**Case Study:**

**OSRM Project**

Tables of Contents

[**I.** **Introduction** 2](#_Toc167843244)

[**II.** **Building & Running OSRM from Source** 2](#_Toc167843245)

[**1.** **Device Requirements** 2](#_Toc167843246)

[**2.** **Building OSRM Backend** 3](#_Toc167843247)

[**2.1. Download sources code from Git** 3](#_Toc167843248)

[**2.2. Install Dependencies** 3](#_Toc167843249)

[**2.3. Building OSRM Backend by CMake in Visual Studio** 5](#_Toc167843250)

[**3.** **Running OSRM Backend – Create local route server in windows** 8](#_Toc167843251)

[**III.** **Issues recorded and solutions** 9](#_Toc167843252)

1. **Introduction**

Open Source Routing Machine (OSRM) is high performance routing engine written in C++ designed to run on OpenStreetMap data.

It contains:

- OSRM backend writing in C++ for Routing Server

- OSRM frontend for User-facing frontend with map

You can read more at: https://github.com/Project-OSRM/osrm-backend/wiki

This document provides the guide (steps by steps) for building and running simple OSRM integration in local machine (Windows 10 64bit).

Some issues during installation & its solutions are also recorded in this document. It also contains a simple demo run guide in local.

**This document contains 2 main parts:**

* + - * **Building & Running OSRM in Windows**
      * **Issues Recorded & Solutions during process**

1. **Building & Running OSRM from Source**

There are 2 ways to building OSRM: by Docker or build from source. In this case study, we choose build from source to .exe and .dll file – that can be run in Windows.  
For other OS, you can follow the instructions [here](https://github.com/Project-OSRM/osrm-backend/wiki/Building-OSRM).

* 1. **Device Requirements**

The device for building OSRM must install some apps & features before start:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Version | Link Download | Description |
| Windows 10 64 bit (or newer) | 10 or newer |  | OS |
| Microsoft Visual Studio 2015 with Microsoft Visual C++ 2015 Redistributable(x64) | 2015 or latest (in demo using 2022 Preview) | https://visualstudio.microsoft.com/ | For compile and running C++ |
| CMake | Minimum required version is 3.7.0 with Boost 1.62 version supported in FindBoost.cmake | https://cmake.org/download/ | For building source |
| Vcpkg | Latest | https://github.com/microsoft/vcpkg | For install dependencies |
| NodeJs | 4.x LTS | https://nodejs.org/en | Running OSRM Frontend |
| Python | Latest |  | For install dependencies by Conan |

* 1. **Building OSRM Backend**

The OSRM Backend is written in C++ and building by CMake. For summary, these steps can break down to 3 main steps:

* + - * Download and extract sources from Git
      * Install Dependencies by Vcpkg
      * Building project by Cmake in Visual Studio

Let’s start.

### **2.1. Download sources code from Git**

If you’ve installed Git, you can checkout by these commands in Git Bash:

git clone https://github.com/Project-OSRM/osrm-backend.git

cd osrm-backend

# optionally checkout a particular revision/tag

git checkout v5.27.1

If you would like to build a specific release (optional), you can list the release tags with git tag -l and then checkout a specific tag using git checkout tags/<tag\_name>.

By the way, if you don’t have Git in your device, you can simplely download source code from this link: <https://github.com/Project-OSRM/osrm-backend/archive/refs/tags/v5.27.1.zip>

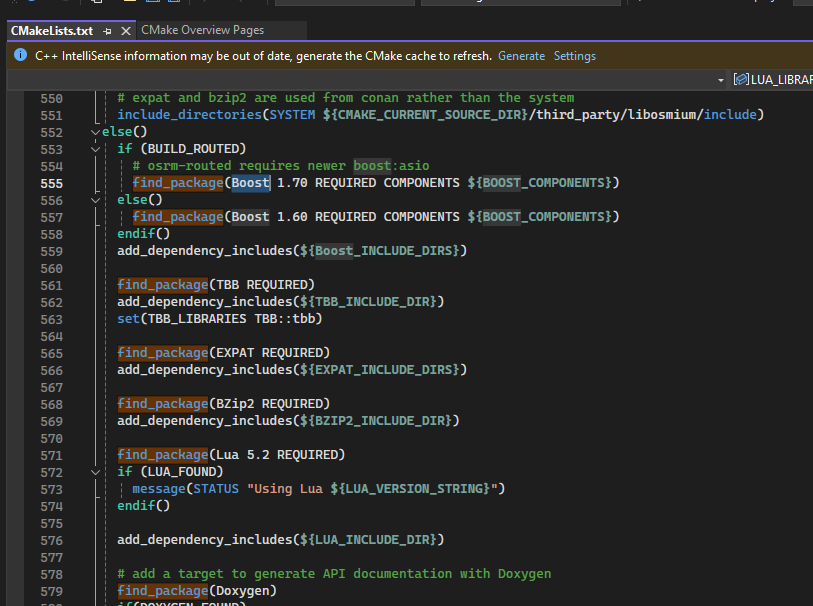
Then extract zip files to destination folder.

