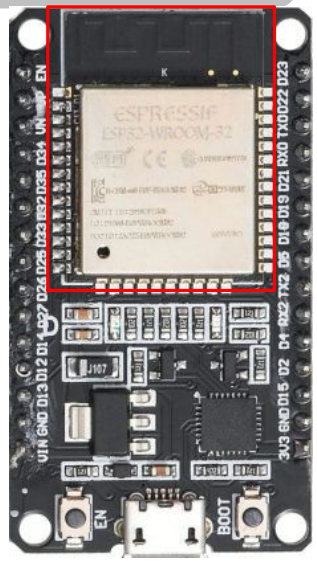
12bitの SAR ADC 18ch
8bitの DAC　2ch
10 x タッチセンサー
温度センサ（ CPUの温度測定と思われる ）
SPI x 4
I2S x 2
I2C x 2
UART x 3
SD/SDIO/MMC ホスト
スレーブ(SDIO/SPI)
イーサネット MACインタフェース DMA 及び
IEEE1588を サポート
CAN　Bus 2.0
モーター用 PWM
LED PWM　最大 16 チャネル

ホールセンサー
超 低消費電力 アナログ プリアンプ
通常の デジタル入出力　GPIO が あります。

以上、多くの周辺インタフェースをサポートしています。 但し、チップの　足ピン数が　49ピンで基板モジュールとしては、38ピン、30ピンで　ピン数が少ないので　内蔵の周辺機器を　一度に多数使うのは、限界があると 思われます。

それと、足ピンの事を調べて行くうちに、ピンはあるけど、使えないピンが ある事が、分かりました。 それは　一部の SPIのインタフェースで、CPUチップ外の　SPIの フラッシュROMに 接続されているピンです。これは　MMUに管理される、ROMのキャッシュを　SRAM上に 設定されているので、MMUにより　いつアクセスされるか分からないので 使えません。次ページで 説明します。

