Examples

The examples are in the `examples` folder:

- Profile Velocity Mode - connect to an ESCON2, enable it and control the velocity from the keyboard.

Each example folder is a complete CMake project of its own, and the top-level `CMakeLists.txt` builds all of them at once. The sections below build the top-level project; the result is the same.

> **Important**
>
> Install the prerequisites first, and read the safety notes in the example's README before running anything - the examples turn the motor shaft.

Presets and output location

`CMakePresets.json` defines four configure presets: `x64-debug`, `x64-release`, `x86-debug`, `x86-release`. The preset decides the build folder and which Motion Library (64- or 32-bit) is used. The built example is

```
build\<preset>\examples\profile_velocity_mode\example_pvm.exe
```

with the Motion Library DLL copied next to it.

Using Visual Studio 2022 or 2026

Visual Studio opens CMake projects directly, without a solution file.

1. Start Visual Studio and choose **Open a local folder** (or *File → Open → Folder...*).
2. Select the **root folder** of the archive - the one that contains `CMakeLists.txt` and `CMakePresets.json`. Visual Studio reads the presets and configures the first one; the output window shows the CMake messages.
3. Select the configuration in the presets dropdown of the toolbar, e.g. *x64 Release*.
4. *Build → Build All* (`Ctrl+Shift+B`). The executable is in `build\<preset>\...` as described above; *Debug → Start Without Debugging* runs it in a console window.

If Visual Studio reports "Rebuild failed" without any error text after you switched presets or deleted a build folder by hand, the preset has no configured cache yet: run *Project → Configure Cache* once, then build.

Using Visual Studio Code

1. Install the extension CMake Tools.
2. Start VS Code **from a Developer PowerShell for VS** (`code .` in the archive's root folder). The presets pin the compiler to `cl.exe`, which is only on the PATH in that shell.
3. Press `F1` and run - `CMake: Select Configure Preset` to choose the preset, - `CMake: Build` to build, - `CMake: Run Without Debugging` to run.
4. To debug, create `.vscode/launch.json` with this configuration; it starts the example that CMake Tools has selected as launch target:

```
{
  "version": "0.2.0",
  "configurations": [
    {
      "name": "Example (external terminal)",
      "type": "cppvsdbg",
      "request": "launch",
      "program": "${command:cmake.launchTargetPath}",
      "cwd": "${command:cmake.launchTargetDirectory}",
      "console": "externalTerminal"
    }
  ]
}
```

Using the command line

The presets need the compiler on the PATH. Open a **Developer PowerShell for VS 2022** (or 2026) from the Start menu for the `x64` presets, or an **x86 Native Tools Command Prompt for VS** for the `x86` presets, then:

```
cd C:\dev\motion_helper
cmake --preset x64-release
cmake --build build\x64-release
build\x64-release\examples\profile_velocity_mode\example_pvm.exe
```

If the configure step fails with `The CMAKE_C_COMPILER: cl.exe is not a full path`, the shell is not a Developer shell. Delete the `build\` folder before retrying; the failed run leaves a cache that later runs would reuse. If it fails with "The compiler on PATH targets 64-bit, but this preset expects 32-bit", the preset and the shell do not match.