



CQ インバータ・キット 2

取り扱い説明書（ソフトウェア編）

Ver.01

2026 年 6 月 24 日

CQ出版社

目 次

1. 開発環境の構築

1.1 STM32CubeIDE の入手とインストール	1 - 6
1.2 STM32CubeMX の入手とインストール	6 - 10
1.3 STM32CubeIDE で筆者プロジェクトを書き込む	11 - 16
1.4 プログラムの編集	16 - 17
1.5 STM32 開発の流れ	18
1.6 main.c を書き換えた後の手順	18, 19
1.7 Src フォルダ下のプログラムについて	19, 20
1.8 CQEV ミニカートレースで使用する場合の設定項目	21
1.8.1 アクセル入力範囲の設定	21, 22
1.8.2 起動時アクセル NG 検出	22
1.8.3 最大デューティ設定	22, 23
1.8.4 起動最低デューティ	23
1.8.5 加速・減速スロープの設定	23, 24
1.8.6 電流検出値の調整	24, 25
1.8.7 電流制限設定	25
1.8.8 低速ポーリング転流とホール割り込み転流	25, 26
1.8.9 TIM2 と TIM6 の役割	26, 27
1.8.10 転流時のデッドタイム	27
1.8.11 回転方向の設定	28
1.8.12 初回走行前の確認項目	28

1、開発環境の構築

1.1 STM32 用統合開発環境 STM32CubeIDE の入手とインストール

下記 Web サイトにアクセスします。

<https://www.st.com/ja/development-tools/stm32cubeide.html>

次に使用している PC の OS やソフトウェアのバージョン確認し「ダウンロード」をクリックします (図 1)。その後、ライセンス契約画面がでるので、「承諾します」をクリックします (図 2)。

ソフトウェア入手

ソフトウェア名	ECCN	OS	バージョン	アクション
STM32CubeIDE	NEC (EU) EAR99 (US)	Windows	2.1.1 Latest	
STM32CubeIDE for Visual Studio Code	NEC (EU) EAR99 (US)	-	-	 Visual Studio Codeマーケットプレイスに移動

GET-SOFTWARE-RECOMMENDS

図 1 STM32CubeIDE のダウンロード

APPLICABLE, NO LICENSE OR OTHER RIGHTS, WHETHER EXPRESS OR IMPLIED, ARE GRANTED UNDER ANY PATENT OR OTHER INTELLECTUAL PROPERTY RIGHTS OF STMICROELECTRONICS OR ANY THIRD PARTY.

 Additional License Terms for STM32CubeIDE 2.1.0

承諾しません 

図 2 ライセンス契約画面

続いて、アカウント関係の画面に移ります (図 3)。MyST アカウントがある方はログインし、ない方は「ゲストとしてダウンロード」をクリックします。ここでアカウントを作成することもできます。

Thanks for having accepted the license.

MySTにログインするか、アカウントを作成し、
ダウンロードを開始してください。



図3 MyST アカウント有無に応じてクリック

ここでは、ゲストとしてダウンロードします。必要な情報を入力し、最後に「ダウンロード用のリンクを入手」をクリックします（図 4）。

ゲストとしてダウンロード

by filling this form you will receive an email with the
link to download the software.

名

姓

業務用メールアドレス *

姓名, メールアドレスを入力

STによるお客様のプロファイル情報の処理方法、またお客様がご自身の個人情報の保護に関する権利を行使する方法については、当社の [プライバシー・ステートメント](#) をご確認ください。

☐ Please keep me informed about future
updates for this software or new software in
the same category

ダウンロード用のリンクを入手

入力したらクリック

図 4 ダウンロードに必要な情報を入力

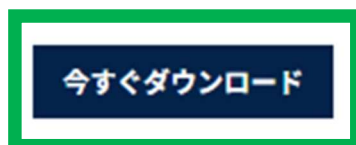
入力したメールアドレス宛にリンクが送信されてくるので、メールにある「今すぐダウンロード」をクリックします（図 5）。

これで、インストーラがダウンロードできました。



日頃は弊社製品をご愛顧いただきまして誠にありがとうございます。

メールアドレスを認証して **STM32CubeIDE-Win** をダウンロードするには、以下のボタンをクリックしてください。



よろしくお願いいたします。
STマイクロエレクトロニクス

図5 入力したメールアドレス宛にリンクが送られてくる

入手したインストーラを解凍して実行します。最初の画面では「Next」をクリックします(図6)。

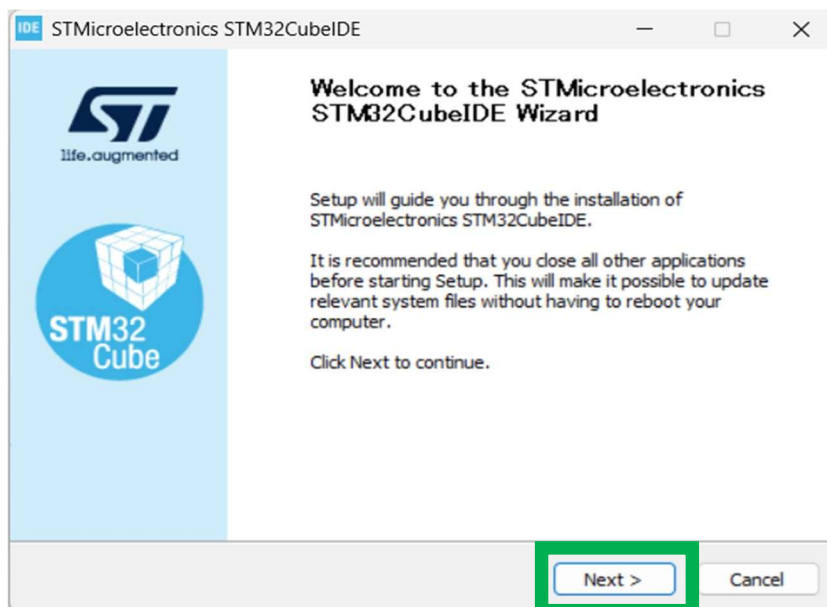


図6 インストーラ実行後の最初の画面

以降, License Agreement の画面では「I Agree」を選択し, インストール先も指定します

(図7). 最後に「Install」をクリックしインストールを実行し (図8), 最後に「Finish」をクリックすればインストール完了です (図9).

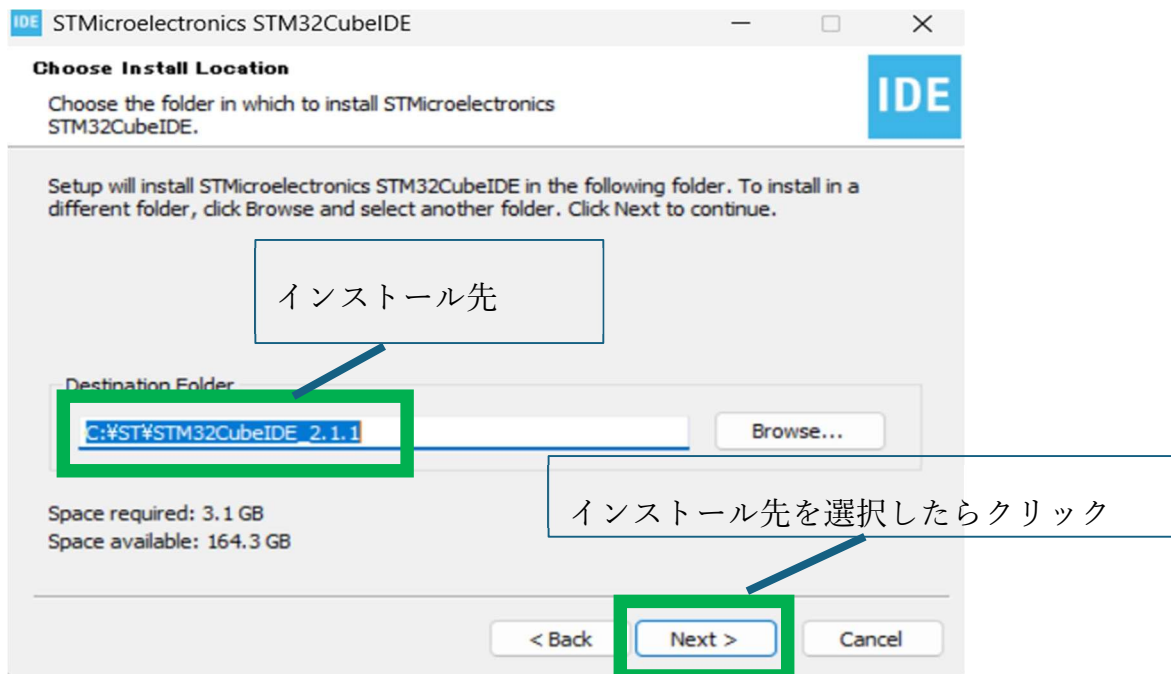


図7 インストール先の指定 (デフォルトで問題ない)

