Something New in Parallel Program

Something New in Parallel Program
- Requirements in Windows
  - compiler
    - install MinGW
    - Better way to install?
      - apt on Ubuntu
      - Chocolatey on Windows
    - Use your gcc to compile your code on cmd
    - Edit your code by something
      - vim
      - Visual Studio Code
  - make
    - install make
    - use make to help compile
    - Why we need make
    - Write your first parallel program.
  - ~~cmake~~
    - install cmake on Windows
    - install cmake on Linux
    - Usage on windows.
- Requirements in Linux
- Download and install OpenBLAS
  - git
  - Compile
  - What is a 'libopenblas.a' file

Requirements in Windows

compiler

install MinGW

~~cygwin also works~~

Visit https://sourceforge.net/projects/mingw/files/ to download MinGW.

Or from official website http://www.mingw.org/ Also works.

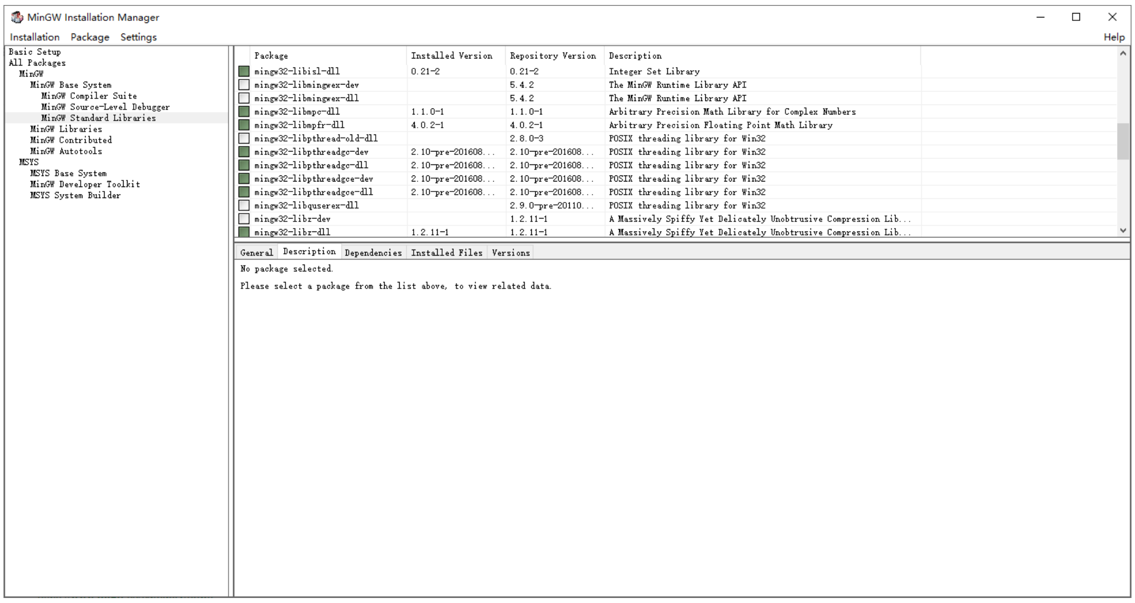
You can follow this Chinese website to learn how to install and set environment .

- **Achtung!**

  To run a parallel program latter , we need to install thread's library or you will find some thing like this

  ```
  gcc main.c -o main -fopenmp
  c:/mingw/bin/../lib/gcc/mingw32/9.2.0/../../../../mingw32/bin/ld.exe: cannot
  find -lpthread
  collect2.exe: error: ld returned 1 exit status
  make: *** [all] 错误 1
  ```

- Use MinGW Installation Manager and mark the lib you need to install.