### **2.2****.** **Install Dependencies**

OSRM Backend Project need some libraries installed in your devices:

* + - * Boost
      * TBB
      * EXPAT
      * BZip2 (included if install boost by vcpkg)
      * Lua
      * Doxygen
      * Osmium
      * ZLIB (included if install boost by vcpkg)

If you have not installed these dependencies before Cmake build, you can have the [**Error1**](#Error1).

These dependencies listed in CMakeLists.txt, as shown below  


We will install all dependencies listed above by Vcpkg – a free C/C++ package manager.

Here are steps

* + 1. Check out and install vcpkg from <https://github.com/microsoft/vcpkg>
       - Download vcpkg: <https://github.com/microsoft/vcpkg/archive/refs/heads/master.zip>  
         (Update=> <https://github.com/Neumann-A/vcpkg/archive/refs/heads/boost-fix-hash.zip> due to [**Error2**](#Error2))
       - Extract zip to vcpkg folder
       - Run file bootstrap-vcpkg.bat
    2. Install all dependencies: Runs these commands in cmd:

vcpkg install Boost

vcpkg install Boost

vcpkg install TBB

vcpkg install EXPAT

vcpkg install Doxygen

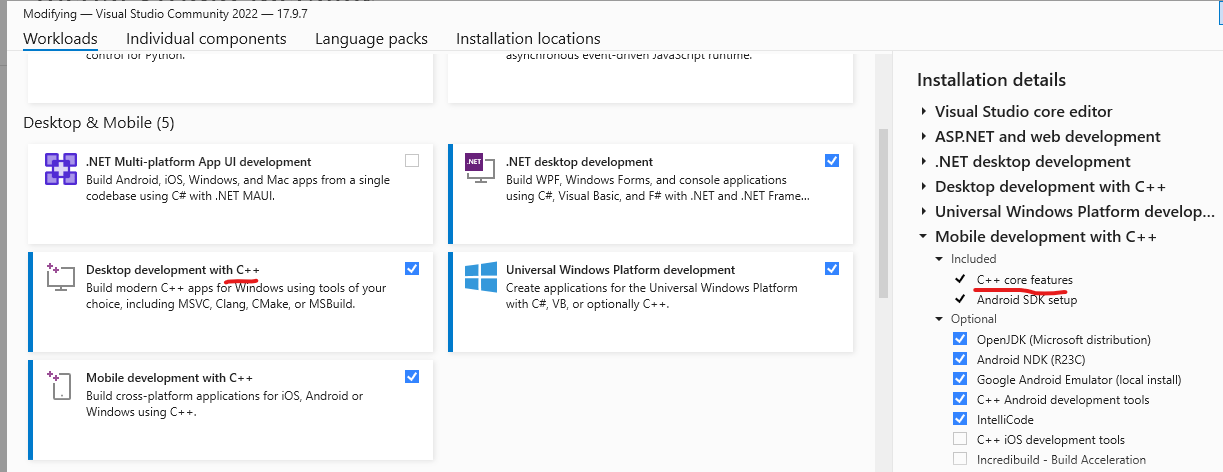
vcpkg install Osmium

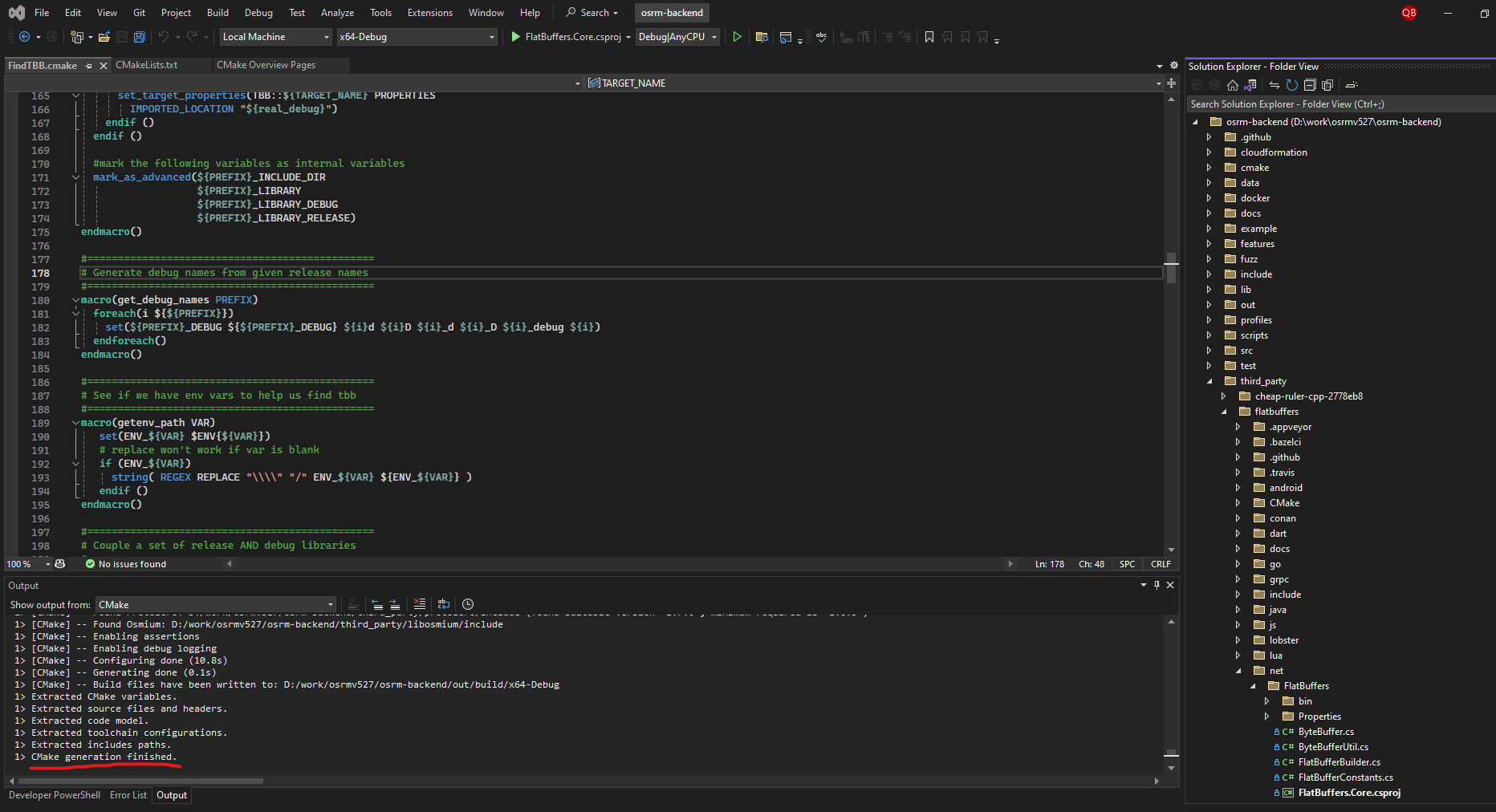
You may have the [**Error2**](#Error2) -> Check and try the solutions.

* + 1. Integrate vcpkg with Visual Studio: Run in cmd(administrator)

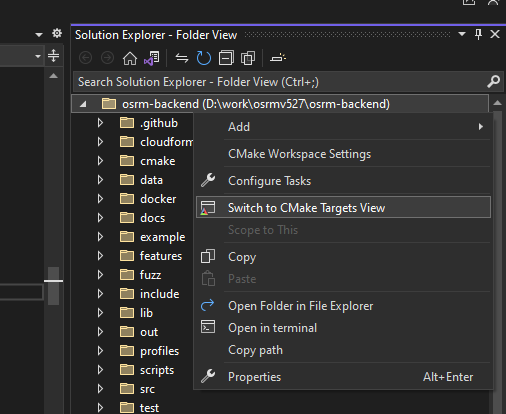
vcpkg integrate install

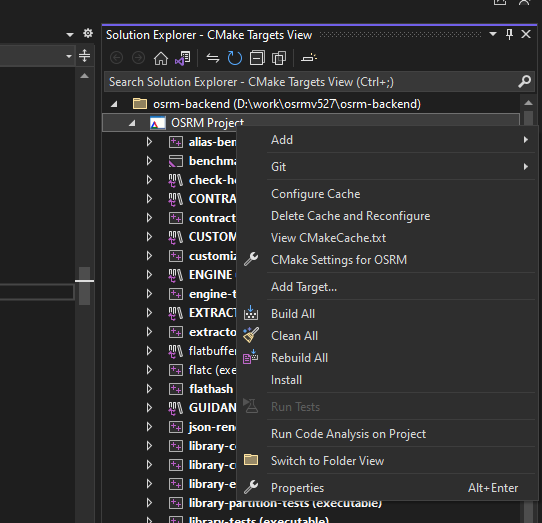
### **2.3. Building OSRM Backend by CMake in Visual Studio**