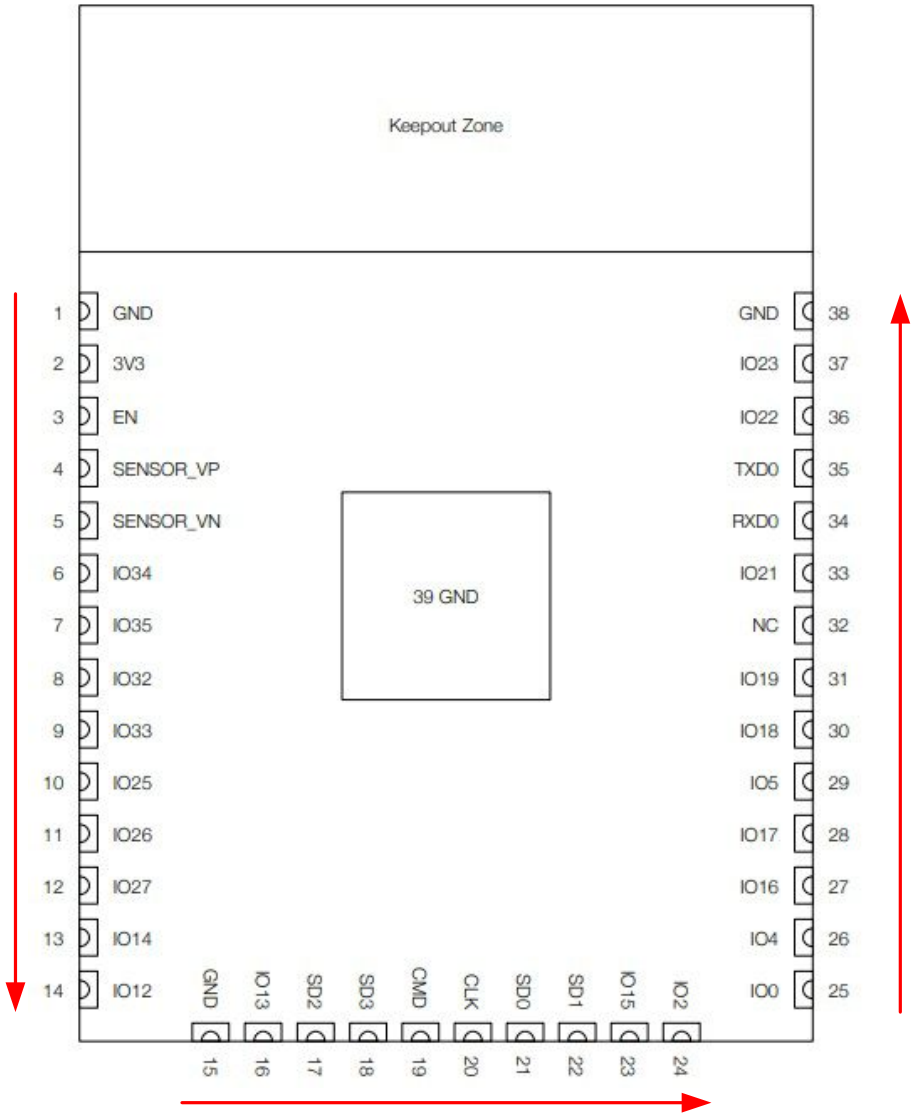
そもそも ESP32とは、どれなのか。？



通常、ESP32の基板というと
右のような画像を想像しますが
実は 右のような基板は、開発
キット基板という事のようです。

Espressif Systems が ESP32と
表現するモジュールは、右の画像
で 赤線で囲った部分です。

因みに Espressif Systems社が
ESP32の ピンアサインの図として
出しているのが、右の図です。



右の図では、左上から、ピン番号が 1番で、下に降りる方向で 1～14 番を 割り付けてあります。
そして、右方向に 15 ～ 24 番です。
次に、上に上がり 25 ～ 38 番を割り付けてあります。 この番号で ピンの機能を 説明してあります。

Name	No.	Type	Function
GND	1	P	Ground
3V3	2	P	Power supply
EN	3	I	Module-enable signal. Active high.
SENSOR_VP	4	I	GPIO36, ADC1_CH0, RTC_GPIO0
SENSOR_VN	5	I	GPIO39, ADC1_CH3, RTC_GPIO3
IO34	6	I	GPIO34, ADC1_CH6, RTC_GPIO4
IO35	7	I	GPIO35, ADC1_CH7, RTC_GPIO5
IO32	8	I/O	GPIO32, XTAL_32K_P (32.768kHz crystal oscillator input), ADC1_CH4, TOUCH9, RTC_GPIO9
IO33	9	I/O	GPIO33, XTAL_32K_N (32.768kHz crystal oscillator output), ADC1_CH5, TOUCH8, RTC_GPIO8
IO25	10	I/O	GPIO25, DAC_1, ADC2_CH8, RTC_GPIO6, EMAC_RXD0
IO26	11	I/O	GPIO26, DAC_2, ADC2_CH9, RTC_GPIO7, EMAC_RXD1
IO27	12	I/O	GPIO27, ADC2_CH7, TOUCH7, RTC_GPIO17, EMAC_RX_DV
IO14	13	I/O	GPIO14, ADC2_CH6, TOUCH6, RTC_GPIO16, MTMS, HSPICLK, HS2_CLK, SD_CLK, EMAC_TXD2
IO12	14	I/O	GPIO12, ADC2_CH5, TOUCH5, RTC_GPIO15, MTDI, HSPIQ, HS2_DATA2, SD_DATA2, EMAC_TXD3
GND	15	P	Ground
IO13	16	I/O	GPIO13, ADC2_CH4, TOUCH4, RTC_GPIO14, MTCK, HSPID, HS2_DATA3, SD_DATA3, EMAC_RX_ER
SHD/SD2*	17	I/O	GPIO9, SD_DATA2, SPIHD, HS1_DATA2, U1RXD
SWP/SD3*	18	I/O	GPIO10, SD_DATA3, SPIWP, HS1_DATA3, U1TXD
SCS/CMD*	19	I/O	GPIO11, SD_CMD, SPICS0, HS1_CMD, U1RTS

この部分が　SPI ROMアクセスで使用するピンで、使用しないで下さい。というピンです。

SCK/CLK*	20	I/O	GPIO6, SD_CLK, SPICLK, HS1_CLK, U1CTS
SDO/SD0*	21	I/O	GPIO7, SD_DATA0, SPIQ, HS1_DATA0, U2RTS
SDI/SD1*	22	I/O	GPIO8, SD_DATA1, SPID, HS1_DATA1, U2CTS
IO15	23	I/O	GPIO15, ADC2_CH3, TOUCH3, MTDO, HSPICS0, RTC_GPIO13, HS2_CMD, SD_CMD, EMAC_RXD3
IO2	24	I/O	GPIO2, ADC2_CH2, TOUCH2, RTC_GPIO12, HSPIWP, HS2_DATA0, SD_DATA0
IO0	25	I/O	GPIO0, ADC2_CH1, TOUCH1, RTC_GPIO11, CLK_OUT1, EMAC_TX_CLK
IO4	26	I/O	GPIO4, ADC2_CH0, TOUCH0, RTC_GPIO10, HSPIHD, HS2_DATA1, SD_DATA1, EMAC_TX_ER
IO16	27	I/O	GPIO16, HS1_DATA4, U2RXD, EMAC_CLK_OUT
IO17	28	I/O	GPIO17, HS1_DATA5, U2TXD, EMAC_CLK_OUT_180
IO5	29	I/O	GPIO5, VSPICS0, HS1_DATA6, EMAC_RX_CLK
IO18	30	I/O	GPIO18, VSPICLK, HS1_DATA7
IO19	31	I/O	GPIO19, VSPIQ, U0CTS, EMAC_TXD0
NC	32	-	-
IO21	33	I/O	GPIO21, VSPIHD, EMAC_TX_EN
RXD0	34	I/O	GPIO3, U0RXD, CLK_OUT2
TXD0	35	I/O	GPIO1, U0TXD, CLK_OUT3, EMAC_RXD2
IO22	36	I/O	GPIO22, VSPIWP, U0RTS, EMAC_TXD1
IO23	37	I/O	GPIO23, VSPID, HS1_STROBE
GND	38	P	Ground
GND	39	P	Ground

