How to compile OpenCV 2.4 in Windows for Visual Studio

In this project (Realtime video mosaic) there is two libraries which by default it is not compiled. You must compile it yourself for your system.

Nonfree and **GPU** are the two libraries it is not in the OpenCV by default.

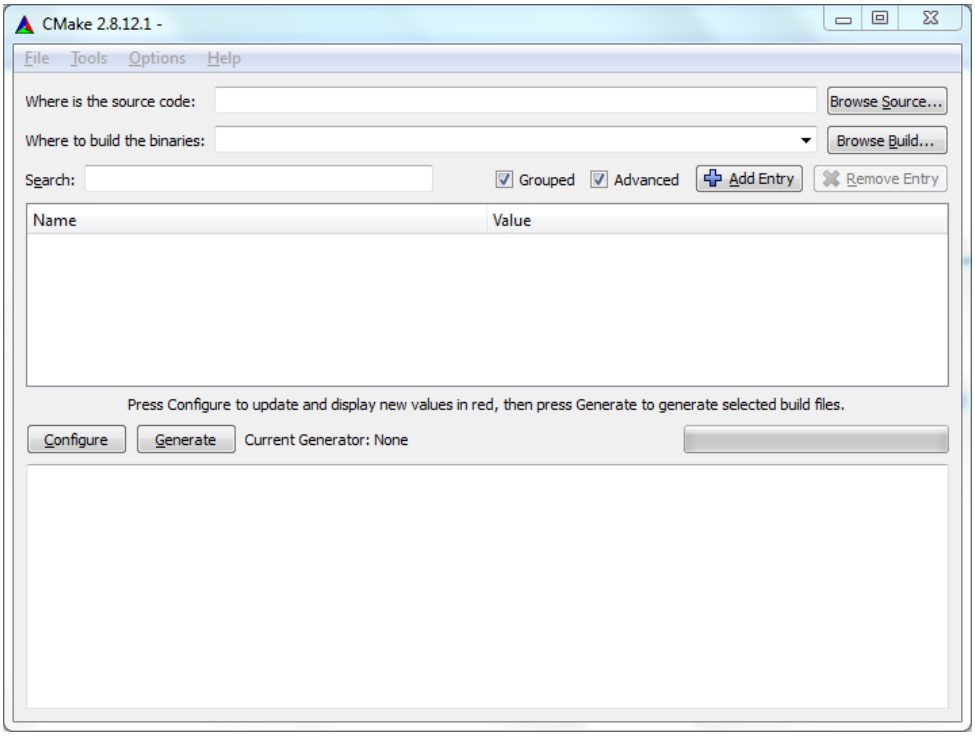
Before compile you need to install CUDA and download TBB.

Here is the instruction of how to compile it:

1. Download the source code of OpenCV 2.411 from

https://sourceforge.net/projects/opencvlibrary/files/opencv-win/2.4.11/opencv-2.4.11.exe/download