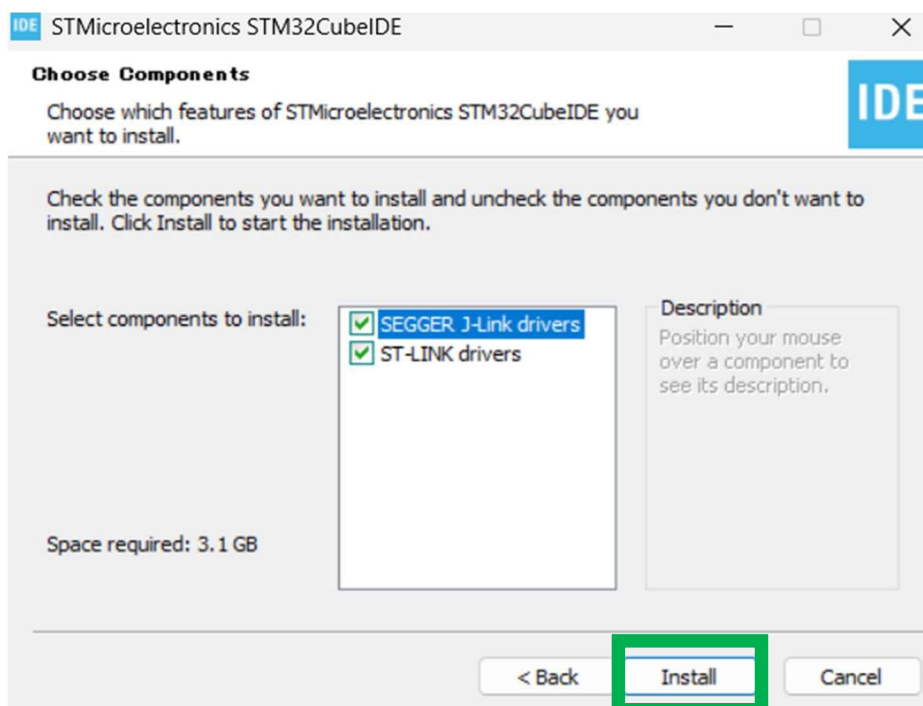


図8 「Install」をクリックしてインストールを実行



図9 「Finish」をクリックすればインストール完了

1.2 STM32CubeMXの入手とインストール

STM32CubeMXは、STM32マイコンの各種設定をGUI画面上で行うためのツールです。GPIOの入出力設定、クロック設定、タイマ、UART、A-Dコンバータ、PWMなどの周辺機能を画面操作で設定でき、設定内容に応じた初期化コードを自動生成します。従来はレジスタ設定を手作業で記述する必要がありましたが、CubeMXを使うことで設定ミスを減らし、開発効率を大幅に向上できます。また、STM32CubeIDEと連携することで、生成したプロジェクトをそのままビルド・デバッグへ進められます。

本書では主に筆者提供プロジェクトを利用するため、CubeMXを直接使用する機会は少ないですが、将来的に周辺機能設定を変更する際に利用します。

まずはSTマイクロエレクトロニクス公式ウェブ・サイトへアクセスします。

<https://www.st.com/ja/development-tools/stm32cubemx.html>

STM32CubeIDEのときと同じように使用しているPCのOSやソフトウェアのバージョンを確認して「ダウンロード」をクリックします（図10）。

ソフトウェア入手

ソフトウェア名	対応ハードウェア	ECCN	OS	バージョン	アクション
STM32CubeMX	All MCU/MPU except STM32C5 Series	NEC (EU) 5D992.c (US)	Windows ▾	6.17.0 Latest ▾	 ダウンロード
STM32CubeMX2	STM32C5 Series ⓘ	NEC (EU) 5D992.c (US)	Windows ▾	1.0.1 Latest ▾	 ダウンロード

GET-SOFTWARE-RECOMMENDS

図 10 STM32CubeMX のダウンロード

以降は STM32CubeIDE の入手、インストールのときと同様にライセンス契約画面やアカウント関連の画面が出てきますので、同じように選択すれば STM32CubeIDE のインストーラがダウンロードできます。

ダウンロードしたら zip ファイルを展開し exe ファイルを実行します。すると、図 11 の画面がでてくるので、推奨されている「Install for me only」をクリックします。次に画面の Starting STM32CubeMX 6.17.0 installation では「Next」をクリックし、LICENSE AGREEMENT の画面では図 12 のようにチェックを入れて「Next」をクリックします。その後、図 13 も同じようにチェックを入れて「Next」をクリックします。

