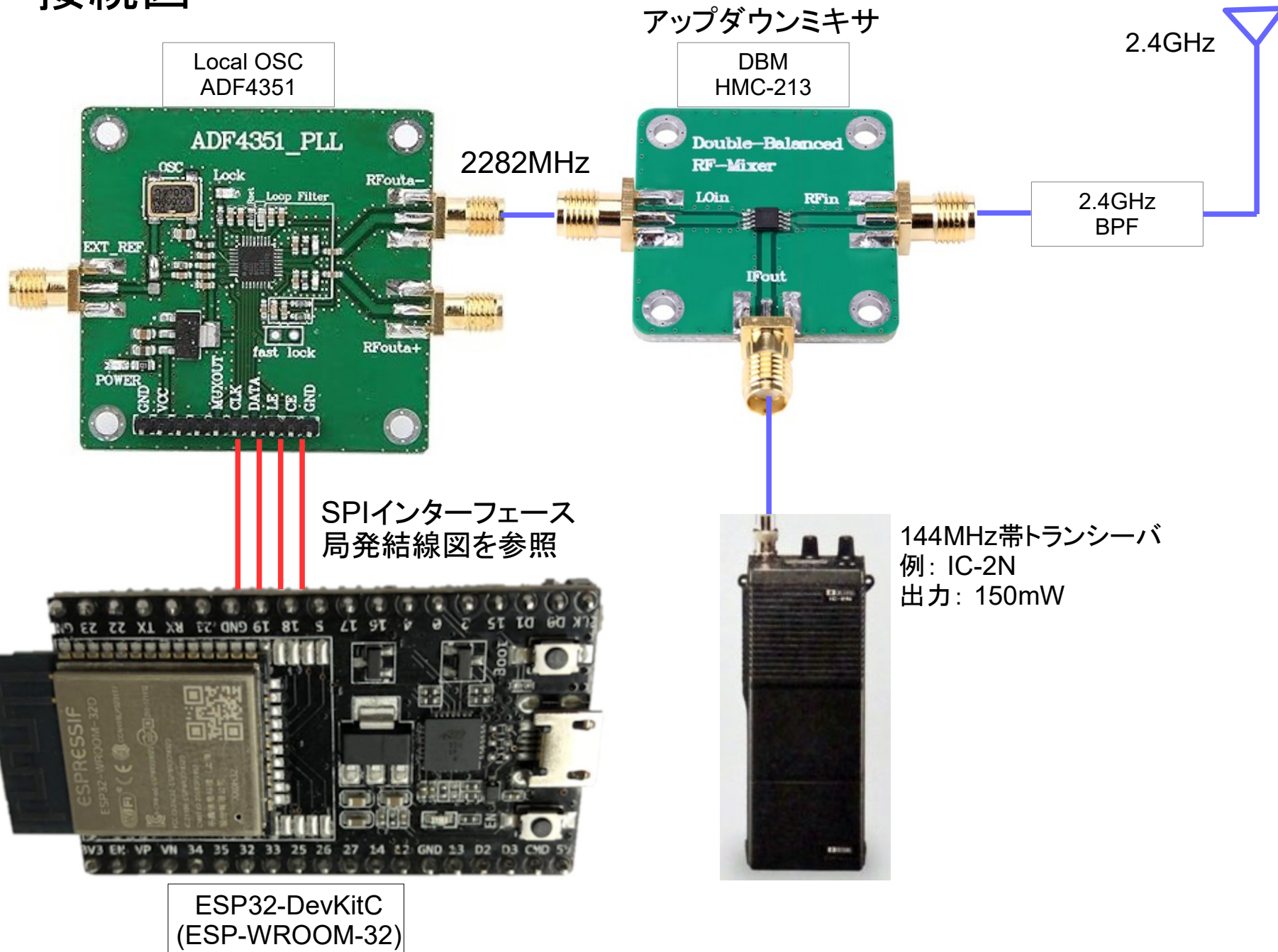


## 2.4GHz簡易トランシーバの実験

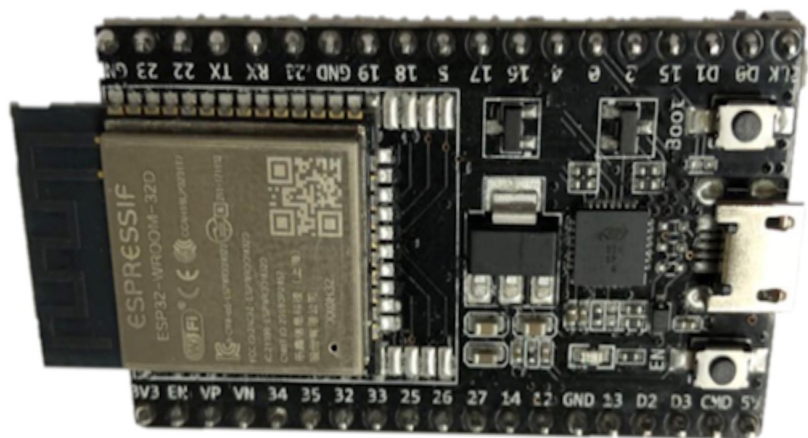
従来、マイクロ波用のトランスバータを製作する場合、局発部に多段にわたる逓倍回路が必要となります。これが自作を困難にしていると思います。ここでは、ADF4351\_PLL基板とマイコン基板を使用して手軽に局発部（マイクロ波発振）を作り、双方向ミキサーを通して簡単に2.4GHzトランシーバを実験してみます。

1. )2.4GHz 簡易トランシーバ 接続図
2. )局発(Local OSC) - 1 PLL基板とマイコン基板について
3. )局発(Local OSC) - 2 ADF4351\_PLL基板とESP32結線図
4. )局発(Local OSC) - 3 周波数(レジスタ値)決定ツール
5. )局発(Local OSC) - 4 ソースプログラム①
6. )局発(Local OSC) - 5 ソースプログラム②
7. )Arduino IDE プログラム統合開発環境