2. After extracting you need to compile the source. I used CMakeGUI version 2.8.12.1.

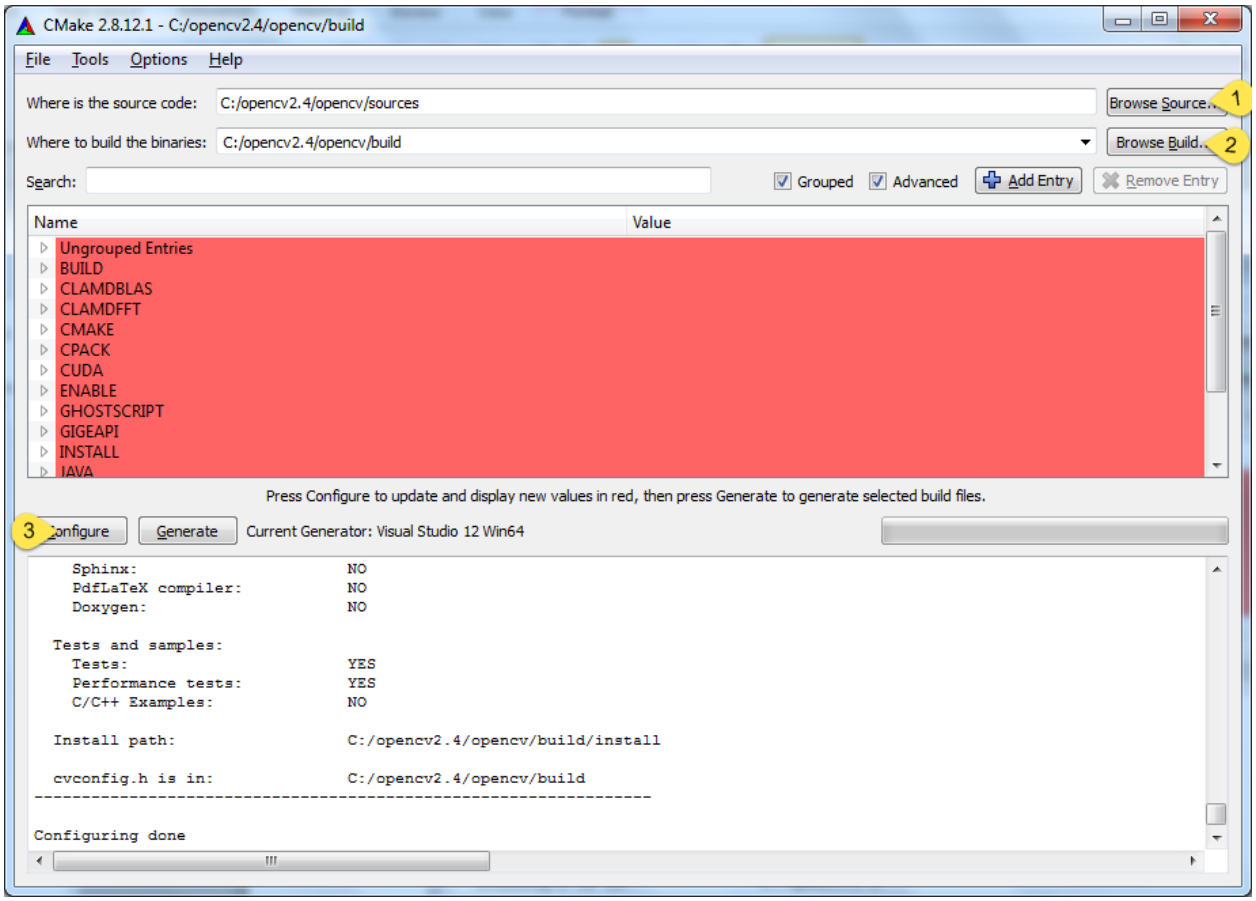


First use **Browse Source** and address the source code.

Second use **Browse Build** and address where you want to make the compiled libraries.

Third use **Configure** to set the compiler. Select **Visual Studio 12 Win64** and use native compiler.

After these steps you can see something similar to the image below.



Make sure in CUDA section libraries are found. If not you need to install CUDA or address them manually.

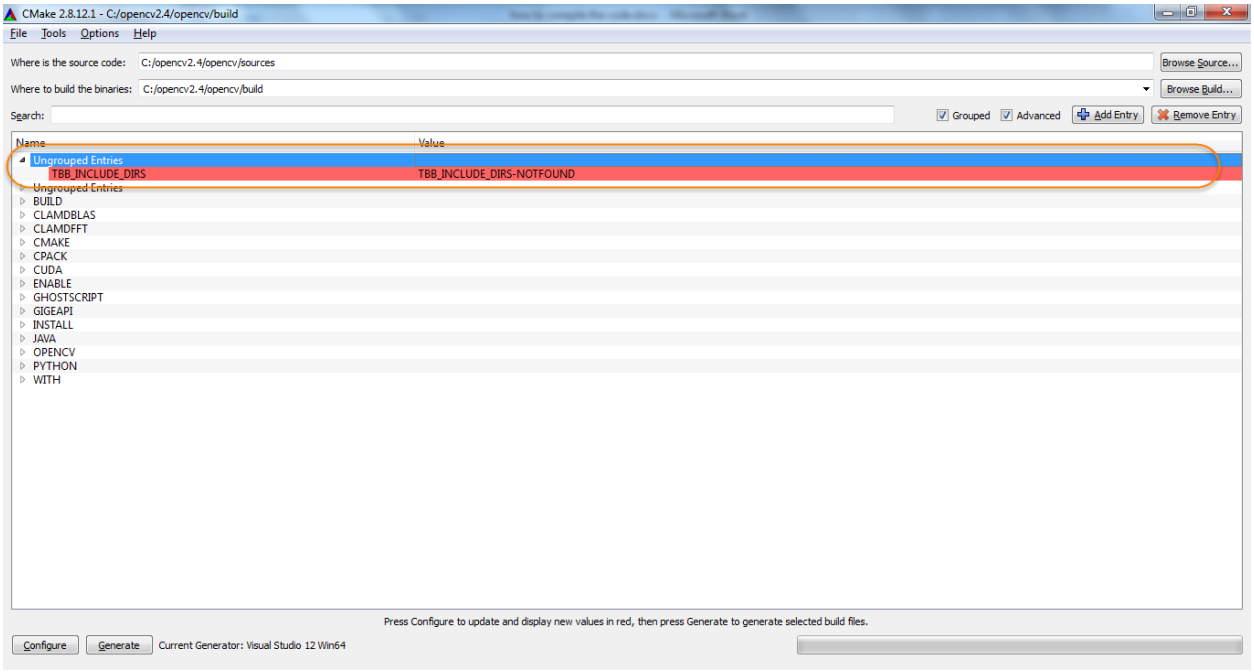
Make sure in BUILD section Build_OPENCV_NONFREE is checked.