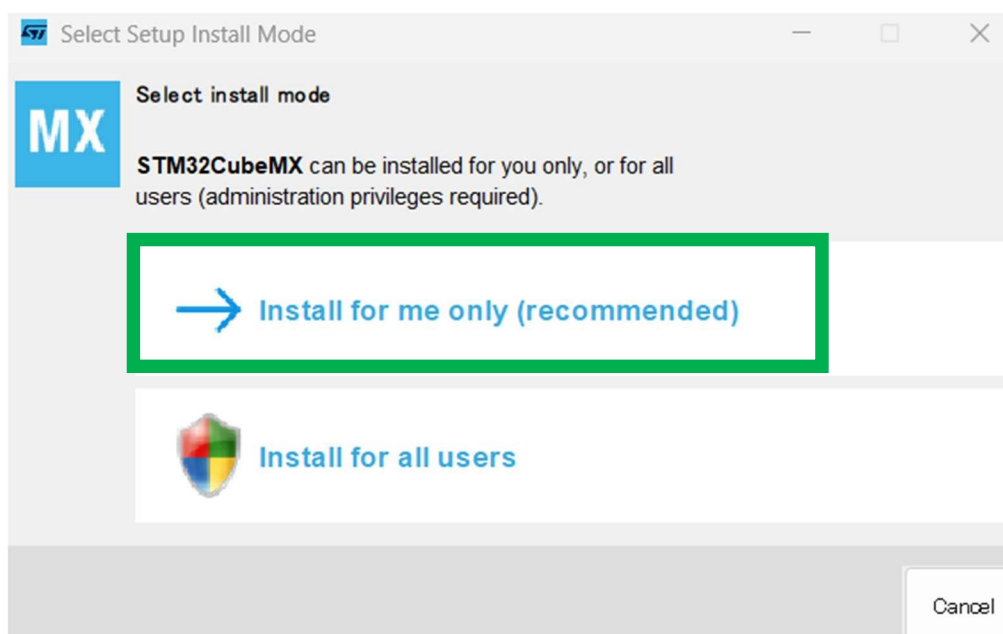


図 11 ユーザ設定画面

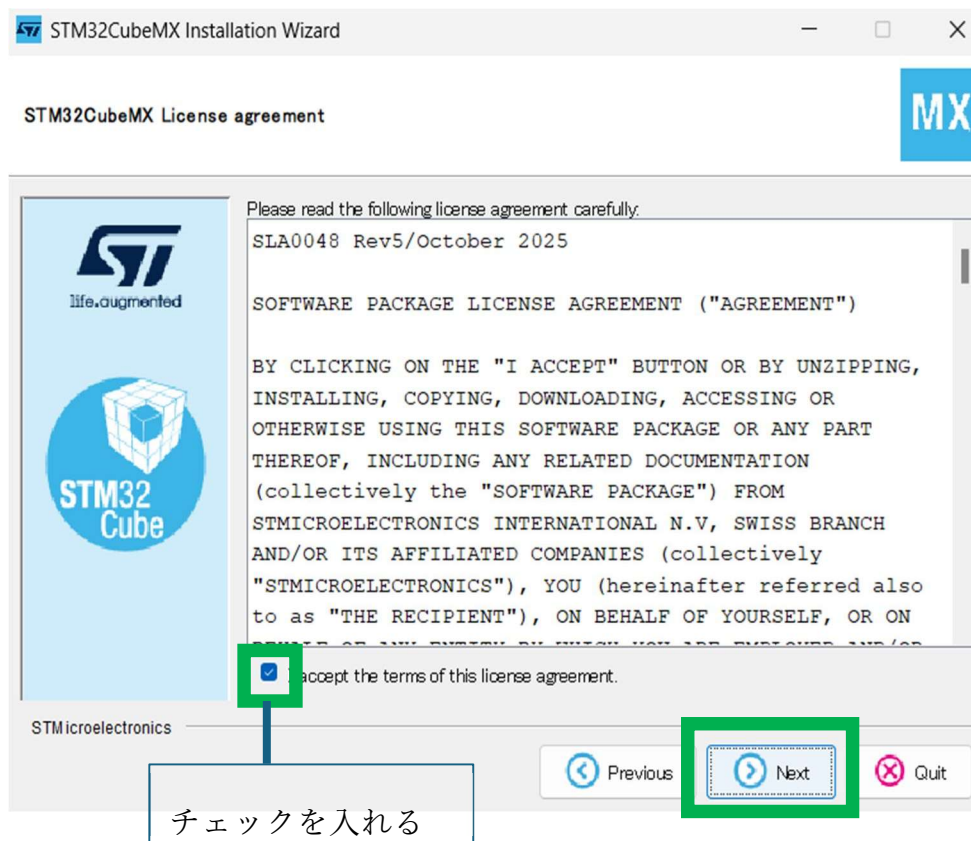


図 12 ライセンス画面

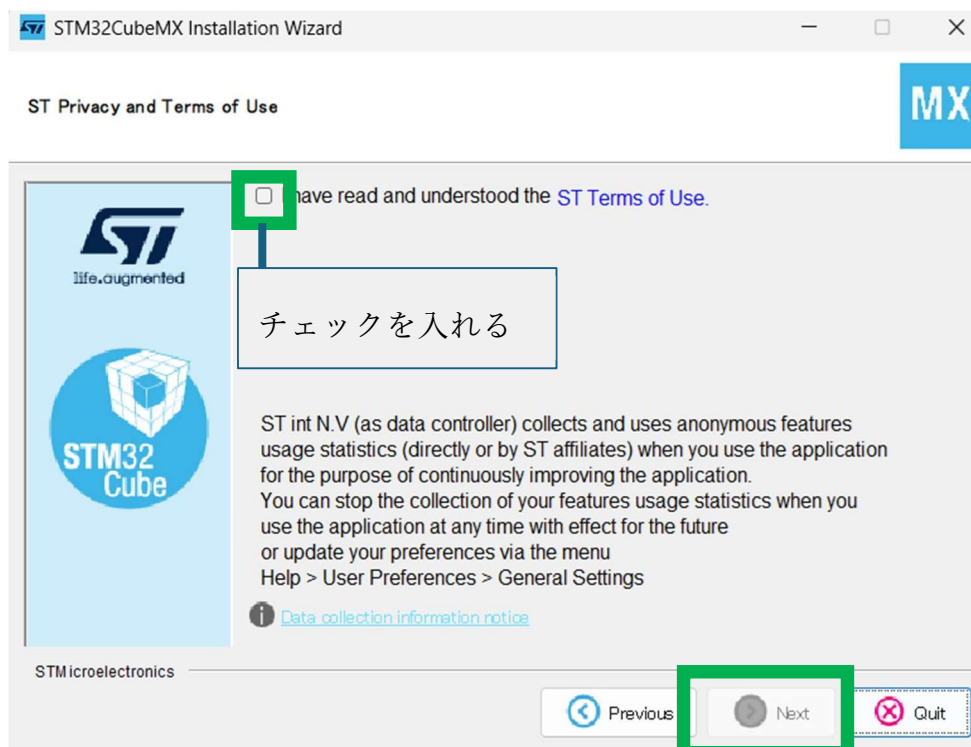


図 13 ST Privacy and Terms of Use 画面

続いて、図 14 のようにパスの設定画面がでてくるので、「Next」をクリックします。
ここで、フォルダがない場合は図 15 のようなメッセージがでるので「OK」をクリックします。

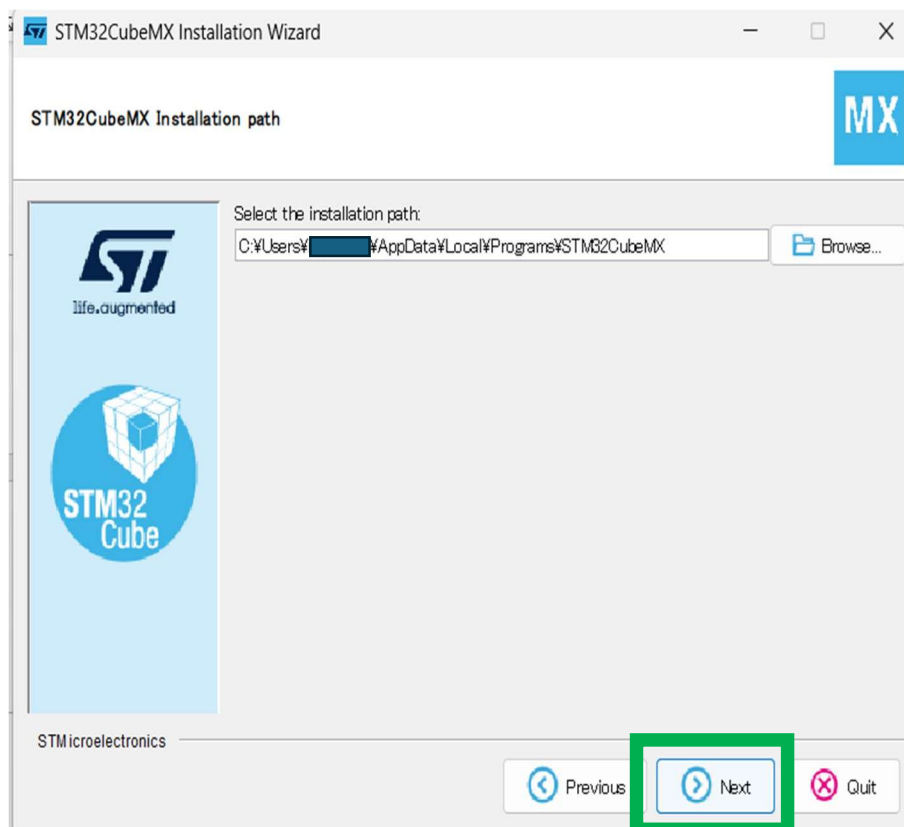


図 14 パスの設定画面

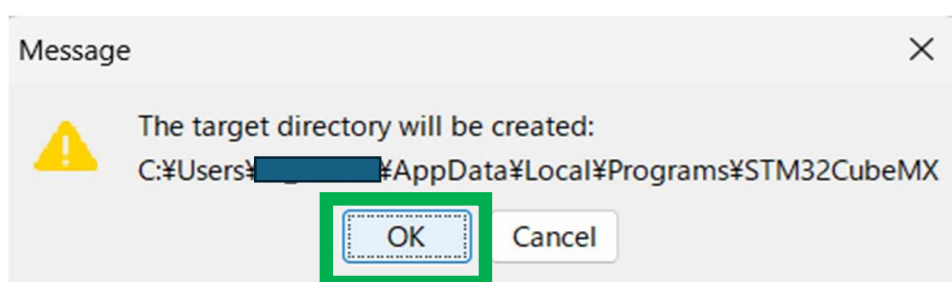


図 15 ディレクトリの新規作成

その次の画面は既定設定のままで OK です (図 16).

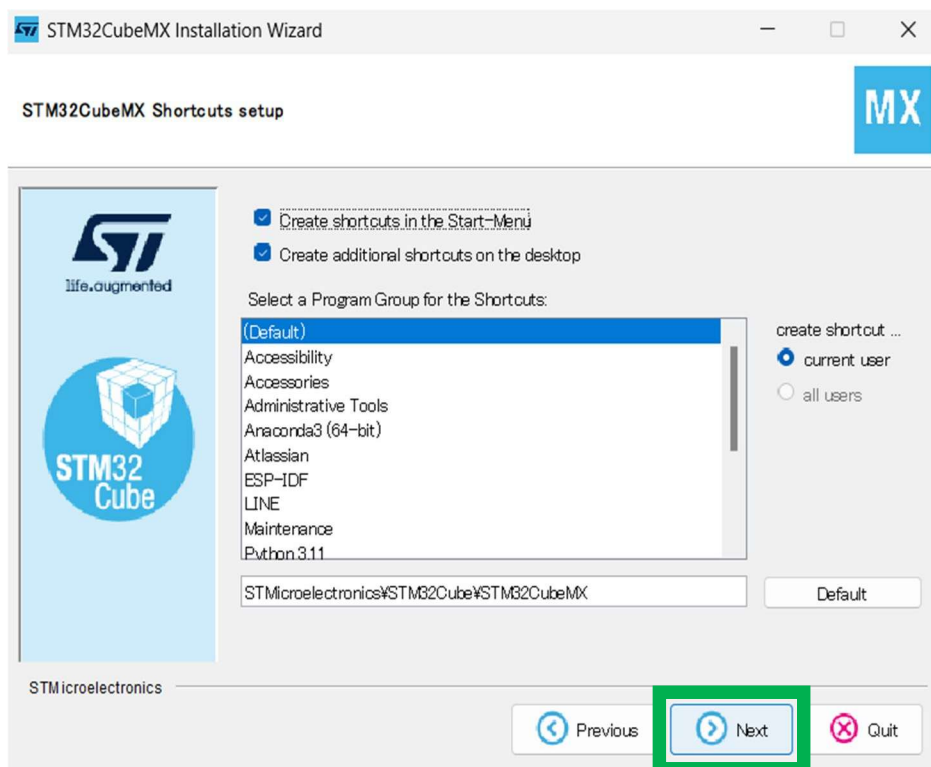


図 16 セットアップ画面

以降、「Next」をクリックしていき、最後に「Done」をクリックすればインストール完了です（図 17）。

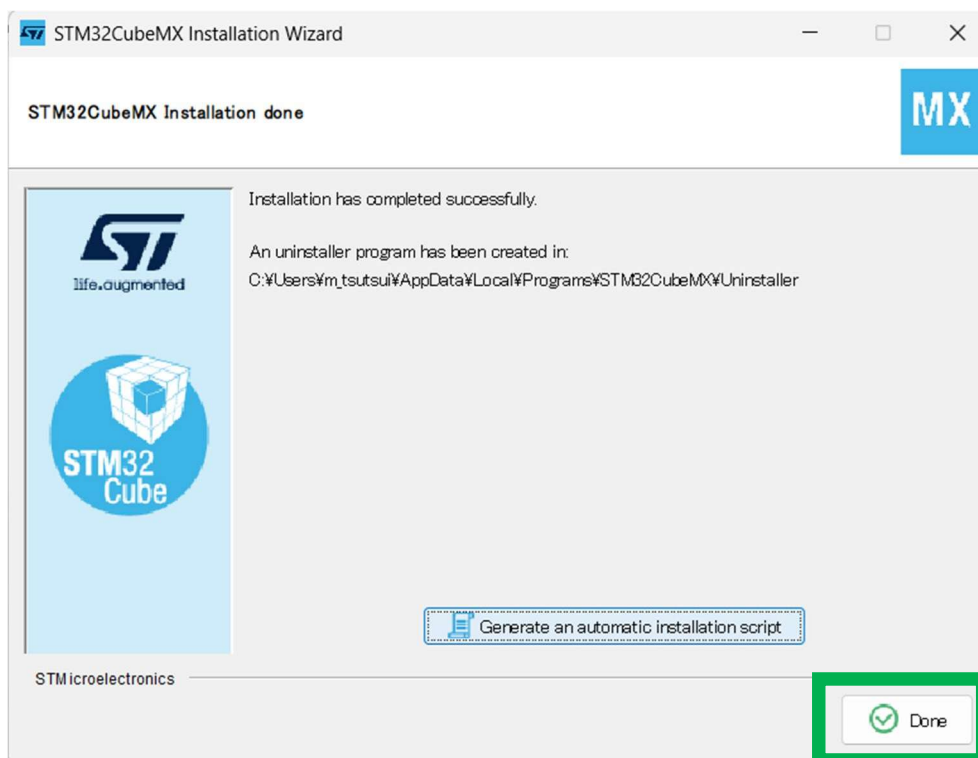


図 17 「Done」をクリックすればインストール完了

1.3 STM32CubeIDE で筆者プロジェクトを書き込む

ここでは筆者が提供するプロジェクト（CQ_Motor_CubeSimple0X.zip）をマイコンボードに書き込む方法について説明します（X はバージョンで異なる）。