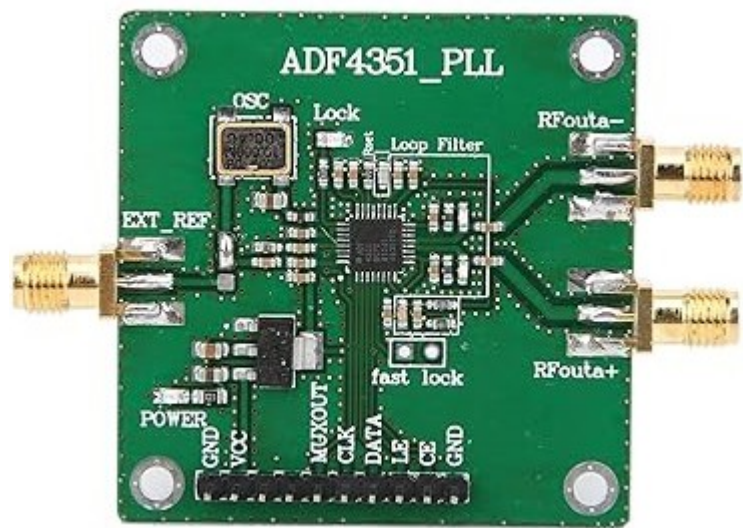
# 2.4GHz 簡易トランシーバ 接続図



## 局発(Local OSC) – 1 使用したPLL基板とマイコン基板について



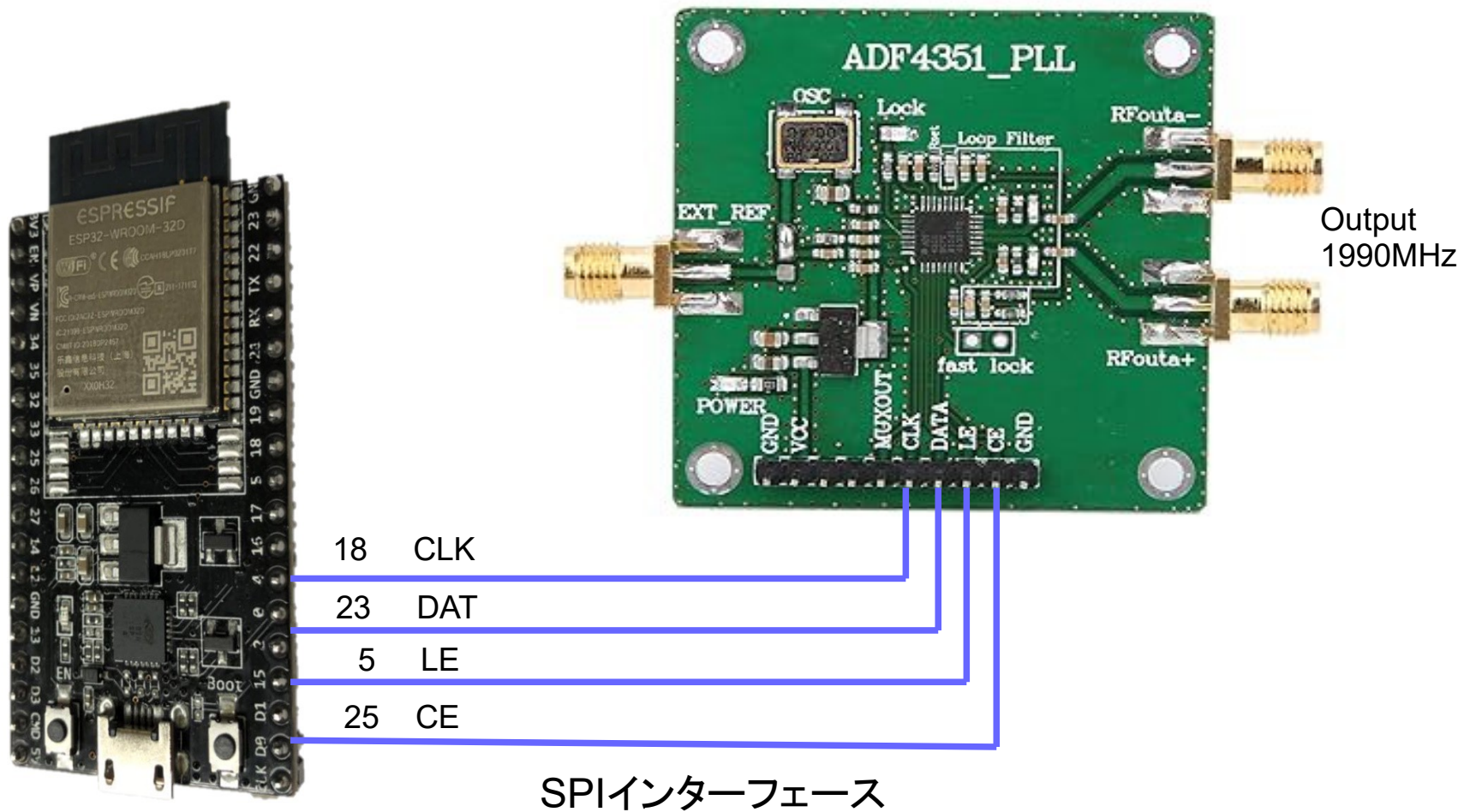
ESP32-DevKitCはESP-WROOM-32と呼ばれるマイコンを搭載したワイヤレスモジュールです。WiFiとBluetoothを搭載したMCUで、動作温度が $-40^{\circ}\text{C}$ から $+125^{\circ}\text{C}$ まで安定した動作が可能な堅牢な設計、超低消費電力設計、さまざまなモジュールを内蔵した高度な統一性を意識した設計などの特徴があります。ESP32マイコンはデュアルコア、動作クロックは240MHz、RAMは520KB、フラッシュメモリは64MBと優れたスペックとなっており、マイコンとしては十分すぎるほどの性能です。