Make sure in WITH section WITH_CUDA is checked.

In WITH section check WITH_TBB and click configure again. TBB is used for parallelism in CPU. There is no parallelism in code yet but it is better to have it for future use.

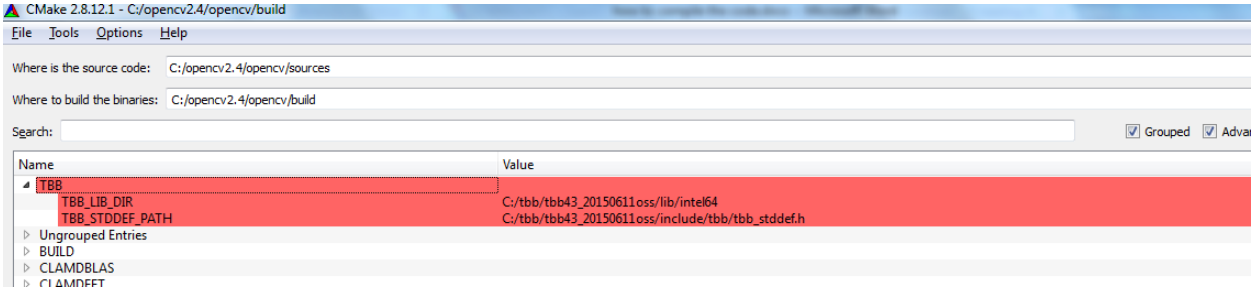
You can download it from here: https://www.threadingbuildingblocks.org/download

After clicking CONFIGURE the new section will appear. In this section you should address where is your TBB include directory.

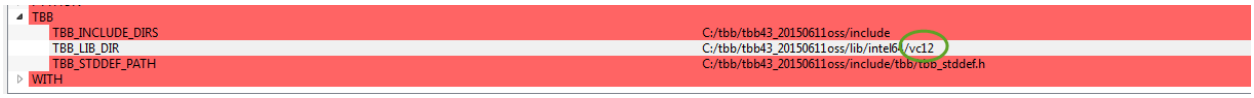


After addressing click CONFIGURE again.

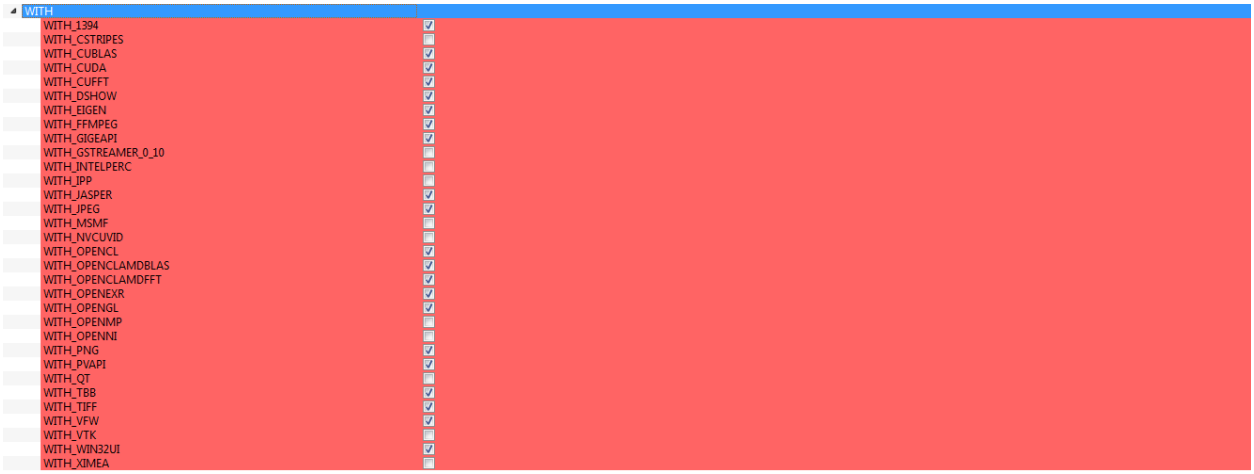
The new section named TBB will appear. It should be similar to the image below:



Note thad you must add /vc12 to the end of TBB dir. you must address where TBB_debug.lib and TBB.lib are available directly (not in his subfolders)



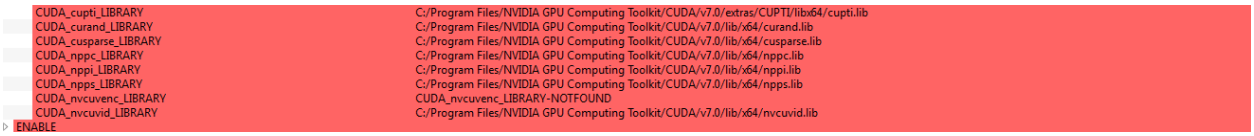
Here is the WITH section.



WITH	
WITH_1394	☑
WITH_CSTRIPES	☐
WITH_CUBLAS	☑
WITH_CUDA	☑
WITH_CUFFT	☑
WITH_DSHOW	☑
WITH_EIGEN	☑
WITH_FFMPEG	☑
WITH_GIGEAPI	☑
WITH_GSTREAMER_0_10	☐
WITH_INTELPERC	☐
WITH_IPP	☐
WITH_JASPER	☑
WITH_JPEG	☑
WITH_MSMF	☐
WITH_NVCUVID	☐
WITH_OPENCL	☑
WITH_OPENCLAMDBLAS	☑
WITH_OPENCLAMDFFT	☑
WITH_OPENEXR	☑
WITH_OPENGL	☑
WITH_OPENMP	☐
WITH_OPENNI	☐
WITH_PNG	☑
WITH_PVAPI	☑
WITH_QT	☐
WITH_TBB	☑
WITH_TIFF	☑
WITH_VFW	☑
WITH_VTK	☐
WITH_WIN32UI	☑
WITH_XIMEA	☐

Here is CUDA section:

CUDA	
CUDA_64_BIT_DEVICE_CODE	☑
CUDA_ARCH_BIN	2.0 2.1(2.0)
CUDA_ARCH_PTX	
CUDA_ATTACH_VS_BUILD_RULE_TO_CUDA_FILE	☑
CUDA_BUILD_CUBIN	☐
CUDA_BUILD_EMULATION	☐
CUDA_CUDART_LIBRARY	C:/Program Files/NVIDIA GPU Computing Toolkit/CUDA/v7.0/lib/x64/cudart.lib
CUDA_CUDA_LIBRARY	C:/Program Files/NVIDIA GPU Computing Toolkit/CUDA/v7.0/lib/x64/cuda.lib
CUDA_FAST_MATH	☑
CUDA_GENERATED_OUTPUT_DIR	
CUDA_GENERATION	Fermi
CUDA_HOST_COMPILATION_CPP	☑
CUDA_HOST_COMPILER	$(VCInstallDir)bin
CUDA_NVCC_EXECUTABLE	C:/Program Files/NVIDIA GPU Computing Toolkit/CUDA/v7.0/bin/nvcc.exe
CUDA_NVCC_FLAGS	
CUDA_NVCC_FLAGS_DEBUG	
CUDA_NVCC_FLAGS_MINSIZEREL	
CUDA_NVCC_FLAGS_RELEASE	
CUDA_NVCC_FLAGS_RELWITHDEBINFO	
CUDA_PROPAGATE_HOST_FLAGS	☑
CUDA_SDK_ROOT_DIR	CUDA_SDK_ROOT_DIR-NOTFOUND
CUDA_SEPARABLE_COMPILATION	☐
CUDA_TARGET_CPU_ARCH	
CUDA_TARGET_OS_VARIANT	
CUDA_TARGET_TRIPLET	
CUDA_TOOLKIT_INCLUDE	C:/Program Files/NVIDIA GPU Computing Toolkit/CUDA/v7.0/include
CUDA_TOOLKIT_ROOT_DIR	C:/Program Files/NVIDIA GPU Computing Toolkit/CUDA/v7.0
CUDA_VERBOSE_BUILD	☐
CUDA_VERSION	7.0
CUDA_cublas_LIBRARY	C:/Program Files/NVIDIA GPU Computing Toolkit/CUDA/v7.0/lib/x64/cublas.lib
CUDA_cufft_LIBRARY	C:/Program Files/NVIDIA GPU Computing Toolkit/CUDA/v7.0/lib/x64/cufft.lib



