Hogeschool van Arnhem en Nijmegen

# Installation Desktop IDE and wxWidgets ToolKit with DSPESEL (DSBprt)

# for

# HAN Electrical/Embedded Systems Engineering

## Inhoudsopgave

# 1 Software IDE installation

There are several options for software development of desktop software. HAN ESE supports the possibility to perform the DSPESEL practicum on the following platforms:

- Apple Mac OS X
- Microsoft Windows 7 and higher
- Ubuntu Linux or FreeBSD

The DSPESEL practicum is largely performed on the basis of the wxWidgets toolkit. The choice for this toolkit is based on the following points:

- wxWidgets is a mature cross-platform GI toolkit that can also be used commercially. The design of the toolkit is similar to Qt.
- WxWidgets is fully open source.
- WxWidgets produces applications that are indistinguishable from the platform standard.
- Students must learn a variety of tools, and not only know other common platforms (such as Qt).

A comparison between wxWidgets and Qt can be found here .

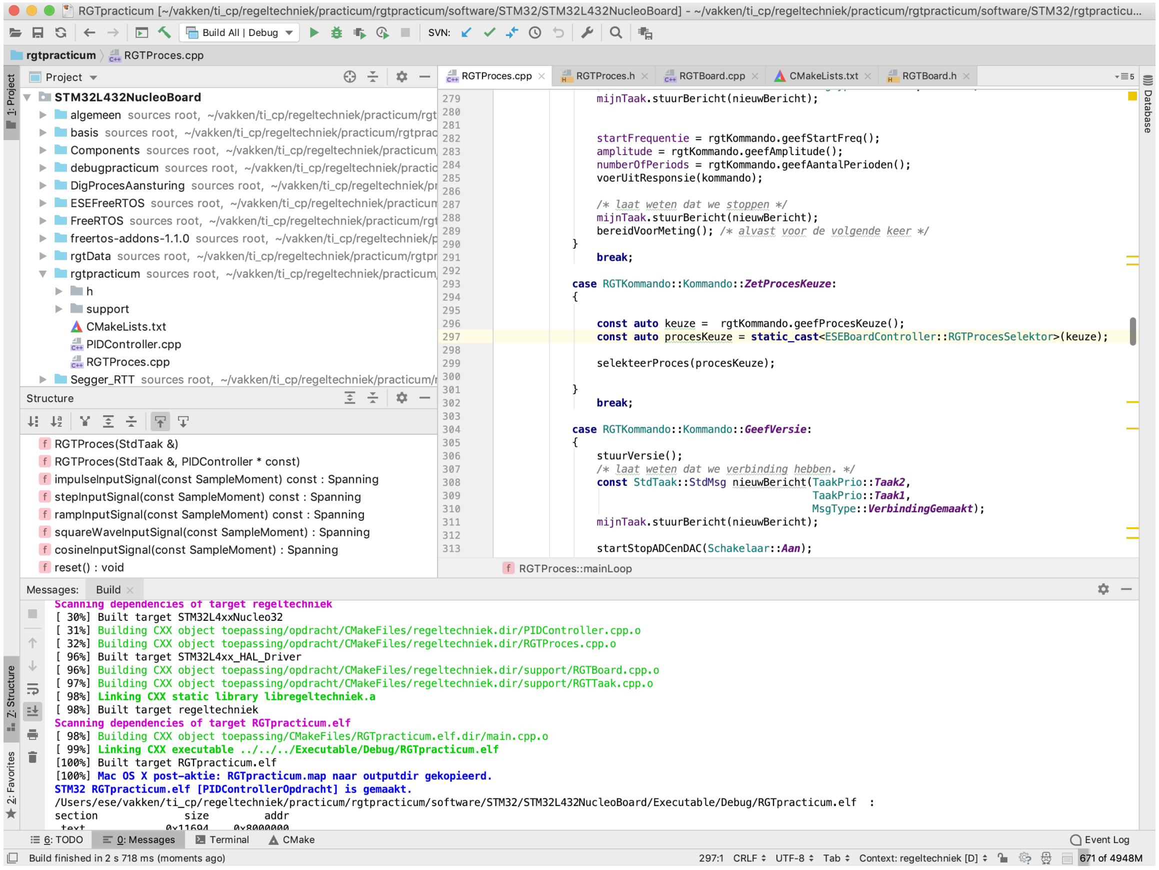
The following sections explain how wxWidgets is used in the practicum. JetBrains CLion is used on Unix platforms, MS Visual Studio is used on Microsoft Windows.