ADF4351は、外部のループ・フィルタと外部からのリファレンス周波数を使うことによって、フラクショナルN(分数分周型)またはインテジャーN(整数分周型)のPLL周波数シンセサイザを実現することができます。ADF4351は、2200MHz~4400MHzの基本出力周波数としての電圧制御発振器(VCO)を内蔵しています。さらに、1/2/4/8/16/32または64の分周回路は、35 MHzまで低いRF出力周波数を発生させることができます。全ての内蔵レジスタの制御は、簡単な3線インターフェースを介して行われます。このデバイスは、3.0V~3.6Vの電源範囲で動作します。



# 局発(Local OSC) – 2 ADF4351\_PLL基板とESP32結線図



ESP32-DevKitC  
(ESP-WROOM-32)

# 局発(Local OSC) – 3 周波数(レジスタ値)設定ツール 出力周波数2282MHz

ADF4351のレジスタ値決定ツール(ADF435x software)。  
レジスタのR5からR0まで順次書き込みます。

The screenshot displays the 'Analog Devices ADF435x Software' application window. The interface is divided into several sections:

- RF Settings:** Includes fields for Output and VCO frequencies (both set to 2282 MHz), channel spacing (10 kHz), output divider (1), reference frequency (100 MHz), R counter (4), PFD frequency (25 MHz), prescaler (8/9), feedback signal (Fundamenta), and phase adjust (0. Off).
- Register 2:** Configures Low Noise/Spur Mode (Low noise mod), Muxout (3-state output), Double buff (Disabled), Charge pump current (2.50), LDF (FRAC-N), LDP (10 ns), PD Polarity (Positive), Powerdown (Disabled), CP 3-state (Disabled), and Counter reset (Disabled).
- Register 3:** Configures Band Select Clock Mode (Low), Charge Cancellation (Disabled), Clock Divider Value (150), CLK Div Mode (Clock Divider Off), ABP (6 ns (FRAC-N)), and CSR (Disabled).
- Register 4:** Configures VCO Powerdown (Disabled), MTL D (Disabled), Aux Output Select (Divided), Aux Output Enable (0. Disabled), Aux Output Power (-4 dBm), RF Output Enable (1. Enabled), RF Output Power (+5 dBm), and Band Select Clock (Auto set, Divisor 200, Freq 125.000 kHz).
- Register 5:** Configures LD Pin Mode (Digital Lock Detect).

At the bottom, a 'Registers' section shows a sequence of register addresses: 0x 2D8038, 0x 80080C9, 0x 10E42, 0x 4B3, 0x 8C803C, and 0x 580005, each with a corresponding 'Write' button. A 'Write All Registers' button is also present.

The status bar at the bottom left indicates 'No device connected'. The bottom right corner shows 'Device in use ADF4351', 'Software version 4.5.0', and the Analog Devices logo.

# 局発(Local OSC) – 4 ソースプログラム①

<https://github.com/Ryushane/ADF4351-ESP32/tree/master> から  
ライブラリをダウンロードし、ADF4351.cpp、ADF4351.h を  
下記のソースファイルと同じディレクトリに配置します。

```
////////////////////////////////////
//   ADF4351 RF Generator           /
//   Spot-OSC-2282MHz.ino          /
//   2023-2-16 - 2024-11-11   JA4CXX /
//// ADF4351 //////////////////////////////////
//   MCU-Pin      ADF4351 Board-Pin
//   IO = 18      CLK
//   IO = 23      DAT
//   IO = 5  LE
//   IO = 25      CE
//-----
//   Board Power +5V
//// SSD1306 //////////////////////////////////
//   MCU-Pin      SSD1306 Board-Pin
//   IO = 21      SDA
//   IO = 22      SCL
//-----
//   Board Power +3.3V
////////////////////////////////////
#include "ADF4351.h"
#include <SPI.h>
#include "SSD1306.h"//ディスプレイ用ライブラリを読み込み

#define clock 18
#define data 23
#define LE 5
#define CE 25

ADF4351  adf4351(clock,data,LE,CE); // declares object PLL of type ADF4351
SSD1306  display(0x3c, 21, 22); //SSD1306インスタンスの作成(I2Cアドレス,SDA,SCL)
```