* **I**nstall Visual Studio 2022 Community with C++ features
* Choose Open a local folder and choose OSRM Backend Source folder
* After Visual Studio open source folder -> Cmake will automatic generate source to CMake Project – the successfully shown as below:



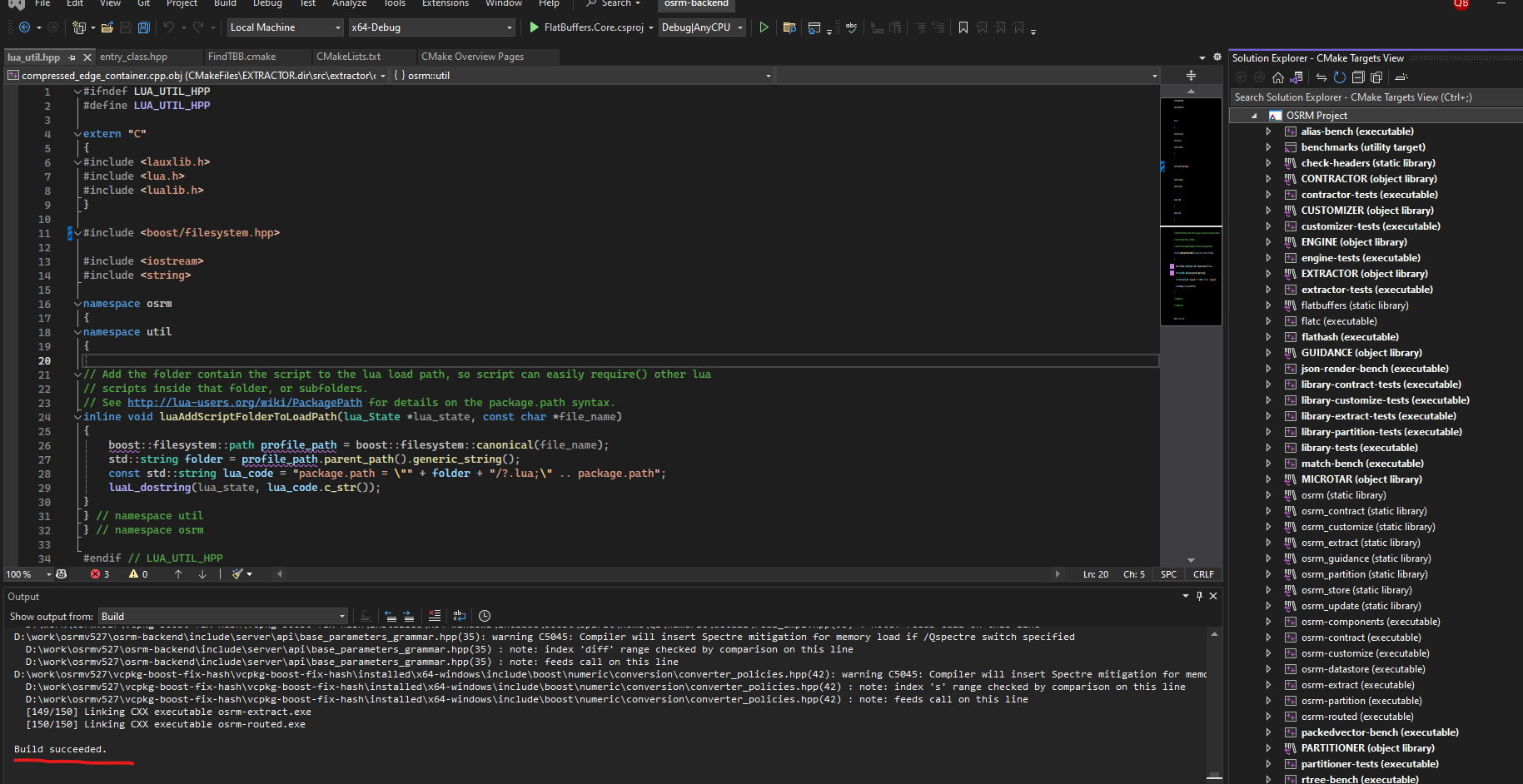
You may also have the [Error3](#Error3) -> Check and try the solutions