入手先は、<https://www.cqpub.co.jp/interface/download/2026/8/EVcart20260601.zip>

まずは、入手した CQ_Motor_CubeSimple0X.zip を展開しておきます。ここでは、展開先はデスクトップとして説明します。続いて、STM32CubeIDE を起動します。起動すると図 18 の画面が表示されます。

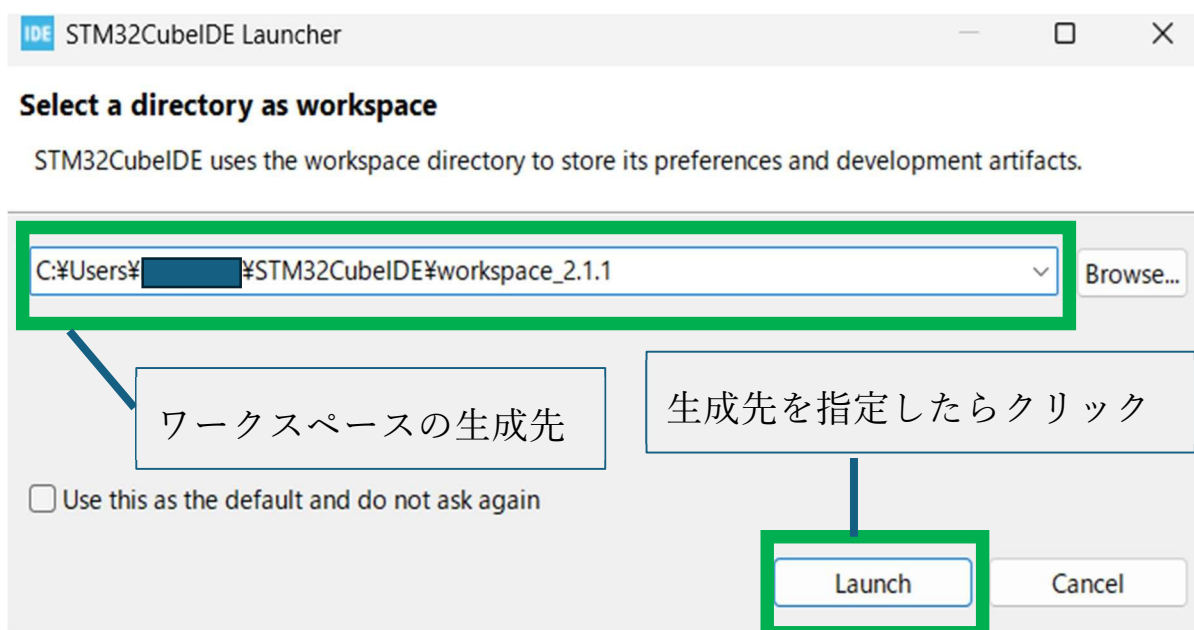


図 18 ワークスペース・ディレクトリの生成先を指定

ワークスペースの生成先を指定したら「launch」をクリックします。ここで、追加の質問はせず、この設定をデフォルトとして適用とする場合は「Use this as the default and do not ask again」にチェックを入れます。無事に起動すると図 19 の画面が表示されます。

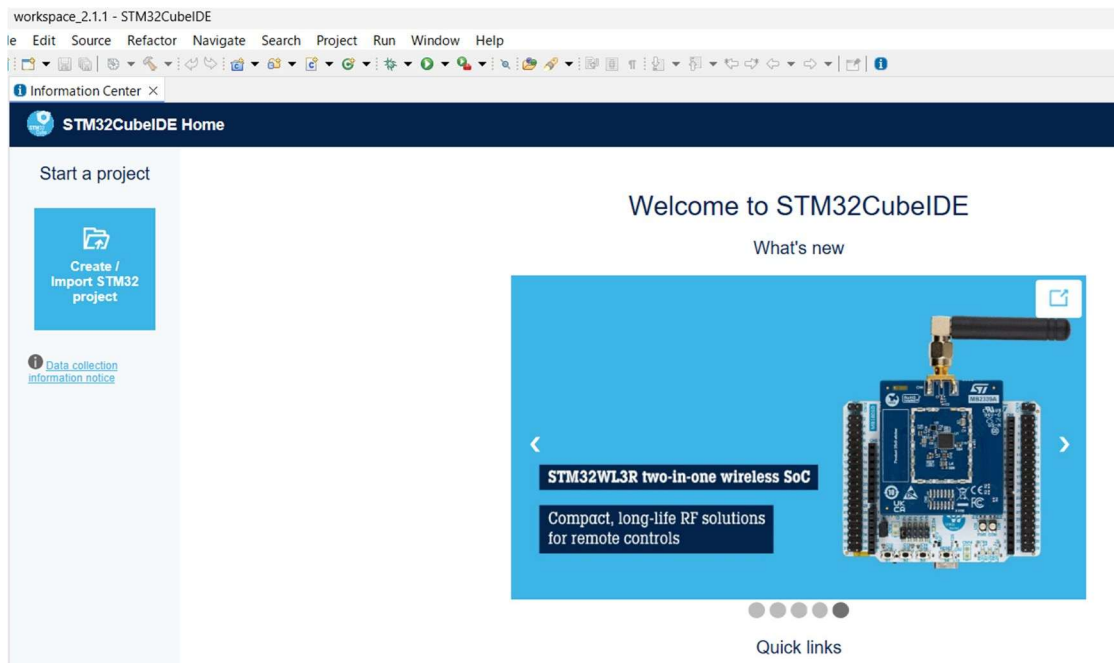


図 19 STM32CubeIDE 起動時の画面

次に、「File」→「Open Projects from File System」と進み（図 20）、Import Source のところで「Directory…」をクリックします（図 21）。