	Package	Installed Version	Repository Version	Description
🟩	mingw32-libisl-dll	0.21-2	0.21-2	Integer Set Library
☐	mingw32-libmingwex-dev		5.4.2	The MinGW Runtime Library API
☐	mingw32-libmingwex-dll		5.4.2	The MinGW Runtime Library API
🟩	mingw32-libmpc-dll	1.1.0-1	1.1.0-1	Arbitrary Precision Math Library for Complex Numbers
🟩	mingw32-libmpfr-dll	4.0.2-1	4.0.2-1	Arbitrary Precision Floating Point Math Library
☐	mingw32-libpthread-old-dll		2.8.0-3	POSIX threading library for Win32
🟩	mingw32-libpthreadgc-dev	2.10-pre-201608...	2.10-pre-201608...	POSIX threading library for Win32
🟩	mingw32-libpthreadgc-dll	2.10-pre-201608...	2.10-pre-201608...	POSIX threading library for Win32
🟩	mingw32-libpthreadgce-dev	2.10-pre-201608...	2.10-pre-201608...	POSIX threading library for Win32
🟩	mingw32-libpthreadgce-dll	2.10-pre-201608...	2.10-pre-201608...	POSIX threading library for Win32
☐	mingw32-libquserex-dll		2.9.0-pre-20110...	POSIX threading library for Win32
☐	mingw32-libz-dev		1.2.11-1	A Massively Spiffy Yet Delicately Unobtrusive Compression Lib...
🟩	mingw32-libz-dll	1.2.11-1	1.2.11-1	A Massively Spiffy Yet Delicately Unobtrusive Compression Lib...

No package selected.

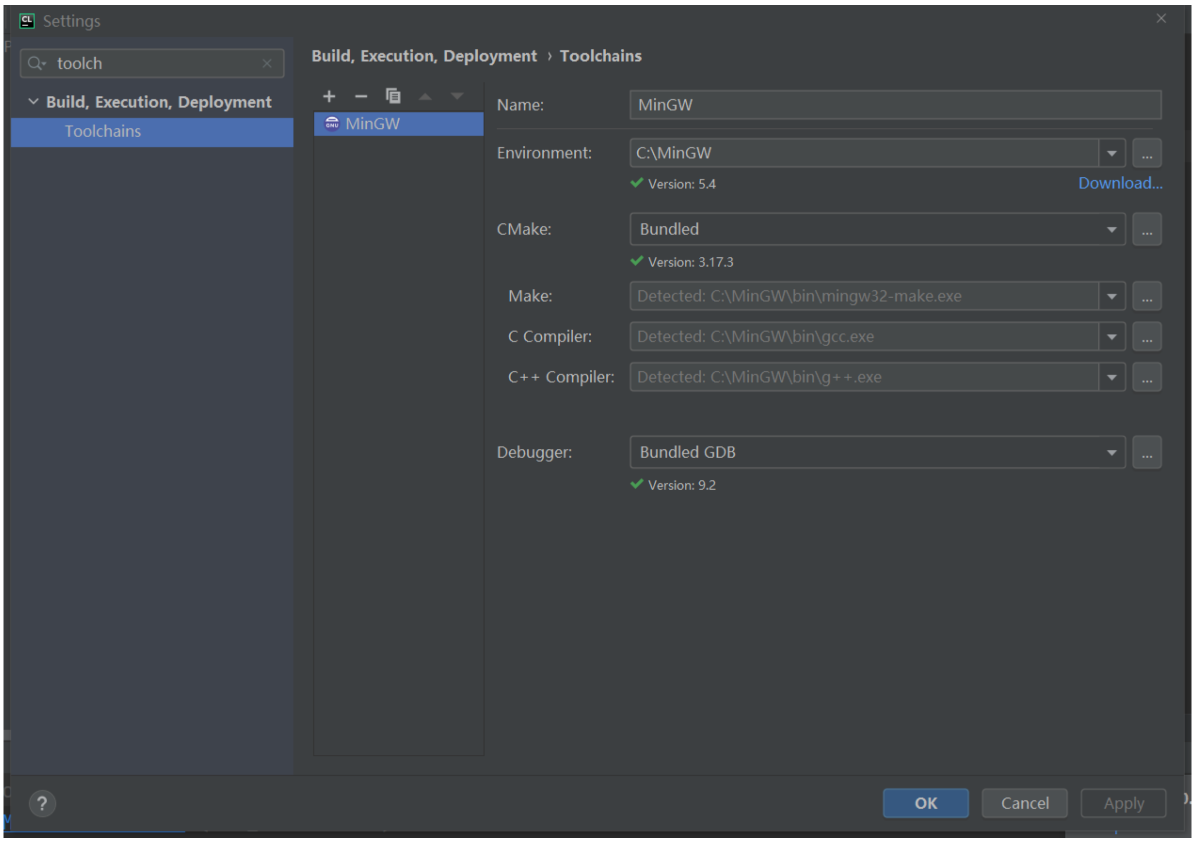
Please select a package from the list above, to view related data.