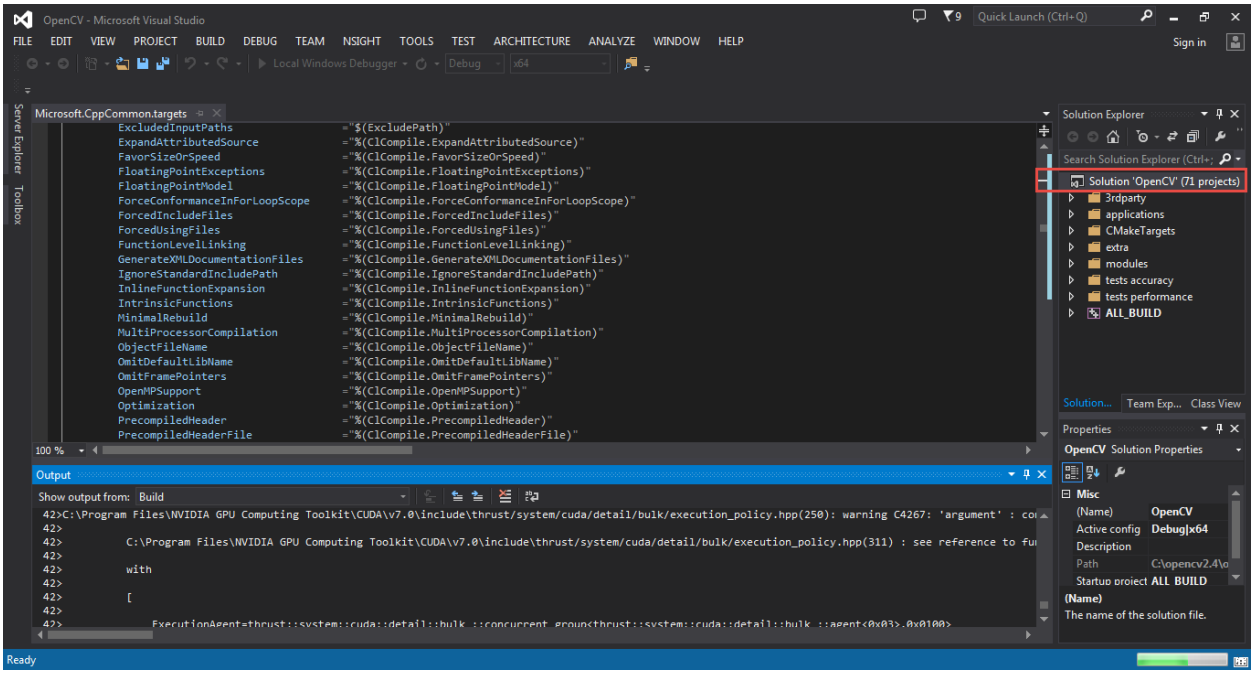
CUDA_cupti_LIBRARY	C:/Program Files/NVIDIA GPU Computing Toolkit/CUDA/v7.0/extras/CUPTI/libx64/cupti.lib
CUDA_curand_LIBRARY	C:/Program Files/NVIDIA GPU Computing Toolkit/CUDA/v7.0/lib/x64/curand.lib
CUDA_cusparse_LIBRARY	C:/Program Files/NVIDIA GPU Computing Toolkit/CUDA/v7.0/lib/x64/cusparse.lib
CUDA_nppc_LIBRARY	C:/Program Files/NVIDIA GPU Computing Toolkit/CUDA/v7.0/lib/x64/nppc.lib
CUDA_nppi_LIBRARY	C:/Program Files/NVIDIA GPU Computing Toolkit/CUDA/v7.0/lib/x64/nppi.lib
CUDA_npps_LIBRARY	C:/Program Files/NVIDIA GPU Computing Toolkit/CUDA/v7.0/lib/x64/npps.lib
CUDA_nvcuvenc_LIBRARY	CUDA_nvcuvenc_LIBRARY-NOTFOUND
CUDA_nvcuvid_LIBRARY	C:/Program Files/NVIDIA GPU Computing Toolkit/CUDA/v7.0/lib/x64/nvcuvid.lib
ENABLE	

You must select **Fermi** or **Kepler** for your graphic card in CUDA_GENERATION. **Don't use Auto**. You can search in the internet and find out if your graphic card generation is Fermi or Kepler. Kepler is better and Fermi is older than kepler. If your graphic card is new version it is probably Kepler version.