* Right click at project -> Choose Switch to CMake Targets View  
  
* In Cmake Targets View, right click at OSRM Project -> Build All

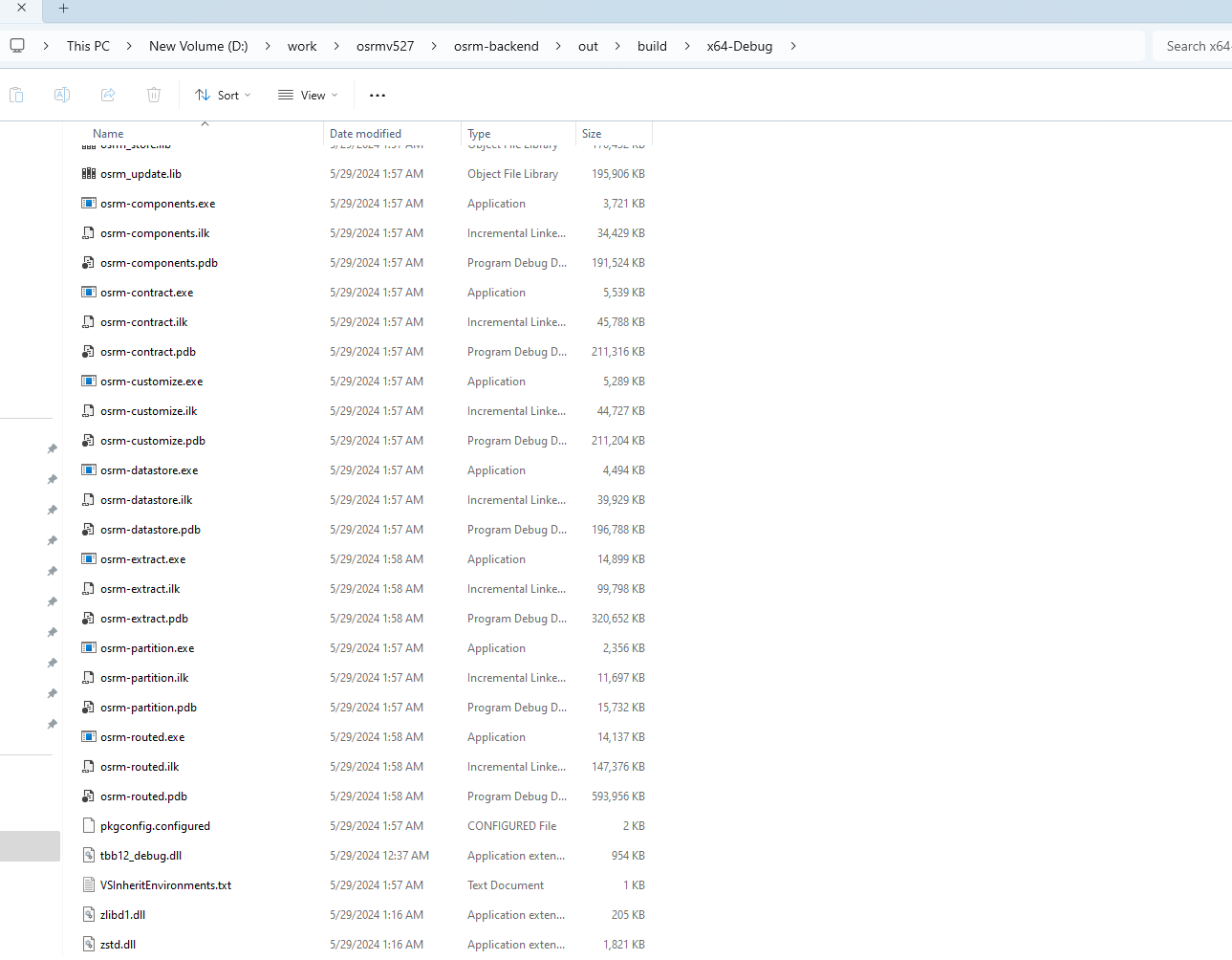


You may have [**Error4**](#Error4)in this step -> Check solutions

The successfully build shown as below:

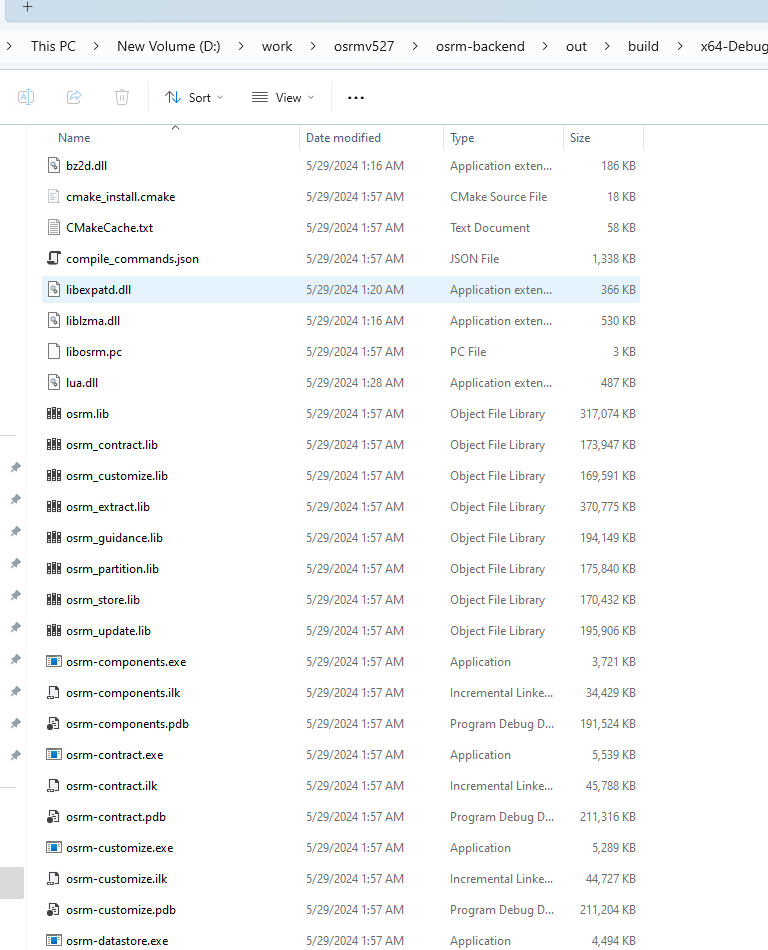


Check the output folder build: .\osrm-backend\out\build\x64-Debug



* 1. **Running OSRM Backend – Create local route server in windows**

After Building OSRM Backend, you already had the build folder contains:



Open CMD in that folder and follow instructions from here: <https://github.com/Project-OSRM/osrm-backend/wiki/Running-OSRM>.

There are 2 pre-processing pipelines available: Contraction Hierarchies (CH) and Multi-Level Dijkstra (MLD) (recommended)

Here are details steps:

* Download map from: <http://download.geofabrik.de/asia/south-korea-latest.osm.pbf> (You can also download other map from: <http://download.geofabrik.de>, but check the [Disk-and-Memory-Requirements](https://github.com/Project-OSRM/osrm-backend/wiki/Disk-and-Memory-Requirements) before execute map) and put it in build folder.
* For MLD pipelines: Go to build folder and type these commands in cmd:

osrm-extract south-korea-latest.osm.pbf

osrm-partition south-korea-latest.osrm

osrm-customize south-korea-latest.osrm

osrm-routed --algorithm=MLD south-korea-latest.osrm

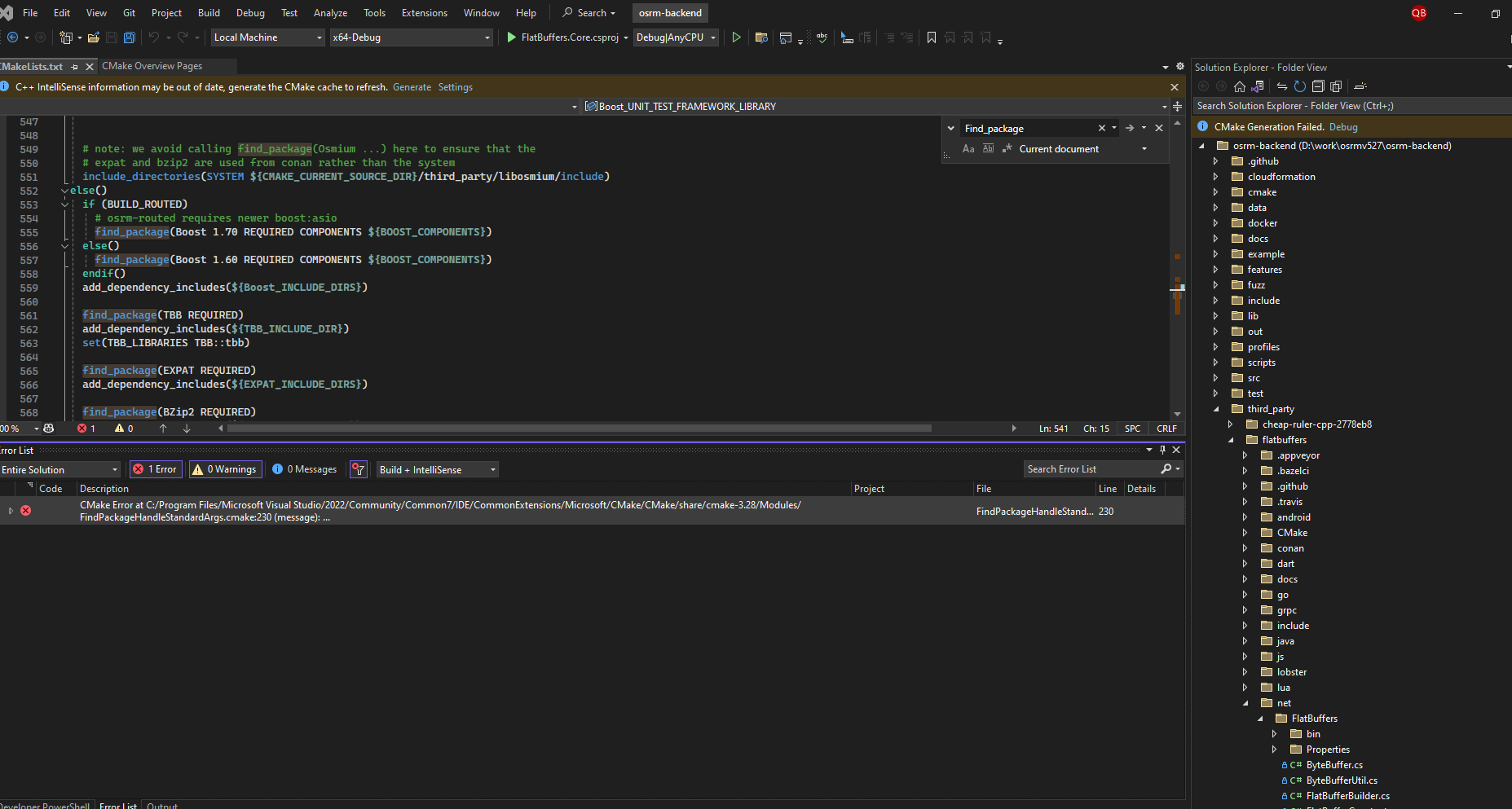
You may also have Error5 -> Check solutions below

1. **Issues recorded and solutions**

*Error1*: Could NOT find Boost (missing: Boost\_INCLUDE\_DIR date\_time chrono

filesystem iostreams program\_options regex system thread

unit\_test\_framework) (Required is at least version "1.70")



**Reason**: Cmake can not find dependencies (Boost, …) installed in Windows.

**Solutions**: Install all dependencies in windows by vcpkg and linked it to visual studio (Follow steps provided in [2.2**. Install Dependencies**](#InstallD))

*Error2*: Vcpkg install boost Error

**Message**: File does not have the expected hash: url: [https://github.com/boostorg/core/archive/boost-1.85.0.tar.gz](vscode-file://vscode-app/c:/Users/Admin/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html)