- Other lib cannot be found ?

  stackoverflow helps you out.

Source code need a compiler to be compiled to executable file.

If you are familiar enough with your IDE , you can find where your compiler located easily.


Settings
toolch
Build, Execution, Deployment
Toolchains
Build, Execution, Deployment › Toolchains
MinGW
Download...
OK
Cancel
Apply

Name: MinGW

Environment: C:\MinGW
✓ Version: 5.4

CMake: Bundled
✓ Version: 3.17.3

Make: Detected: C:\MinGW\bin\mingw32-make.exe

C Compiler: Detected: C:\MinGW\bin\gcc.exe

C++ Compiler: Detected: C:\MinGW\bin\g++.exe

Debugger: Bundled GDB
✓ Version: 9.2

Set bin/ in environment and open cmd/terminal to test.

Better way to install?

apt on Ubuntu

If you have a Ubuntu

```
sudo apt install gcc
sudo apt install g++
```

Chocolatey on Windows

Windows also has similar package tool like apt in Ubuntu(a linux)

Different from linux , we need to downloads this package manager by ourselves.

- Official website tells how to install
- This article explains why tools like Chocolatey is in need.

~~Me? of course I didn't use , I got a Ubuntu to use.~~

Use your gcc to compile your code on cmd

We write a hello world program before.

```
#include <stdio.h>

int main()
{
    printf("Hello, World! \n");
    return 0;
}
```

And put it on Desktop and use a power shell or cmd to do this.

```
PS C:\Users\Kouushou\Desktop> notepad main.c
PS C:\Users\Kouushou\Desktop> gcc main.c -o main
PS C:\Users\Kouushou\Desktop> .\main.exe
Hello, World!
PS C:\Users\Kouushou\Desktop>
```

You can compile your code in cmd now.

Edit your code by something

vim

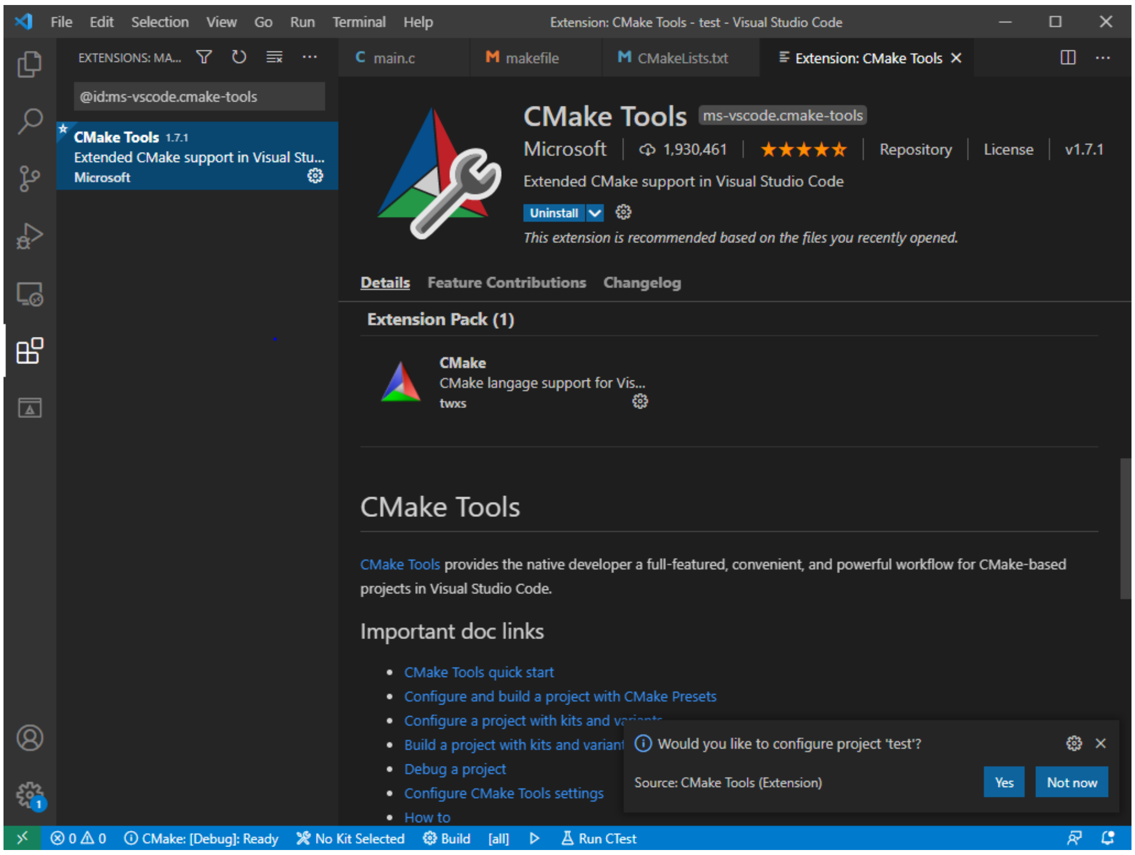
- Download vim and use skills of set environment to let your cmd/power shell/terminal to know vim.
- Learn to use vim , vim on windows is the same as linux's