この部分が　SPI ROMアクセスで使用するピンで、使用しないで下さい。というピンです。

Espressif Systems社が　示す ESP32のピンアサインの図で、17 ～ 22 番のピン（GPIO9、GPIO10、GPIO11、GPIO6、GPIO7、GPIO8 ）が　使用できない事になります。

因みに、周りのピンが　38ピンあるので、ESP32　38ピンの開発ボードで ピンアサインが、どうなっているか確認したところ
信号の並びは、近い感じは　しますが
同じでは　ありませんでした。

実際の開発ボードのピンの ピンアサイン

上の基板ピンのシルク印刷ですが、VINは USBケーブルからの 5Vです。 GNDは そのまま グランドです。
D13、D12、D14、D27、D25、D33、D32、D35、D34 は GPIOの bit番号です。
D13 ～ D34 は GPIO13 ～ GPIO34 に なります。
VN、VP は 何かセンサーを接続する端子のようですが GPIOとしても使えるようです。
VP=GPIO36、VN=GPIO39 の 様です。
ENは リセット信号です。

A/Dコンバータのピンを 探すとき

ESP32　ピン ファンクション一覧の一部

SENSOR_VP	4	I	GPIO36, ADC1_CH0, RTC_GPIO0
SENSOR_VN	5	I	GPIO39, ADC1_CH3, RTC_GPIO3
IO34	6	I	GPIO34, ADC1_CH6, RTC_GPIO4
IO35	7	I	GPIO35, ADC1_CH7, RTC_GPIO5
IO32	8	I/O	GPIO32, XTAL_32K_P (32.768kHz crystal oscillator input), ADC1_CH4, TOUCH9
IO33	9	I/O	GPIO33, XTAL_32K_N (32.768kHz crystal oscillator output), ADC1_CH5, TOUCH8

ESP32　ピン ファンクション一覧から、ADC1の ピンを
探し出す例です。 何故か　CH1と　CH2が　無いです。

この表で、分かる事は

基板ピン シルク表示：　VP　　ADC1_CH0
基板ピン シルク表示：　VN　　ADC1_CH3
基板ピン シルク表示：　D34　　ADC1_CH6
基板ピン シルク表示：　D35　　ADC1_CH7
基板ピン シルク表示：　D32　　ADC1_CH4
基板ピン シルク表示：　D33　　ADC1_CH5　と、なります。

目的の　機能のピンの探し方は
理解 頂けましたでしょうか。？

内蔵周辺回路の紹介

12bitの SAR ADC 18ch

ADC1と ADC2に分かれているようですが、計 18チャネルあります。 SARというのは、組込みマイコン内の ADCとしては、標準的な逐次比較型の ADコンバータです。

ADC1が 0ch ～ 7ch で、ADC2が 0ch ～9ch です。 足すと 18chですが、ADC1側の ch1と ch2 は、足ピンが出てないので、内部で CPU の 温度測定と、もう一つ何かに使われていると思われます。 12bitなので、0 ～ 3.3V が 0 ～ 4095 の 量子化数に変換されます。

8bitの DAC 2ch

DACは ADCの逆の デジタル アナログ変換です。 分解能 8bitなので 量子化数 0 ～ 255 を 0 ～ 3.3V に 変換します。 2チャネルあります。12bitの ADCに比べると 分解能が ちょっと荒いですね。

10 x タッチセンサー

タッチセンサーの入力が 10 あるのでしょう。タッチセンサーは、液晶表示器の前に付けて、液晶に表示されているボタンを 押すとかする用途に使用するものと思われますが、申し訳ありませんが、私は使った事が無いので詳しい事は、分かりません。