The DSPESEL practicum uses CMake for the project structure.

## 1.1 Jetbrains CLion  (Apple Mac OS X and Linux)

CLion is a superior development environment (IDE) for software projects based on the CMake project structure.

CLion looks as depicted below (Mac OS X version) :


RGTpracticum [~/vakken/ti_cp/regeltechniek/practicum/rgtpracticum/software/STM32/STM32L432NucleoBoard] - ~/vakken/ti_cp/regeltechniek/practicum/rgtpracticum/software/STM32/rgtpracticu...
Build All | Debug
SVN:
rgtpracticum
RGTProces.cpp
1: Project
Project
STM32L432NucleoBoard
algemeen sources root, ~/vakken/ti_cp/regeltechniek/practicum/rgt
basis sources root, ~/vakken/ti_cp/regeltechniek/practicum/rgtprac
Components sources root, ~/vakken/ti_cp/regeltechniek/practicum
debugpracticum sources root, ~/vakken/ti_cp/regeltechniek/practic
DigProcesAansturing sources root, ~/vakken/ti_cp/regeltechniek/pr
ESEFreeRTOS sources root, ~/vakken/ti_cp/regeltechniek/practicum
FreeRTOS sources root, ~/vakken/ti_cp/regeltechniek/practicum/rgt
freertos-addons-1.1.0 sources root, ~/vakken/ti_cp/regeltechniek/p
rgtData sources root, ~/vakken/ti_cp/regeltechniek/practicum/rgtpr
rgtpracticum sources root, ~/vakken/ti_cp/regeltechniek/practicum
h
support
CMakeLists.txt
PIDController.cpp
RGTProces.cpp
Segger_RTT sources root, ~/vakken/ti_cp/regeltechniek/practicum/r
Structure
RGTProces(StdTaak &)
RGTProces(StdTaak &, PIDController * const)
impulseInputSignal(const SampleMoment) const : Spanning
stepInputSignal(const SampleMoment) const : Spanning
rampInputSignal(const SampleMoment) const : Spanning
squareWaveInputSignal(const SampleMoment) : Spanning
cosineInputSignal(const SampleMoment) : Spanning
reset() : void
RGTProces.cpp
RGTProces.h
RGTBoard.cpp
CMakeLists.txt
RGTBoard.h
5
Database
RGTProces::mainLoop
Messages:
Build
7: Structure
2: Favorites
6: TODO
0: Messages
Terminal
CMake
Event Log
Build finished in 2 s 718 ms (moments ago)
297:1
CRLF
UTF-8
Tab
Context: regeltechniek [D]
671 of 4948M

```
            mijnTaak.stuurBericht(nieuwBericht);


            startFrequentie = rgtKommando.geefStartFreq();
            amplitude = rgtKommando.geefAmplitude();
            numberOfPeriods = rgtKommando.geefAantalPerioden();
            voerUitResponsie(kommando);

            /* laat weten dat we stoppen */
            mijnTaak.stuurBericht(nieuwBericht);
            bereidVoorMeting(); /* alvast voor de volgende keer */
        }
            break;

        case RGTKommando::Kommando::ZetProcesKeuze:
        {

            const auto keuze =  rgtKommando.geefProcesKeuze();
            const auto procesKeuze = static_cast<ESEBoardController::RGTProcesSelektor>(keuze);

            selekteerProces(procesKeuze);

        }
            break;

        case RGTKommando::Kommando::GeefVersie:
        {
            stuurVersie();
            /* laat weten dat we verbinding hebben. */
            const StdTaak::StdMsg nieuwBericht(TaakPrio::Taak2,
                                               TaakPrio::Taak1,
                                               MsgType::VerbindingGemaakt);
            mijnTaak.stuurBericht(nieuwBericht);

            startStopADCenDAC(Schakelaar::Aan);
```

```
Scanning dependencies of target regeltechniek
[ 30%] Built target STM32L4xxNucleo32
[ 31%] Building CXX object toepassing/opdracht/CMakeFiles/regeltechniek.dir/PIDController.cpp.o
[ 32%] Building CXX object toepassing/opdracht/CMakeFiles/regeltechniek.dir/RGTProces.cpp.o
[ 96%] Built target STM32L4xx_HAL_Driver
[ 96%] Building CXX object toepassing/opdracht/CMakeFiles/regeltechniek.dir/support/RGTBoard.cpp.o
[ 97%] Building CXX object toepassing/opdracht/CMakeFiles/regeltechniek.dir/support/RGTTaak.cpp.o
[ 98%] Linking CXX static library libregeltechniek.a
[ 98%] Built target regeltechniek
Scanning dependencies of target RGTpracticum.elf
[ 98%] Building CXX object toepassing/CMakeFiles/RGTpracticum.elf.dir/main.cpp.o
[ 99%] Linking CXX executable ../../../Executable/Debug/RGTpracticum.elf
[100%] Built target RGTpracticum.elf
[100%] Mac OS X post-aktie: RGTpracticum.map naar outputdir gekopieerd.
STM32 RGTpracticum.elf [PIDControllerOpdracht] is gemaakt.
/Users/ese/vakken/ti_cp/regeltechniek/practicum/rgtpracticum/software/STM32/STM32L432NucleoBoard/Executable/Debug/RGTpracticum.elf  :
section              size         addr
.text              0x11694     0x8000000
```