Visual Studio Code

- Download VS Code , and install ~~installer will set every thing for you~~
- after you install try this

  ```
  PS C:\Users\Kouushou\Desktop> code .
  ```

- Visual Studio Code has many Extensions to help you out



Extension: CMake Tools - test - Visual Studio Code

@id:ms-vscode.cmake-tools

CMake Tools 1.7.1
Extended CMake support in Visual Stu...
Microsoft

CMake Tools ms-vscode.cmake-tools
Microsoft | 1,930,461 | ★★★★★ | Repository | License | v1.7.1
Extended CMake support in Visual Studio Code
This extension is recommended based on the files you recently opened.

Extension Pack (1)

CMake
CMake langage support for Vis...
twxs

CMake Tools

CMake Tools provides the native developer a full-featured, convenient, and powerful workflow for CMake-based projects in Visual Studio Code.

Important doc links

- CMake Tools quick start
- Configure and build a project with CMake Presets
- Configure a project with kits and variants
- Build a project with kits and variant
- Debug a project
- Configure CMake Tools settings
- How to

Would you like to configure project 'test'?
Source: CMake Tools (Extension)

⊗ 0 ⚠ 0 CMake: [Debug]: Ready No Kit Selected Build [all] Run CTest

With this extension we can find the highlight of CMakeLists.txt

make

install make

MinGW installed before contains make called mingw32-make , but install a new make is still in need.

Visit make for windows to find and download a make.

If you have Chocolatey installed , open your power shell and type

```
choco install make
```

use make to help compile

Write our first makefile

```
all:
    gcc main.c -o main
```

Open a power shell to use

- use 'ls' to show files

```
PS C:\Users\Kouushou\Desktop\test> ls


    目录: C:\Users\Kouushou\Desktop\test


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----         2021/4/28     13:56             87 main.c
-a----         2021/4/28     14:18             27 makefile
```

- type in make and file can be compile

```
PS C:\Users\Kouushou\Desktop\test> make
gcc main.c -o main
PS C:\Users\Kouushou\Desktop\test> .\main.exe
Hello, World!
PS C:\Users\Kouushou\Desktop\test>
```

Why we need make

If we need to compile a project with many source files and include many library , we need to type

```
gcc main.c test.c -I C:\include\bench\include -L C:\Users\Temp\Lib -lmkl_rt -lm
-lpthread -fopenmp -o main
```

But once you write a makefile just type

```
make
```

Also makefile can find requirements smartly

```
Compile_Exe:Lib
    #compile Exe is needing compile a Lib first
Lib:
    #compile a Lib
```

In this situation

```
make Compile_Exe    #compile Lib first, then Exe
make Lib            #just compile a Lib
```

notice makefile has a strict grammar , use search engine to learn.

Write your first parallel program.