## 局発(Local OSC) – 5 ソースプログラム②

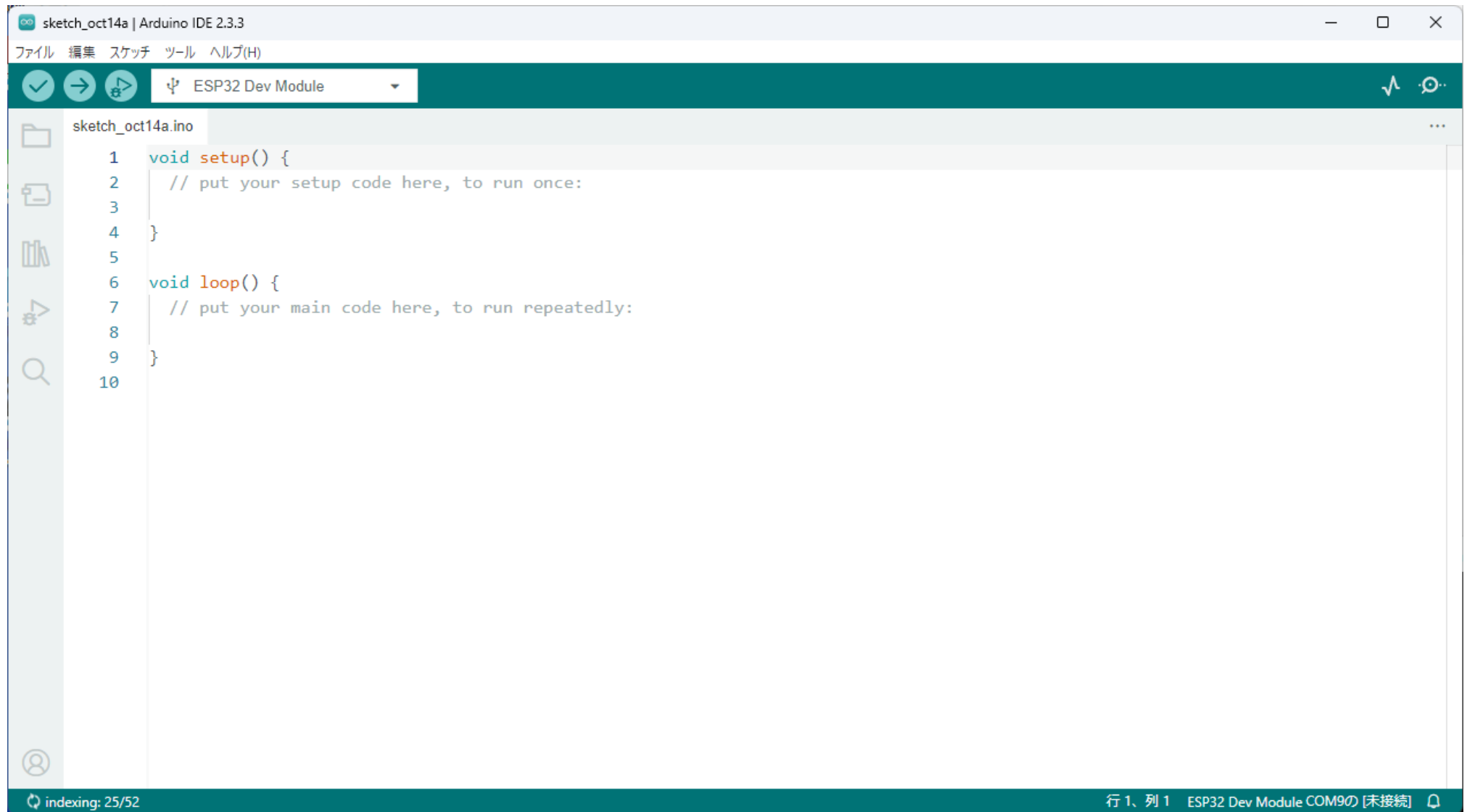
```
void setup() {
  Serial.begin(115200);
  Serial.println("Program started!");
  adf4351.begin();

  display.init(); //ディスプレイを初期化
  display.setFont(ArialMT_Plain_16); //フォントを設定 ArialMT_Plain_10, ArialMT_Plain_16, ArialMT_Plain_24
  display.drawString(0, 0, "Spot Oscillator"); // (0,0)の位置にWiFi connectedを表示
  display.drawString(0, 48, "JA4CXX");

  //発振周波数(2282MHz) <--- 2427MHz - 2282MHz = 145MHz
  adf4351.WriteRegister(0x00580005); // -R5- default value LD working mode
  adf4351.WriteRegister(0x8C803C); // -R4- output divider = auto // band select clock divider
  adf4351.WriteRegister(0x000004B3); // -R3- default Antibacklash pulse width
  adf4351.WriteRegister(0x00010E42); // -R2- Low noise mode, R Counter=4
  adf4351.WriteRegister(0x80080C9); // -R1- prescaler=8/9, MOD=2
  adf4351.WriteRegister(0x2D8038); // -R0- int = 91 frac = 7
  Serial.println("All registers have been written!");
  display.drawString(0, 24, "Freq: 2282MHz");
  Serial.println("Freq: 2282MHz");
  display.display(); //指定された情報を描画
}

void loop() {
  // put your main code here, to run repeatedly:
}
```