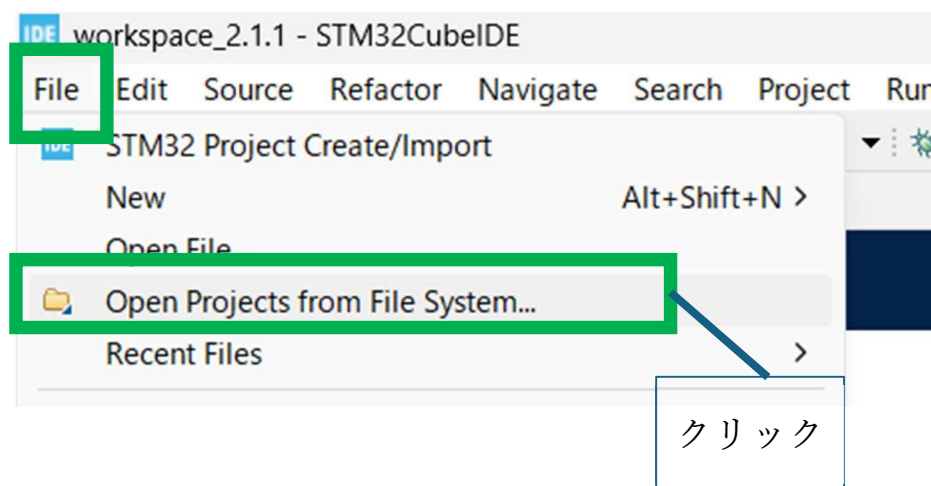


図 20 左上にある「File」から「Open Projects from File System」をクリック

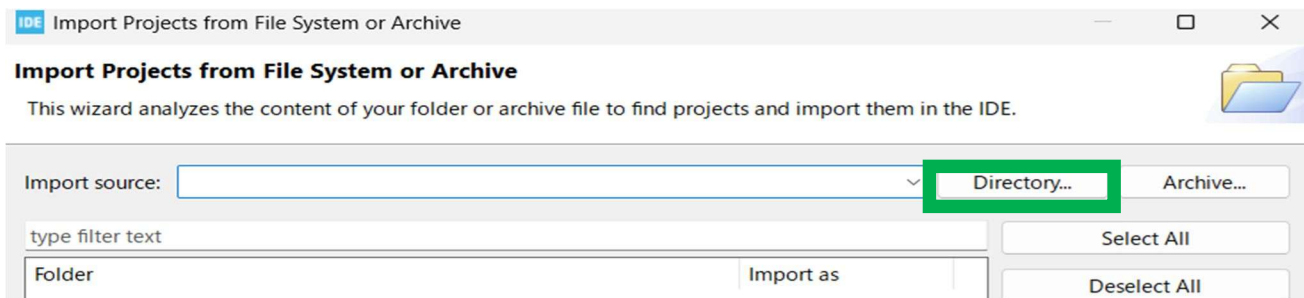


図 21 Import source の「Directory…」をクリック

クリックするとフォルダ選択の画面になるので、CQ_Motor_CubeSimple0X.zip の展開したフォルダを指定します（図 22）。

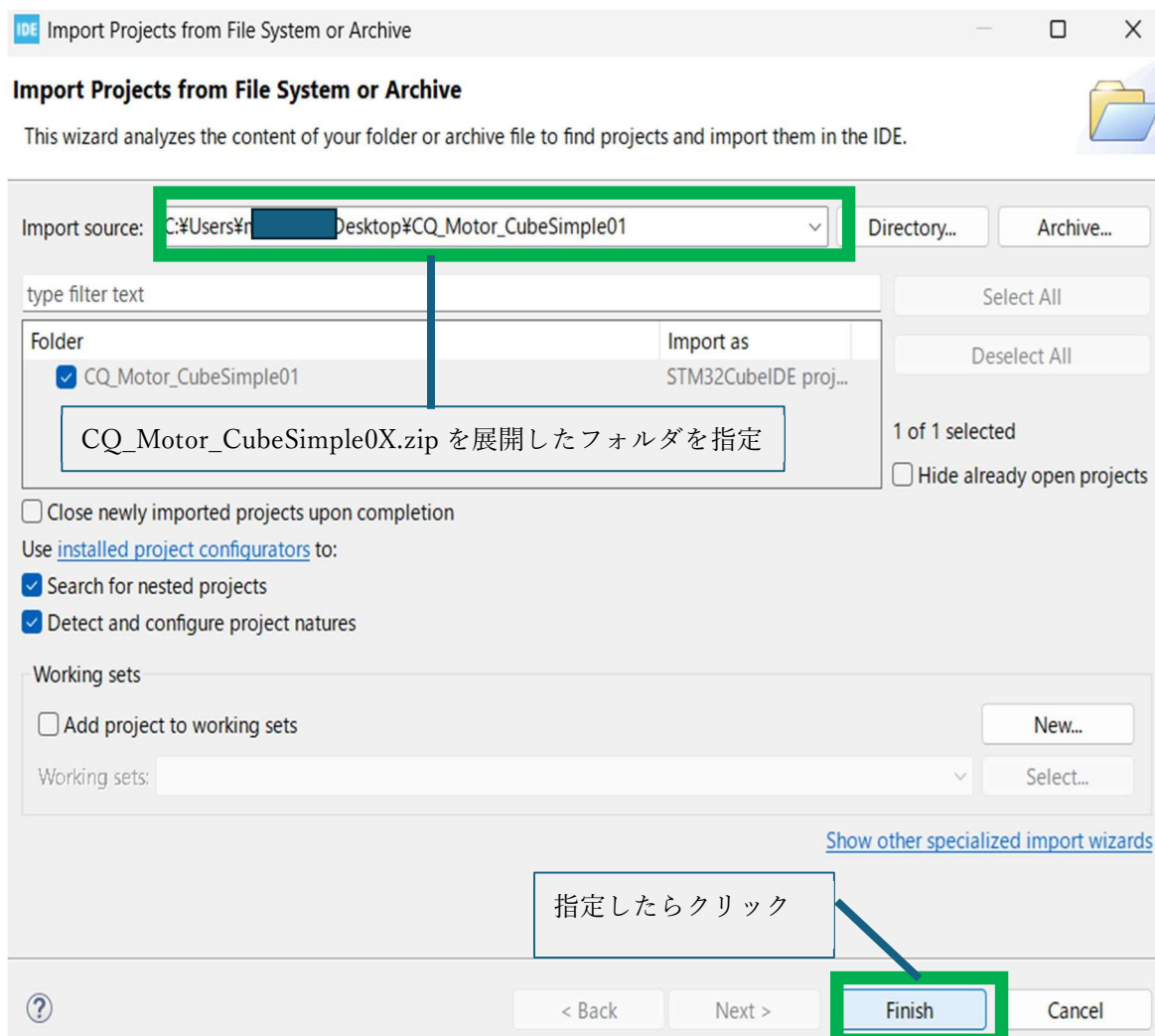


図 22 CQ_Motor_CubeSimple0X.zip の展開先フォルダを指定し「Finish」をクリック

すると、図 23 のようにプロジェクト・フォルダに必要なファイルなどが表示されます。

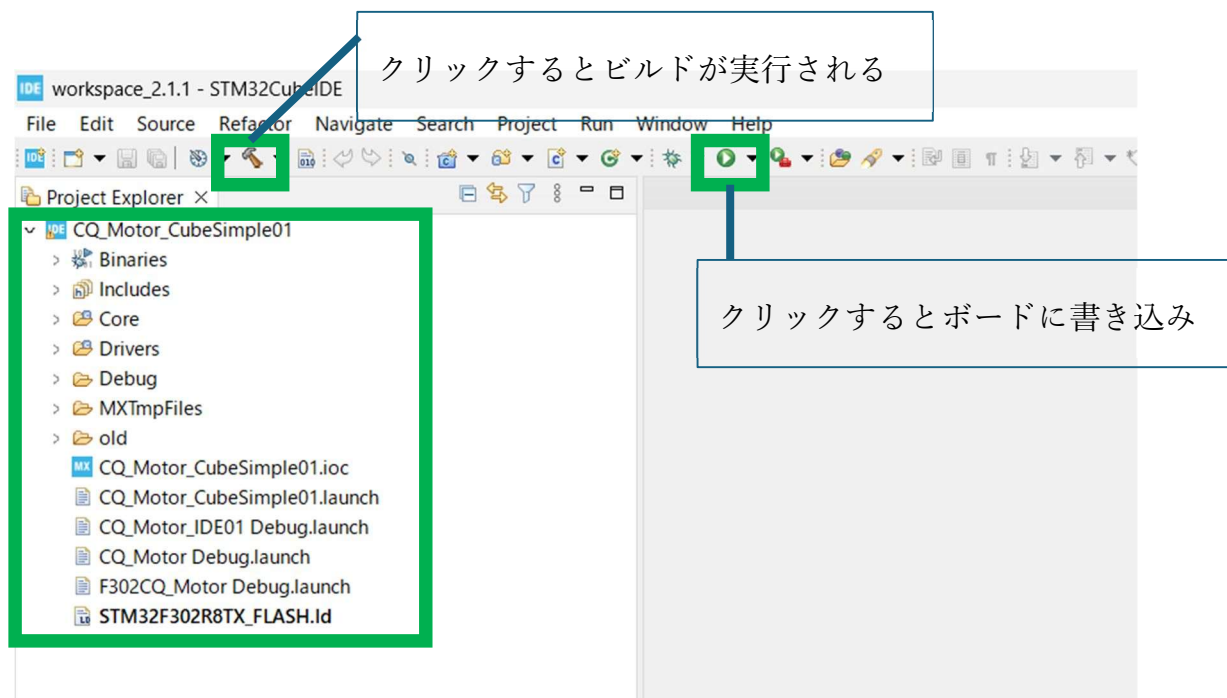


図 23 プロジェクト・フォルダの構成が表示される

プロジェクト・フォルダの中身が表示されたら、上部のハンマー（）ボタンをクリックしビルドを実行します。正常終了すると、.bin や.elf ファイルが生成されます。その後、Nucleo ボードと PC を USB 接続し、上部の▶（Run）をクリックします。このとき、図 24 のような画面がでたら「OK」をクリックし、次の画面で「Open in update mode」をクリック後、「Upgrade」をクリックします(図 25)。Upgrade が終わると左下に「Upgrade successful」と表示されるので、表示されたら右上の×印をクリックし画面を閉じます。