Prepare a C code of multi-thread

```
#include <stdio.h>
#include <omp.h>
int main()
{
#pragma omp parallel for
    for(int i = 0 ; i < 10 ; ++i){
        printf("Hello, World! by %d\n",i);
    }
    return 0;
}
```

Change makefile

```
all:
    gcc main.c -o main -fopenmp
```

use make and run main.exe

```
PS C:\Users\Kouushou\Desktop\test> make
gcc main.c -o main -fopenmp
PS C:\Users\Kouushou\Desktop\test> .\main.exe
Hello, World! by 1
Hello, World! by 4
Hello, World! by 2
Hello, World! by 8
Hello, World! by 9
Hello, World! by 7
Hello, World! by 6
Hello, World! by 0
Hello, World! by 3
Hello, World! by 5
```

This is a multi-thread program.

~~cmake~~

install cmake on Windows

- click to download
- install

install cmake on Linux

```
sudo apt install cmake
```

Usage on windows.

Unfortunately , I didn't let cmake to run properly on my laptop, if any fellow successfully use cmake to compile your code on windows . Let me know.

- cmake tutorials
- ~~We will use it later~~

Requirements in Linux

Strongly suggest to get a linux environment , install it on your hardware , or rent a server even vmware. no matter using java python or cuda , linux do better ~~if you are not doing well though the installation before~~.

```
sudo apt install g++
sudo apt install gcc
sudo apt install make
sudo apt install cmake
```

and you can do most thing before

Download and install OpenBLAS

git

- Official of git
- using git to manage your code
- https://github.com/ is the biggest open source platform
- In our land https://gitee.com/ is faster than git
- git tutorial

-
  ```
  git clone https://github.com/xianyi/OpenBLAS
  ```

  this command to donwload OpenBLAS

Compile

- Use cmake to Compile OpenBLAS
  - Change directory to OpenBLAS
  - cmake will generate many files once we use command

    ```
    cmake
    ```

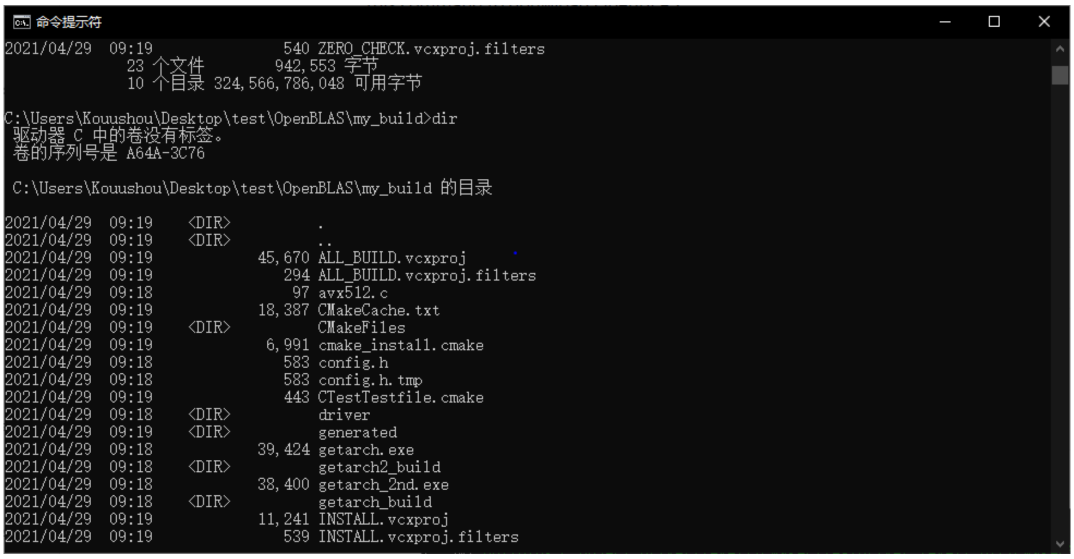
  - We need to use a new directory to make origin clean