# Arduino IDE プログラム統合開発環境



起動時



# Arduino IDE プログラム統合開発環境 — ボードとポートを選択

The screenshot shows the Arduino IDE 2.3.3 interface. The top menu bar includes 'ファイル', '編集', 'スケッチ', 'ツール', and 'ヘルプ(H)'. The 'Tools' menu is open, showing 'Arduino Nano' selected, with a dropdown menu listing 'Arduino Nano COM9' and '不明 COM4'. A red box highlights the option '他のボードとポートを選択...' (Select other board and port...). A dialog box titled '他のボードとポートを選択' (Select other board and port) is open, providing instructions: 'スケッチを書き込みたい場合には、ボードとポートの両方を選択してください。ボードのみを選択した場合、コンパイルはできますが、スケッチの書き込みはできません。' (If you want to upload a sketch, please select both the board and the port. If you only select the board, compilation is possible, but the sketch cannot be uploaded). The dialog has two columns: 'ボード' (Board) and 'ポート' (Port). Under 'ボード', 'ESP32 Dev Module' is selected with a checkmark. Under 'ポート', 'COM9 Serial Port (USB)' is selected with a checkmark. There are 'キャンセル' (Cancel) and 'OK(O)' buttons at the bottom of the dialog. The background shows a sketch with code for an ESP32, including pin definitions and SPI library includes. The output window at the bottom shows the successful upload of the sketch to the board.

```
1 // 23 DAT
2 // 5 LE
3 // 25 CE
4 // -----
5 // MCU-pin SSD1306
6 // 21 SDA
7 // 22 SCL
8 // -----
9 // Board Power +5V
10 // -----
11 // IO26 = Bit0 pin
12 // IO27 = Bit1 pin
13 // Case 0 = 1990MHz
14 // Case 1 = 1980MHz
15 // Case 2 = 1970MHz
16 // Case 3 = 1960MHz
17 // -----
18 #include "ADF4351.h"
19 #include <SPI.h>
```

出力

```
Writing at 0x000585ab... (100 %)
Wrote 314368 bytes (176912 compressed) at 0x00010000 in 3.1 seconds (effective 815.2 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
```