温度センサ

一つは、CPUの温度測定と思われます。

SPI x 4

SPIが　4チャネルあるのでしょうか。
そのうち、一つは　MMU管理下の　SPIの ROM と思われるので、自由に使用できるのは、3チャネルと　思われます。

I2S x 2

　I2Sの インタフェースが　2チャネルあります。
I2Sは　デジタルオーディオ用の　高分解能 DAコンバータの　デジタル側のインタフェースです。 bitクロックと　Wordクロックと　bitシリアルのデータの 3本のデータで構成されます。
　データは　32bitのデータが 2つ(L と R)で 1サンプルです。

I2C x 2

　I2Cのインタフェースは、SCKと　SDAの　2線式の　汎用デジタルインタフェースです。

UART x 3

　UARTは　マイコンでよく使用する調歩同期のシリアル通信インタフェースです。 それが、3チャネル実装されてます。 そのうち 1本が、USB シリアルICに接続されています。

Arduino IDEでプログラムする際に Arduino IDE の シリアルモニタに 文字列を表示する場合は Serial という Objectを 使用します。

SD/SDIO/MMC ホスト

　SDカードのアクセスなので　SPIか　SDカード専用のインタフェースで　SDカードを　アクセスします。 ESP32が　マスタ側に なります。

スレーブ(SDIO/SPI)

　あまり使用する事はないと思いますが、スレーブ側というか、SDカードになりすます機能という事でしょうか。？

イーサネット MACインタフェース DMA 及び IEEE1588を サポート

イーサネットという事は、有線LANを サポートする機能がある。 ということでしょう。 MACアドレスも 書き込んであるので、物理層があれば使えそうな気もしますが、どうやって繋ぐかが、問題になりそうです。

CAN Bus 2.0

CAN Busは ドイツの電動工具メーカーのBOSHが 作成した 通信規格で 元々は 自動車専用の Car Area Network でしたが、その後 車以外でも使えるように Control Area Network に変ったようです。 車のイグニッションノイズの多い環境でも安定して使えるようにパケットサイズが 小さく取り決められています。

モーター用 PWM

どのようなモーターを 想定しているかで、PWMの出し方が 変わってくると思いますがACサーボモーターも含め、三相交流的なモーターを 想定しているのではと 思います。

LED PWM 最大 16 チャネル

これは、単純に 直流を ON、OFFする PWM機能ではないかと思います。

それが、16チャネルあるという事です

ホールセンサー

ホール素子は、磁気センサーですが、方位磁石の用途で使われたりします。 交流的な 磁気はコイルを使用したセンサでもセンシング出来ますが、直流的な磁気はホール素子でないと検出出来ません。

超 低消費電力 アナログ プリアンプ
これは、センサーの入力電圧が、低い時に
信号を増幅するために使用するのでしょうが
超 低消費電力であって、超 低ノイズでは無い
ので、かなり低い電圧のセンサを計るのは、
S/Nが悪くて、多分 使えないと思われます。

私としては、センサ信号にアンプを 入れるの
であれば、OPAMPを使う方が 自由度が高い
ので OPAMPを使うと思います。 極端な例で
すが マイクロボルトオーダーの 熱電対の場
合は 多少高価になりますが、インスツルメン
テーションアンプを 使う事になります。

ESP32の 内蔵周辺回路の紹介は
これで、終りです。 次は、実験に 移ります。

Serial_Send.ino

```
1   static  int ctr;
2
3   void setup() {
4     ctr = 0;
5     Serial.begin( 115200 );
6
7     Serial.println( "" );
8     Serial.println( "* Hello, welcome to the world of ESP32." );
9   }
10
11  void loop() {
12    char  tx[80];
13
14    delay( 1000 );
15    ctr++;
16    sprintf( tx, "Count = %d", ctr );
17    Serial.println( tx );
18  }
```

LED_blinking.ino

```
1   #define LED_R  2  // 赤LED
2   #define LED_Y  4  // 黄色LED
3   static  char  sw; // 点滅切替フラグ
4
5   void setup() {
6     pinMode( LED_R, OUTPUT );
7     pinMode( LED_Y, OUTPUT );
8     digitalWrite( LED_R, HIGH );
9     digitalWrite( LED_Y, HIGH );
10  }
11
```

LED_blinking.ino

```
11
12  void loop() {
13    delay( 500 );
14    sw++;
15    sw = sw & 0x01;
16    if( sw == 0 )
17    {
18      digitalWrite( LED_R, LOW );
19      digitalWrite( LED_Y, HIGH );
20    }
21    else
22    {
23      digitalWrite( LED_R, HIGH );
24      digitalWrite( LED_Y, LOW );
25    }
26  }
```