    ```
    C:\Users\Kouushou\Desktop\test\OpenBLAS>mkdir my_build

    C:\Users\Kouushou\Desktop\test\OpenBLAS>cd my_build

    C:\Users\Kouushou\Desktop\test\OpenBLAS\my_build>cmake ..
    ```

  - And then generate many files



```
2021/04/29  09:19               540 ZERO_CHECK.vcxproj.filters
              23 个文件        942,553 字节
              10 个目录 324,566,786,048 可用字节

C:\Users\Kouushou\Desktop\test\OpenBLAS\my_build>dir
 驱动器 C 中的卷没有标签。
 卷的序列号是 A64A-3C76

 C:\Users\Kouushou\Desktop\test\OpenBLAS\my_build 的目录

2021/04/29  09:19    <DIR>          .
2021/04/29  09:19    <DIR>          ..
2021/04/29  09:19            45,670 ALL_BUILD.vcxproj
2021/04/29  09:19               294 ALL_BUILD.vcxproj.filters
2021/04/29  09:18                97 avx512.c
2021/04/29  09:19            18,387 CMakeCache.txt
2021/04/29  09:19    <DIR>          CMakeFiles
2021/04/29  09:19             6,991 cmake_install.cmake
2021/04/29  09:18               583 config.h
2021/04/29  09:18               583 config.h.tmp
2021/04/29  09:19               443 CTestTestfile.cmake
2021/04/29  09:18    <DIR>          driver
2021/04/29  09:19    <DIR>          generated
2021/04/29  09:18            39,424 getarch.exe
2021/04/29  09:18    <DIR>          getarch2_build
2021/04/29  09:18            38,400 getarch_2nd.exe
2021/04/29  09:18    <DIR>          getarch_build
2021/04/29  09:19            11,241 INSTALL.vcxproj
2021/04/29  09:19               539 INSTALL.vcxproj.filters
```

- As we can see cmake generating many files associate with **Visual Studio** , but we need makefile
- If you get your compiler and make ready , then you can use

  ```
  cmake -G "MinGW Makefiles" ..
  ```

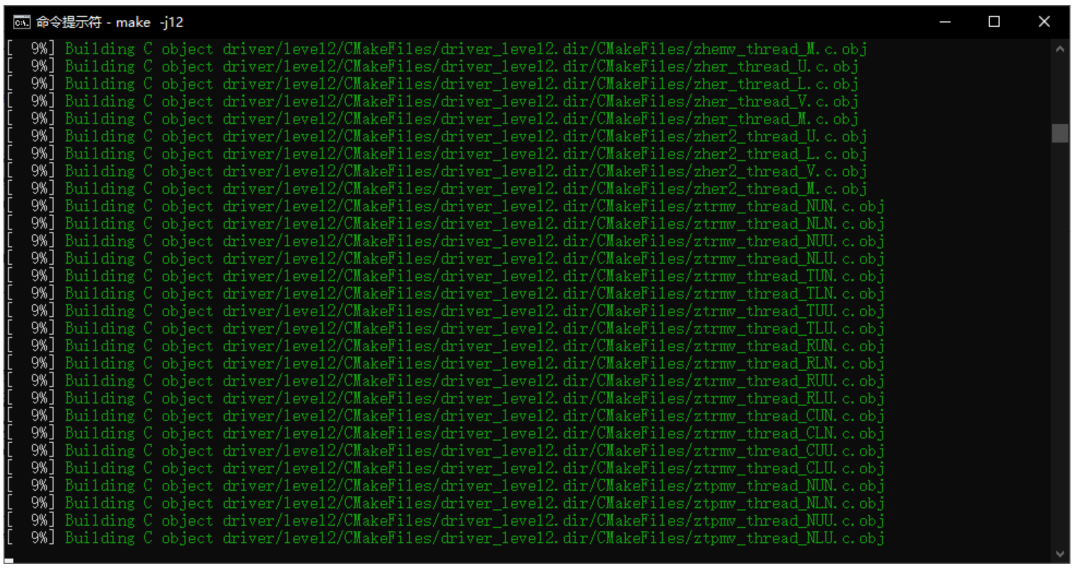
  to generate mingw Makefiles

  - **You might need fortran compiler in this step ,go back to *MinGW Installation Manager* to install**
- Then

  ```
  make
  ```