# Arduino IDE — プログラムをコンパイルしてバグ潰し

Local-OSC | Arduino IDE 2.3.3

ファイル 編集 **スケッチ** ツール ヘルプ(H)

検証・コンパイル Ctrl+R  
書き込み Ctrl+U  
構成と書き込み  
書き込み装置を使って書き込む Ctrl+Shift+U  
コンパイル済みバイナリをエクスポート Alt+Ctrl+S  
デバッグに最適化  
スケッチフォルダを表示 Alt+Ctrl+K  
ライブラリをインクルード  
ファイルを追加.....

```
25 // case 3 - 1500MHz
26 //////////////////////////////////////////////////
27
28
29 #include "ADF4351.h"
30 #include <SPI.h>
31 #include "SSD1306.h"//ディスプレイ用ライブラリを読み込み
32
33 #define clock 18
34 #define data 23
35 #define LE 5
36 #define CE 25
37
38 ADF4351 adf4351(clock,data,LE,CE); // declares object PLL of type ADF4351
39 SSD1306 display(0x3c, 21, 22); //SSD1306インスタンスの作成 (I2Cアドレス,SDA,SCL)
40
41 void setup() {
42   Serial.begin(115200);
43   Serial.println("Program started!");
44   adf4351.begin();
45
46   pinMode(26, INPUT_PULLUP); // 入力ピン(プルアップ抵抗付)
47   pinMode(27, INPUT_PULLUP); // 入力ピン(プルアップ抵抗付)
```

出力

最大1310720バイトのフラッシュメモリのうち、スケッチが314009バイト (23%) 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20800バイト (6%) を使っていて、ローカル変数で306880バイト使うことができます。

行 29、列 1 ESP32 Dev Module COM9の

# Arduino IDE — プログラムを基板に書き込みます

The screenshot displays the Arduino IDE interface. The top menu bar includes 'ファイル', '編集', 'スケッチ', 'ツール', and 'ヘルプ(H)'. The toolbar shows icons for saving, running, and uploading. The board is set to 'ESP32 Dev Module'. The sketch editor shows the following code:

```
1 ///////////////////////////////////////////////////  
2 // ADF4351 RF Generator /  
3 // Local-OSC.ino /  
4 // 2023-2-16 - 2023-4-4 JA4CXX /  
5 ///////////////////////////////////////////////////  
6 // MCU-Pin ADF4351 Board-Pin  
7 // 18 CLK  
8 // 23 DAT  
9 // 5 LE  
10 // 25 CE  
11 //-----  
12 // Board Power +5V  
13 //-----  
14 // MCU-Pin SSD1306 Board-Pin  
15 // 21 SDA  
16 // 22 SCL  
17 //-----  
18 // Board Power +3.3V  
19 //-----  
20 // IO26 = Bit0 pinMode(26, INPUT_PULLUP);  
21 // IO27 = Bit1 pinMode(27, INPUT_PULLUP);  
22 // Case 0 = 1990MHz  
23 // Case 1 = 1980MHz  
24 // Case 2 = 1970MHz  
25 // Case 3 = 1960MHz  
26 //-----  
27  
28  
29 #include "ADF4351.h"  
30 #include <SPI.h>
```

A context menu is open over the sketch, with the following options:

- 検証・コンパイル (Ctrl+R)
- 書き込み (Ctrl+U)**
- 構成と書き込み
- 書き込み装置を使って書き込む (Ctrl+Shift+U)
- コンパイル済みバイナリをエクスポート (Alt+Ctrl+S)
- デバッグに最適化
- スケッチフォルダを表示 (Alt+Ctrl+K)
- ライブラリをインクルード ▶
- ファイルを追加.....

The output window shows the following text:

```
出力  
Writing at 0x000585ab... (100 %)  
Wrote 314368 bytes (176912 compressed) at 0x00010000 in 3.1 seconds (effective 815.2 kbit/s)...  
Hash of data verified.  
  
Leaving...  
Hard resetting via RTS pin...
```

A notification box at the bottom right indicates '書き込み完了' (Upload completed).