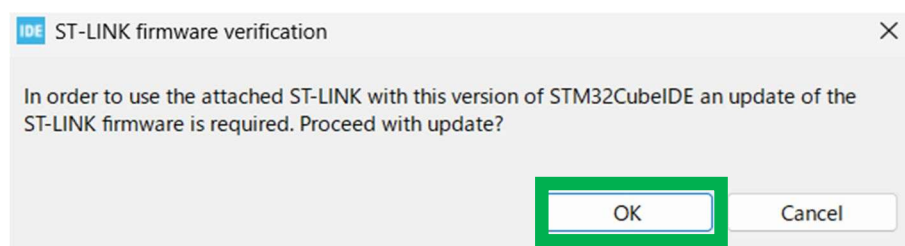


図 24 ST-LINK のファームウェアを更新

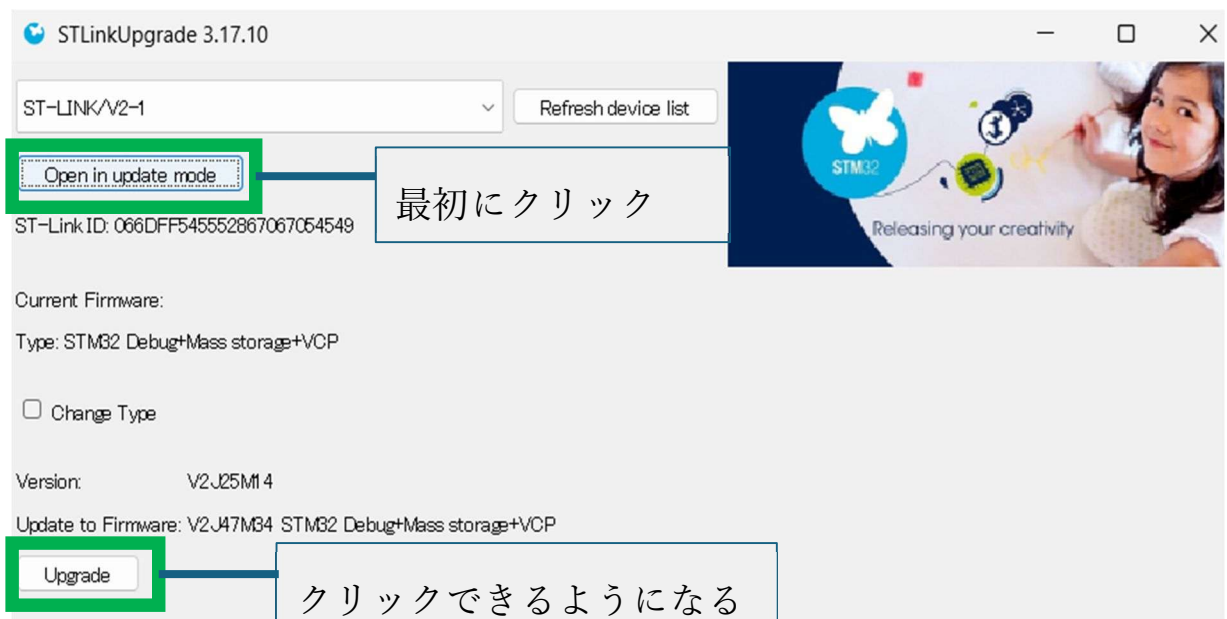
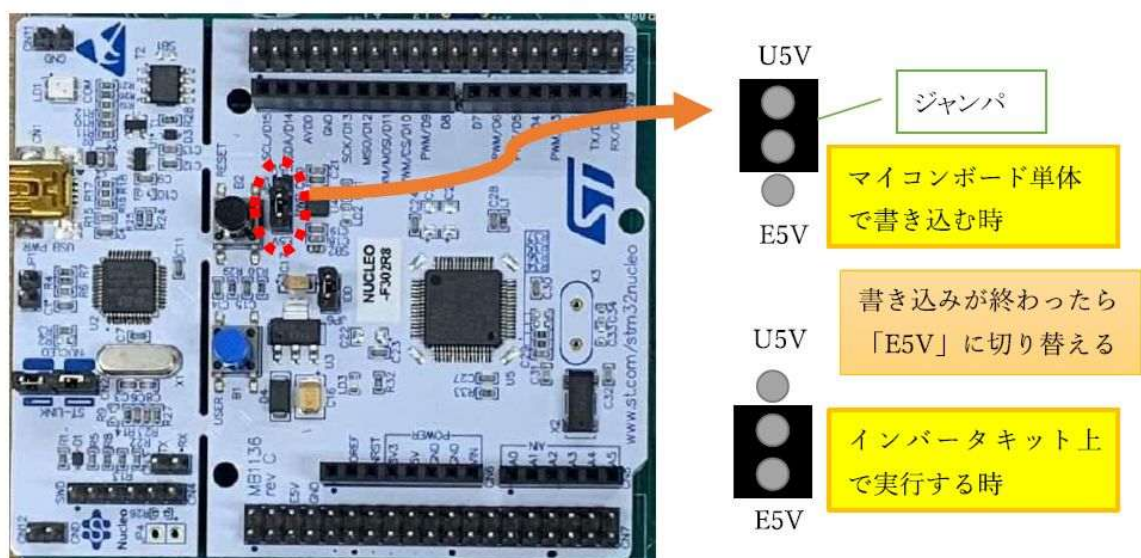


図 25 「Open in update mode」をクリックすると「Upgrade」ボタンが有効になる

無事に書き込みが終わると、図 26 のように Console に Download verified successfully と表示されれば成功です。これは STM32 への転送成功、書き込み検証成功を意味します。



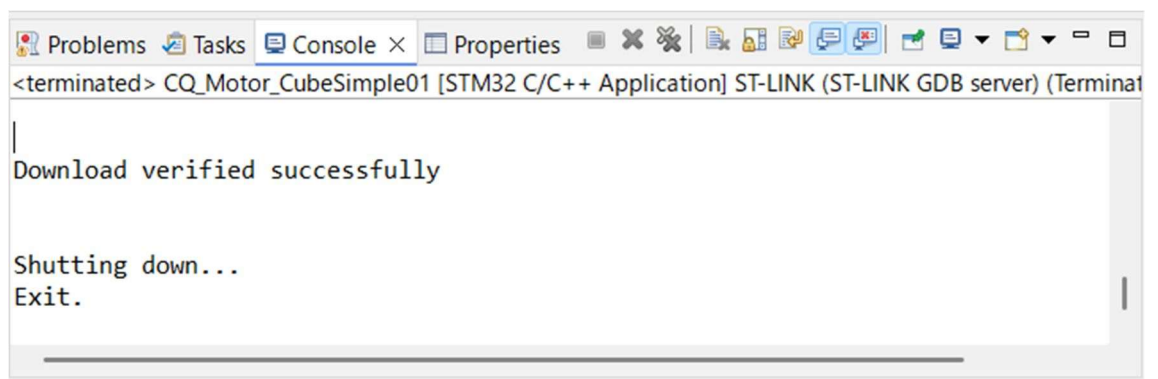


図 26 ボードへの書き込み

書き込みが終わったら、ジャンパ・ピン JP5 を E5V に戻すことを忘れないでください。

ここで、注意点として、Drivers 以下は巨大ライブラリです。誤って Drivers/以下を開くと、「Editor Scalability」と警告が出ることがあります。これは巨大ファイル警告であり、異常ではありません。基本は、「No」を選択します。

1.4 プログラムの編集

主に編集するのは：

Core

└ Src

└ main.c

です（図 27）。Drivers 以下は STM 公式ライブラリなので、初心者は基本触らないでください。

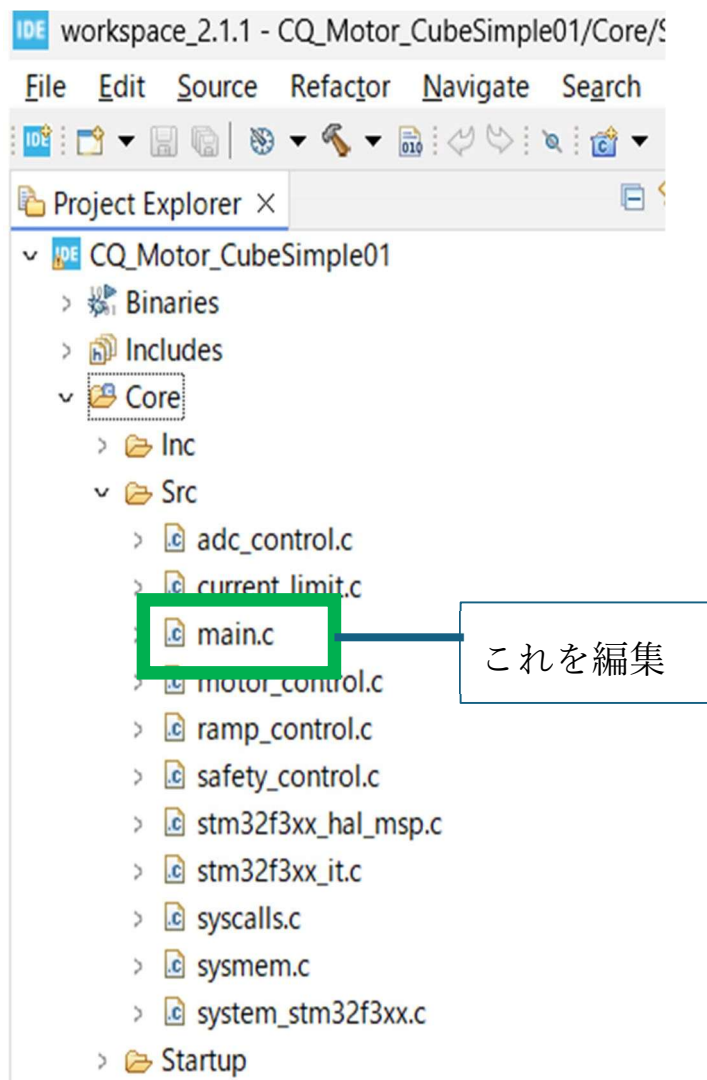


図 27 プログラム編集するのは main.c

1.5 STM32 開発の流れ

STM32 開発は概ね,



という流れで進みます。ここで、それぞれの用語について説明しておきます。

項目	内容
ビルド	ソースコードを機械語へ変換する作業
make	ビルド手順を管理するツール
elf	デバッグ用完成ファイル
bin	STM32 へ書き込む実行ファイル
ST-LINK	STM32 書き込み器