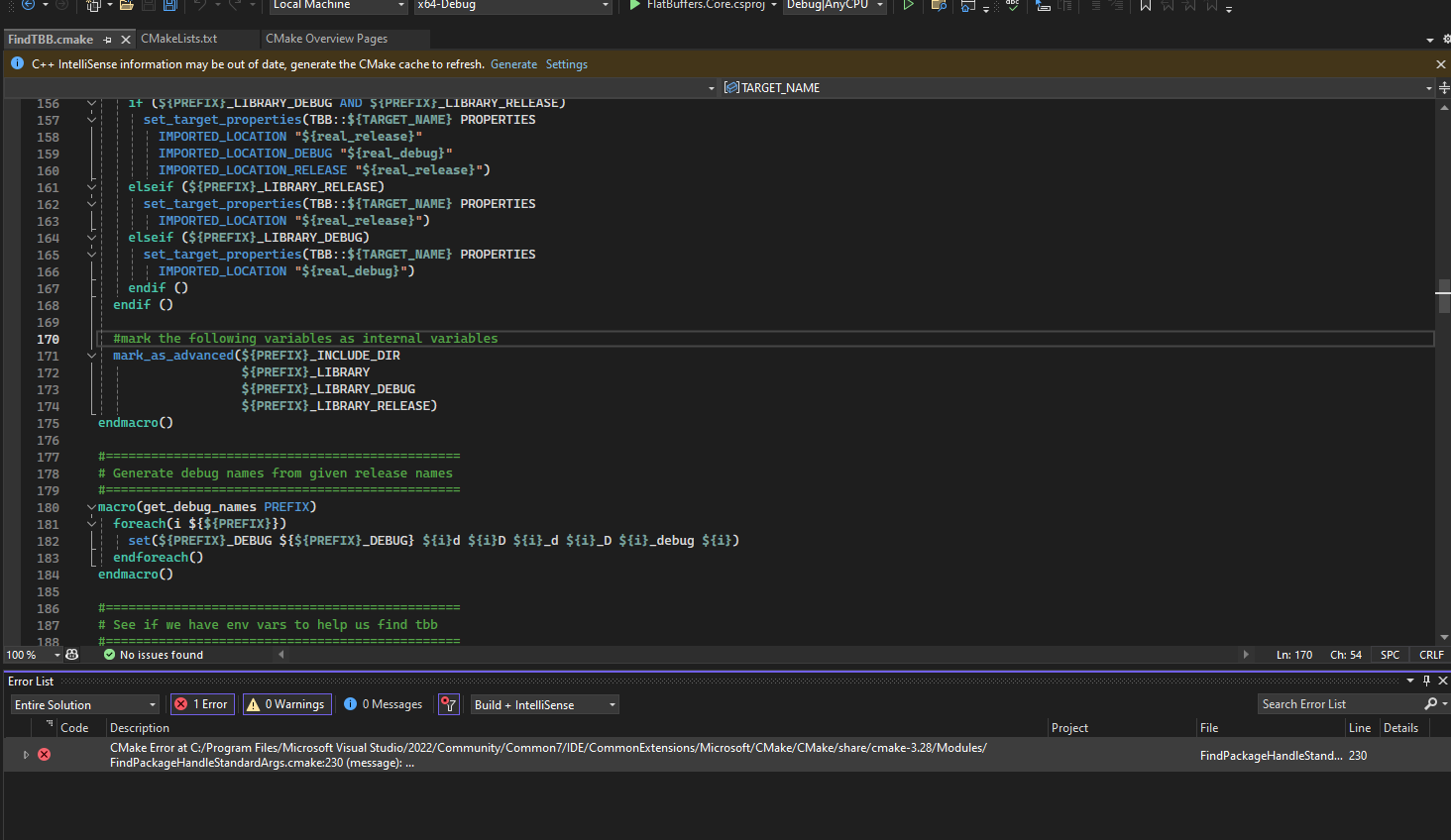
**Reason**: In master branch of vcpkg -> Same bugs with hash value in vcpkg, listed in  
<https://github.com/microsoft/vcpkg/issues/38974>

**Solutions**: Checkout from branch:

<https://github.com/Neumann-A/vcpkg/tree/boost-fix-hash>

and reinstall vcpkg & boost

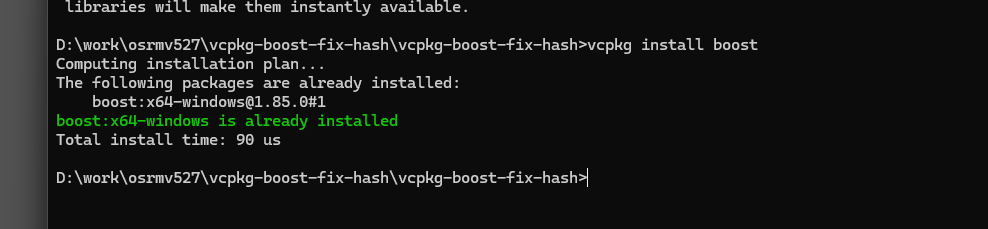
*Error3*: Libraries installed by vcpkg but CMake in Visual Studio not found

  
**Message**: Could NOT find Boost (missing: Boost\_INCLUDE\_DIR date\_time chrono

filesystem iostreams program\_options regex system thread

unit\_test\_framework) (Required is at least version "1.70")

Check the vcpkg, Boost is installed:



**Reason**: Cmake Build using CmakeCache.txt for fast building, so the building error is cache from last error build

**Solution**: delete folder build(.\osrm-backend\out\build) and reopen project by visual studio

*Error4***: Cannot open include file: 'boost/filesystem/convenience.hpp': No such file or directory**

This Error Occurs when building project by CMake in Visual Studio.  
  
**Reason**: The convenience.hpp header file was part of the Boost Filesystem library in older versions of Boost. However, it has been deprecated and removed in more recent versions of Boost.

**Solutions**: if we are installed latest Boost, just find and replace:   
// Old code

//#include <boost/filesystem/convenience.hpp>

// New code

#include <boost/filesystem.hpp>