  or

  ```
  make -j12 #using multy-thread to compile faster
  ```



```
[  9%] Building C object driver/level2/CMakeFiles/driver_level2.dir/CMakeFiles/zhemv_thread_M.c.obj
[  9%] Building C object driver/level2/CMakeFiles/driver_level2.dir/CMakeFiles/zher_thread_U.c.obj
[  9%] Building C object driver/level2/CMakeFiles/driver_level2.dir/CMakeFiles/zher_thread_L.c.obj
[  9%] Building C object driver/level2/CMakeFiles/driver_level2.dir/CMakeFiles/zher_thread_V.c.obj
[  9%] Building C object driver/level2/CMakeFiles/driver_level2.dir/CMakeFiles/zher_thread_M.c.obj
[  9%] Building C object driver/level2/CMakeFiles/driver_level2.dir/CMakeFiles/zher2_thread_U.c.obj
[  9%] Building C object driver/level2/CMakeFiles/driver_level2.dir/CMakeFiles/zher2_thread_L.c.obj
[  9%] Building C object driver/level2/CMakeFiles/driver_level2.dir/CMakeFiles/zher2_thread_V.c.obj
[  9%] Building C object driver/level2/CMakeFiles/driver_level2.dir/CMakeFiles/zher2_thread_M.c.obj
[  9%] Building C object driver/level2/CMakeFiles/driver_level2.dir/CMakeFiles/ztrmv_thread_NUN.c.obj
[  9%] Building C object driver/level2/CMakeFiles/driver_level2.dir/CMakeFiles/ztrmv_thread_NLN.c.obj
[  9%] Building C object driver/level2/CMakeFiles/driver_level2.dir/CMakeFiles/ztrmv_thread_NUU.c.obj
[  9%] Building C object driver/level2/CMakeFiles/driver_level2.dir/CMakeFiles/ztrmv_thread_NLU.c.obj
[  9%] Building C object driver/level2/CMakeFiles/driver_level2.dir/CMakeFiles/ztrmv_thread_TUN.c.obj
[  9%] Building C object driver/level2/CMakeFiles/driver_level2.dir/CMakeFiles/ztrmv_thread_TLN.c.obj
[  9%] Building C object driver/level2/CMakeFiles/driver_level2.dir/CMakeFiles/ztrmv_thread_TUU.c.obj
[  9%] Building C object driver/level2/CMakeFiles/driver_level2.dir/CMakeFiles/ztrmv_thread_TLU.c.obj
[  9%] Building C object driver/level2/CMakeFiles/driver_level2.dir/CMakeFiles/ztrmv_thread_RUN.c.obj
[  9%] Building C object driver/level2/CMakeFiles/driver_level2.dir/CMakeFiles/ztrmv_thread_RLN.c.obj
[  9%] Building C object driver/level2/CMakeFiles/driver_level2.dir/CMakeFiles/ztrmv_thread_RUU.c.obj
[  9%] Building C object driver/level2/CMakeFiles/driver_level2.dir/CMakeFiles/ztrmv_thread_RLU.c.obj
[  9%] Building C object driver/level2/CMakeFiles/driver_level2.dir/CMakeFiles/ztrmv_thread_CUN.c.obj
[  9%] Building C object driver/level2/CMakeFiles/driver_level2.dir/CMakeFiles/ztrmv_thread_CLN.c.obj
[  9%] Building C object driver/level2/CMakeFiles/driver_level2.dir/CMakeFiles/ztrmv_thread_CUU.c.obj
[  9%] Building C object driver/level2/CMakeFiles/driver_level2.dir/CMakeFiles/ztrmv_thread_CLU.c.obj
[  9%] Building C object driver/level2/CMakeFiles/driver_level2.dir/CMakeFiles/ztpmv_thread_NUN.c.obj
[  9%] Building C object driver/level2/CMakeFiles/driver_level2.dir/CMakeFiles/ztpmv_thread_NLN.c.obj
[  9%] Building C object driver/level2/CMakeFiles/driver_level2.dir/CMakeFiles/ztpmv_thread_NUU.c.obj
[  9%] Building C object driver/level2/CMakeFiles/driver_level2.dir/CMakeFiles/ztpmv_thread_NLU.c.obj
```

- And finally we got **cblas.h** located in *my_build/generated/*

  and **libopenblas.a** located in *my_build/lib/*

- **This is not the only way to compile OpenBLAS , still README.md has more detailed.**

What is a 'libopenblas.a' file

- **libopenblas.a** we just compiled is a static lib.
- When we include **cblas.h** and link **libopenblas.a** , then we can make our code using this third lib run properly.