1.6 main.c を書き換えた後の手順

STM32 では、main.c を修正→ビルド→STM32 へ再書き込みという流れになります。

1. main.c を保存

編集後 [Ctrl] + [S] を押して保存する。

2. ビルドする

画面上部の  (ハンマー) ボタンをクリックする。または「Project」→「Build Project」を選択する。

3. ビルド成功確認

Console に「Build Finished」などが表示されれば成功。この時点で bin, elf が更新される。

4. STM32 へ書き込む

画面上部の  (Run) をクリックする。

5. 書き込み成功確認

Console に「Download verified successfully」と表示されれば成功。STM32 は新しいプログラムで動作を開始する。

1.7 Src フォルダ下のプログラムについて

このプログラムは、BLDC（ブラシレス DC）モータを STM32 で制御するためのソフトウェアです。特徴は、機能ごとにファイルを分割している点にあります。大学のプログラミング演習では main.c にすべてを書く例も多いですが、本プログラムでは役割分担を明確にしています。これは実際の組み込み開発でよく使われる構成です。

プログラム全体の流れは、main.c が中心となります。main.c では、STM32 の初期化を行ったあと、

- モータ制御初期化
- A-D コンバータ初期化
- 起動時アクセル安全チェック
- タイマ開始（TIM2：制御周期生成、TIM6：ホール監視）
- 制御ループ実行

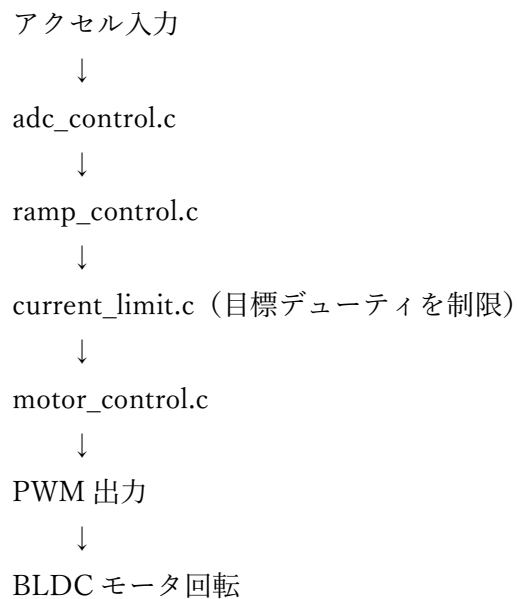
などを実行します。TIM2 が 5ms 周期で制御タイミングを生成し、main 関数の while(1) ループ内で制御処理を実行しています。

まず、adc_control.c ではアクセル値とモータ電流を ADC で取得しています。例えばアクセル電圧を 0～1.0 の範囲に正規化し、そこからデューティ比へ変換しています。デューティ比とは PWM の ON 時間割合であり、モータ速度を決める重要な値です。しかし、アクセル値をそのままモータへ与えると急加速して危険です。そこで ramp_control.c が使われます。このファイルではデューティを徐々に変化させています。例えば、アクセルを急に全開にしても、デューティを少しずつ増加させることで滑らかな加速を実現しています。これは電気自動車や電動工具でも使われる考え方です。

さらに current_limit.c では過電流保護を行っています。モータ電流が設定値を超えた場合、デューティを減少させます。さらに重大な過電流が発生すると、異常状態を保持し、アクセルを戻すまでモータ出力を再開できないようにしています。これはモータや MOSFET の破損防止に重要です。

実際に PWM 信号を出力しているのが motor_control.c です。このプログラムでは TIM1 を用いて三相 PWM を生成しています。また、ホール・センサの状態を読み取り、6 ステップ整流を行っています。BLDC モータでは回転位置に応じて励磁相を切り替える必要がありますが、その切り替えテーブルが drive_table_forward です。

つまり全体としては,



という流れで動作しています. `current_limit.c` が最終的な許可デューティを決定し, `motor_control.c` がその値を PWM 出力へ反映します.

このように機能分割を行うことで, 「速度制御だけ変更したい」, 「保護機能を追加したい」といった場合でも, 他の部分へ影響を与えにくくなります. これは大規模な組み込みソフトウェアで特に重要な考え方です.

1.8 CQEV ミニカートレースで使用する場合の設定項目

本プログラムは、CQ インバータキットを用いて BLDC モータを駆動するためのサンプルプログラムです。CQEV ミニカートレースなどの実車両で使用する場合は、アクセル方式、モータ巻線、車両重量、ギヤ比、タイヤ径、バッテリー電圧などに応じて、必ず各種パラメータを調整してください。

設定が不適切な場合、急発進、過電流、FET 破損、モータ破損、車両の暴走などにつながる可能性があります。初回走行時は、車輪を浮かせた状態で低デューティから確認し、電流、発熱、回転方向、アクセル応答を確認しながら段階的に調整してください。

1.8.1 アクセル入力範囲の設定

アクセル入力、使用するアクセルの種類や電源電圧によって出力範囲が異なります。ボリューム式アクセルとホール式アクセルでは、全閉時および全開時の電圧範囲が異なるため、必ず実車両に搭載するアクセルに合わせて調整してください。

CQEV ミニカート標準アクセルを 3.3V で使用する場合、出力電圧範囲は概ね以下のようになります。

全閉時：約 1.2V

全開時：約 2.4V

本プログラムでは、Core/Src/adc_control.c 内でアクセル有効範囲を設定します。

```
#define ACCEL_CLOSED_VOLTAGE    1.20f
#define ACCEL_OPEN_VOLTAGE      2.40f
```

この設定により、アクセル電圧が約 1.2V のときにアクセル開度 0%、約 2.4V のときにアクセル開度 100%として扱います。

STM32 の ADC 基準電圧を 3.3V、ADC 分解能を 12bit とすると、ADC 値の目安は以下です。

1.2V → 約 1489

2.4V → 約 2978

アクセル実測値が異なる場合は、ACCEL_CLOSED_VOLTAGE と ACCEL_OPEN_VOLTAGE を実車両に合わせて変更してください。

例として、全閉 1.10V、全開 2.55V の場合は以下のように設定します。

```
#define ACCEL_CLOSED_VOLTAGE    1.10f
#define ACCEL_OPEN_VOLTAGE      2.55f
```

アクセルを戻しているにもかかわらずモータが回り出す場合は、アクセル全閉電圧の設定や不感帯を確認してください。

```
#define ACCEL_DEAD_ZONE         0.05f
```

1.8.2 起動時アクセル NG 検出

本プログラムには、電源投入時にアクセルが開いていた場合にモータ出力を禁止する安全機能があります。これは、アクセル故障や操作ミスにより、電源投入と同時に車両が急発進することを防ぐための機能です。

設定箇所は Core/Src/safety_control.c です。

```
#define STARTUP_ACCEL_RESET_LEVEL    0.03f
#define STARTUP_ACCEL_RESET_COUNT_REQUIRED  50U
```

電源投入直後は、アクセル状態にかかわらず出力禁止状態から開始します。その後、アクセル開度が STARTUP_ACCEL_RESET_LEVEL 未満である状態を STARTUP_ACCEL_RESET_COUNT_REQUIRED 回連続で確認すると、出力許可状態になります。

本プログラムでは制御周期が 5ms のため、50U の場合は以下の時間になります。

50 回 × 5ms = 約 250ms

つまり、電源投入後にアクセル全閉状態を約 250ms 連続で確認してから、モータ出力を許可します。

この値を小さくしすぎると、電源投入直後の ADC 値の揺れにより、アクセルが開いているにもかかわらず出力許可になる場合があります。実機確認の結果、50U を標準値とします。

1.8.3 最大デューティ設定

車両の最高速や最大出力は、PWM の最大デューティによって制限できます。設定箇所は Core/Src/adc_control.c です。

```
#define DUTY_MIN          0.00f
#define DUTY_MAX          1.00f
```

DUTY_MAX を大きくすると、アクセル全開時の出力が大きくなります。逆に小さくすると、最高速と最大電流を抑えることができます。

初回走行時は DUTY_MAX を低めに設定し、モータ電流、FET 温度、モータ温度、車速を確認しながら段階的に調整してください。

1.8.4 起動最低デューティ

BLDC モータの 120° 矩形波駆動では、停止状態から最初のホールエッジが発生するまで、現在のロータ位置に応じた初期通電を行う必要があります。デューティが小さすぎると、停止位置やコギングトルクの影響により、モータが起動できない場合があります。

本プログラムでは、起動時の最低デューティを Core/Src/motor_control.c で設定しています。

```
#define MOTOR_START_DUTY_MIN 0.08f
```

この値は、停止状態から起動するとき最低限与える PWM デューティです。起動性を改善する一方で、値を大きくしすぎると急発進の原因になります。

実車両で調整する場合は、車輪を浮かせた状態で確認し、起動できる範囲でできるだけ小さい値に設定してください。

1.8.5 加速・減速スロープの設定

アクセル操作に対して PWM デューティを急激に変化させると、急発進、タイヤの空転、チェーンや駆動系への衝撃、過電流の原因になります。本プログラムでは、デューティを徐々に変化させるスロープ処理を入れています。

設定箇所は Core/Src/ramp_control.c です。

```
#define DUTY_RAMP_UP_STEP    0.002f
#define DUTY_RAMP_DOWN_STEP 0.004f
```