After doing all steps you can click on GENERATE. Then goto where you address build and open "OpenCV.sln" file. Now you need to add TBB_debug.lib and TBB.lib for further process.

Note: After any changes in CMAKE you must first select CONFIGURE button then GENERATE.

Right click on solution and select build.



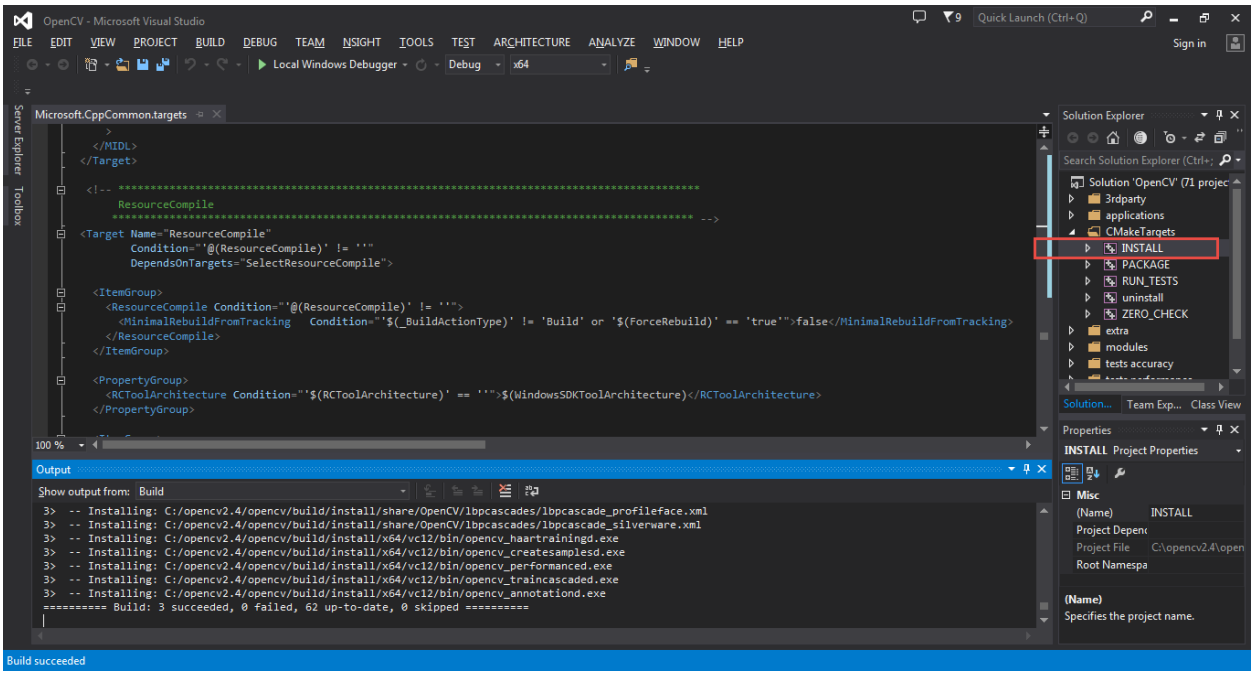
IMPORTANT TIPS: you can first goto module and compile opencv_core first. If this module compiled without error you can proceed otherwise try to configure the cmake correctly first. Without this module you can do nothing.

Note: Compilation may take 2-3 hour.

After compiling successfully compile it for release mode too (compile in debug mode and release mode both) and then your opencv libraries are ready. You can add it to your main project.

After finishing the whole compilation (in release and debug mode both) in order to collect libraries and other stuffs in a folder go to CMakeTargets-> INSTALL and right click and click on build. All you need is in

build/install folder.



Here is the useful link you can use it for more information:

http://docs.opencv.org/2.4/doc/tutorials/introduction/windows_install/windows_install.html