DUTY_RAMP_UP_STEP は加速時のデューティ増加量です。値を大きくすると加速が鋭くなり、小さくすると穏やかになります。

DUTY_RAMP_DOWN_STEP はアクセルを戻したときのデューティ減少量です。値を大きくするとアクセル OFF 時の応答が速くなります。

車両が急に飛び出す、タイヤが空転する、電流制限が頻繁に入る場合は、DUTY_RAMP_UP_STEP を小さくしてください。

1.8.6 電流検出値の調整

本インバータでは、U/V/W 各相のローサイド FET のソース側にシャント抵抗が配置されています。そのため、電流検出には以下の 3 つの ADC 入力を使用します。

CURRENT_U_ADC

CURRENT_V_ADC

CURRENT_W_ADC

ローサイド FET が ON している相だけが、その瞬間のシャント電流を検出できます。そのため、本プログラムでは、現在 ON しているローサイド相に対応する電流値を採用しています。

電流値のゼロ点は、基板個体差、オペアンプのオフセット、部品誤差などによりずれる場合があります。モータを停止し、電流が流れていない状態で各相の ADC 値を確認し、0A 時の電圧に合わせて以下の値を調整してください。

設定箇所は Core/Src/adc_control.c です。

```
#define CURRENT_U_ZERO_VOLTAGE 1.563f
```

```
#define CURRENT_V_ZERO_VOLTAGE 1.563f
```

```
#define CURRENT_W_ZERO_VOLTAGE 1.563f
```

例えば、0A 時の ADC 値が 1940 で、ADC 基準電圧が 3.3V の場合、ゼロ点電圧は以下のようになります。

$$1940 / 4095 \times 3.3V \doteq 1.563V$$

この場合、対応する相のゼロ点設定を以下のようにします。

```
#define CURRENT_U_ZERO_VOLTAGE 1.563f
```

U/V/W 各相で 0A 時の ADC 値が異なる場合は、各相ごとに個別に補正してください。

また、シャント抵抗値や増幅回路を変更した場合は、電流換算ゲインも変更する必要があります。

```
#define CURRENT_GAIN_V_PER_A      0.00545f
```

この値は、1A あたり ADC 入力電圧が何 V 変化するかを表します。標準回路から抵抗値やアンプゲインを変更した場合は、必ず再計算してください。

1.8.7 電流制限設定

モータの巻線仕様、ギヤ比、車両重量、タイヤ径、バッテリー性能はユーザーごとに異なります。そのため、電流制限値は必ず各車両に合わせて設定してください。

設定箇所は Core/Src/current_limit.c です。

```
#define CURRENT_LIMIT_A      30.0f
#define CURRENT_TRIP_A      50.0f
```

CURRENT_LIMIT_A は、通常の電流制限を開始する電流値です。この値を超えると、プログラムは PWM デューティを下げて電流を抑えようとします。

CURRENT_TRIP_A は、重大な過電流と判断する電流値です。この値を超えると出力を停止し、異常状態を保持します。異常状態はアクセルを戻すまで解除されません。

初回走行時は低めの電流制限値から始め、モータ電流、FET 温度、モータ温度、バッテリー電圧降下を確認しながら段階的に調整してください。

1.8.8 低速ポーリング転流とホール割り込み転流

本プログラムでは、低速時と起動時の安定性を高めるため、ホール信号の読み取り方法を回転数に応じて切り替えています。

低速時は、TIM6 による 100 μ s 周期のポーリングでホール状態を確認し、ホール状態が変化した場合に転流します。回転が安定して一定以上の速度になると、ホールセンサのエッジ割り込みによる転流へ切り替えます。

設定箇所は Core/Src/motor_control.c です。

```
#define MOTOR_HALL_POLL_PERIOD_US      100U
#define MOTOR_POLL_TO_EXTI_TICKS      30U
#define MOTOR_POLL_TO_EXTI_COUNT_REQUIRED      6U
#define MOTOR_EXTI_TO_POLL_TIMEOUT_TICKS      50U
```

MOTOR_HALL_POLL_PERIOD_US は、ホール信号をポーリングする周期です。標準設定では 100 μ s です。

MOTOR_POLL_TO_EXTI_TICKS は、ポーリング転流から割り込み転流へ移行する判定に使うホールエッジ間隔です。

$$100\mu\text{s} \times 30 = 3\text{ms}$$

12 極モータの場合、機械 1 回転あたりのホールエッジ数は以下です。

$$6 \text{ 極対} \times 6 \text{ ステップ} = 36 \text{ エッジ/回転} \quad (3 \text{ 相ホールセンサ} + 6 \text{ ステップ整流の場合})$$

ホールエッジ間隔が 3ms 以下になる回転数は以下です。

$$1 / 0.003\text{s} = 333.3 \text{ エッジ/s}$$
$$333.3 / 36 \times 60 \doteq 556\text{rpm}$$

したがって、約 556rpm 以上の状態が 6 回連続すると、ポーリング転流からホール割り込み転流へ移行します。

一方、ホール割り込み転流中に一定時間ホールエッジが来ない場合は、低速と判断してポーリング転流へ戻ります。

$$100\mu\text{s} \times 50 = 5\text{ms}$$

この条件は 12 極モータの場合、約 333rpm に相当します。

$$1 / 0.005\text{s} = 200 \text{ エッジ/s}$$
$$200 / 36 \times 60 \doteq 333\text{rpm}$$

つまり、標準設定では以下のような動作になります。

約 333rpm 未満：ポーリング転流

約 333～556rpm：現在のモードを維持するヒステリシス領域

約 556rpm 以上：ホール割り込み転流

このヒステリシスにより、切り替え付近でポーリング転流と割り込み転流が頻繁に切り替わることを防いでいます。

1.8.9 TIM2 と TIM6 の役割

本プログラムでは、TIM2 と TIM6 を異なる目的で使用しています。

TIM2：5ms 周期の制御処理用

TIM6：100 μ s 周期の低速ホールポーリング用

TIM2 では、アクセル読み取り、電流制限、安全判定、デューティ更新などの制御処理を行います。

TIM6 では、低速時および起動時にホール状態を短周期で監視し、必要に応じて転流します。

64MHz 動作時の TIM6 設定例は以下です。

```
htim6.Init.Prescaler = 63;
```

```
htim6.Init.Period = 99;
```

この設定では、TIM6 の割り込み周期は 100 μ s になります。

$$64\text{MHz} / (63 + 1) = 1\text{MHz}$$
$$1\text{MHz} / (99 + 1) = 10\text{kHz}$$
$$\text{周期} = 100\mu\text{s}$$

TIM6 の Period が 0 になっていると、想定より非常に短い周期で割り込みが発生し、CPU 負荷増加や制御不安定の原因になります。必ず 99 になっていることを確認してください。

1.8.10 転流時のデッドタイム

転流時には、旧通電相を OFF してから新しい通電相を ON するまでに、ごく短い待ち時間を入れています。これは、出力切り替え時の重なりや不要な電流スパイクを防ぐためです。

設定箇所は Core/Src/motor_control.c です。

```
#define MOTOR_COMMUTATION_DEADTIME_NOP 20U
```

この値は、転流時に挿入する NOP 命令の回数です。通常は標準値のままで使用してください。変更する場合は、オシロスコープでゲート信号、相電流、電源電流を確認し、異常な電流スパイクが発生しないことを確認してください。

1.8.11 回転方向の設定

モータの相線接続、ホールセンサ配線、車両の駆動系により、正転方向が異なる場合があります。回転方向が意図と逆の場合は、Core/Src/motor_control.c 内のホール信号から転流ステップへの変換テーブルを確認してください。

主な設定箇所は以下です。

hall_to_step_forward

hall_to_step_reverse

車両として前進する方向を MOTOR_DIR_FORWARD として設定してください。変更後は、必ず低いデューティで無負荷確認を行い、意図した方向へ回転することを確認してから走行試験を行ってください。

1.8.12 初回走行前の確認項目

実車両で走行する前に、必ず以下を確認してください。

1. アクセル全閉時にモータが回らないこと
2. アクセル全閉時および全開時の ADC 値が想定範囲にあること
3. 電源投入時にアクセルを開いていると出力禁止になること
4. アクセルを一度全閉まで戻すと出力許可になること
5. モータが停止状態から確実に起動すること
6. モータの回転方向が車両の前進方向と一致していること
7. 低速回転時に異音や引っ掛かりがないこと
8. 電流検出値が 0A 付近で正しく補正されていること
9. 無負荷時に不自然なスパイク状電流が発生しないこと
10. 過電流時に電流制限またはトリップが動作すること
11. FET、シャント抵抗、モータ、配線、コネクタが異常発熱しないこと
12. タイヤを浮かせた状態で低デューティから動作確認すること
13. 初回走行時は電流制限値と最大デューティを低めに設定すること

本プログラムは車両ごとの最終設定を自動で保証するものではありません。安全な走行のため、各ユーザーが自分の車両仕様に合わせて設定値を確認・調整してください。

<<本キットに関する問い合わせ先>>

お問い合わせは、弊社オンライン・サポート・サイト

<https://interface.cqpub.co.jp/cqmotor2/>

パスワード kit2qw

からお願いいたします。

または、

CQ 出版(株)

Interface 編集部

E-mail: supportinter@cqpub.co.jp

TEL: 03-5395-2122

野村/永井/佐藤

まで。

本キット開発協力：小野塚精機株式会社



CQ インバータ・キット 2 取り扱い説明書（ソフトウェア編）

Ver. 01

2026 年 6 月 24 日 第 1 版発行

小野塚精機(株), CQ 出版社 2026

発行所 CQ 出版(株)

〒112-8619 東京都文京区千石 4-29-14

編集担当：野村 英樹