*Jetbrains CLion*

The benefits of CLion are:

- A superior IDE to work with modern features such as refactoring, very good navigation and full support for version management with SVN, Git and others.
- A much better support for modern C ++ variants (currently: C ++ 14).
- Flexible project design based on CMake.
- A superior and very fast debugger.

### 1.1.1 Installatie van CLion

Download Jetbrains Clion for the desired platform.

If errors occur, then something is probably not installed correctly in the previous steps. CLion comes with a 30-day trial period, after which a license is required.

Optional: Register with Jetbrains immediately to obtain a student license. When obtaining a license, make sure you use an official student.han.nl email address (not google / hotmail, etc.) for identification.

### 1.1.2 Installatie of wxWidgets

The installation can be performed must be performed in various steps.

Proceed as follows:

1. [Only for Apple Mac OS X]

   If not present, download and install Apple XCode via the App Store.

   Install through MacPorts (or HomeBrew) the wxWidgets toolkit and optionally the clang compiler and subversion :

   - Macport wxWidgets link
   - Macports LLVM Link
   - MacPorts SVN link

2. [FreeBSD/Linux]

   If not present, download and install the gcc and g ++ and or llvm c ++ compilers for Ubuntu Linux. FreeBSD has standard llvm clzang C / C ++ compilers installed.

3. In FreeBSD, install the wxgtk3 toolkit via Packages / Ports. In Ubuntu, do something similar with apt-get (not entirely clear, changes quite often). See the wx installlation guide for more information.

### 1.1.3 Usage of CLion

CLion can be used to build an (embedded) software project if a suitable CMake project file (CMakeLists.txt) is available, in which are defined:

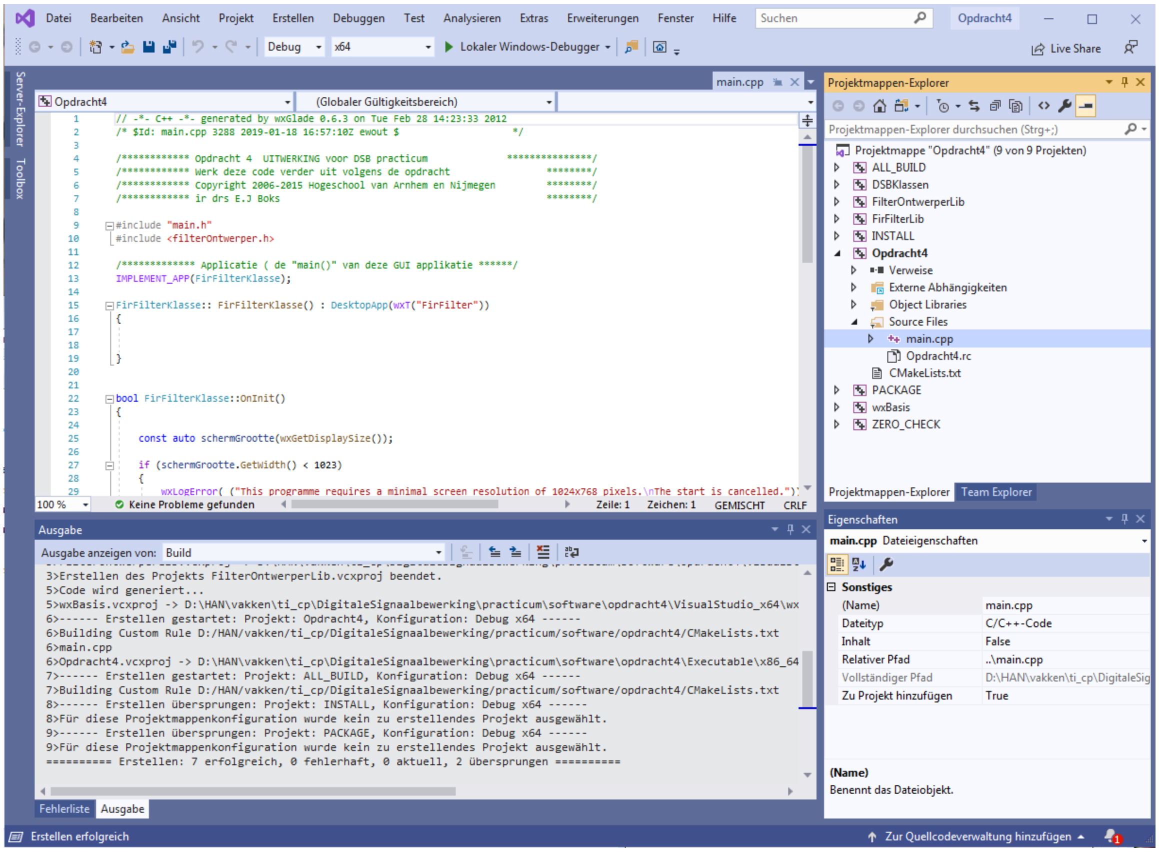
- The used compiler toolchain.
- The target processor.
- All options for the assembler, compiler and linker in the toolchain.
- All source files that must be compiled into the binary.
- All paths for header files.
- The desired form of the binary.
- Any custom steps that are required in the project.

Various tutorials and lots of information about CMake can be found on the internet. The practicals explain how each practical project must be configured for Cmake, Clion and Ozone.

## 1.2 Microsoft Visual Studio (Microsoft Windows)

Microsoft Visual Studio is used on the Microsoft Windows platform.

Visual Studio is a professional and very pleasant IDE that (of course) is very suitable for writing applications for Microsoft Windows. The reason that we opted for Visual Studio instead of CLion is the fact that the Microsoft compiler delivers better binaries for Windows than the GCC compiler that CLion works with on Windows. Microsoft keeps up with the times and integration with tools that are common in the Unix world is becoming more and more self-evident, making Visual Studio easier to handle alongside other tools.



```
// -*- C++ -*- generated by wxGlade 0.6.3 on Tue Feb 28 14:23:33 2012
/* $Id: main.cpp 3288 2019-01-18 16:57:10Z ewout $                */

/************* Opdracht 4  UITWERKING voor DSB practicum          ***************/
/************* Werk deze code verder uit volgens de opdracht              ********/
/************* Copyright 2006-2015 Hogeschool van Arnhem en Nijmegen      ********/
/************* ir drs E.J Boks                                            ********/

#include "main.h"
#include <filterOntwerper.h>

/************** Applicatie ( de "main()" van deze GUI applikatie ******/
IMPLEMENT_APP(FirFilterKlasse);

FirFilterKlasse:: FirFilterKlasse() : DesktopApp(wxT("FirFilter"))
{


}


bool FirFilterKlasse::OnInit()
{

    const auto schermGrootte(wxGetDisplaySize());

    if (schermGrootte.GetWidth() < 1023)
    {
        wxLogError( ("This programme requires a minimal screen resolution of 1024x768 pixels.\nThe start is cancelled.")
```

```
3>Erstellen des Projekts FilterOntwerperLib.vcxproj beendet.
5>Code wird generiert...
5>wxBasis.vcxproj -> D:\HAN\vakken\ti_cp\DigitaleSignaalbewerking\practicum\software\opdracht4\VisualStudio_x64\wx
6>------ Erstellen gestartet: Projekt: Opdracht4, Konfiguration: Debug x64 ------
6>Building Custom Rule D:/HAN/vakken/ti_cp/DigitaleSignaalbewerking/practicum/software/opdracht4/CMakeLists.txt
6>main.cpp
6>Opdracht4.vcxproj -> D:\HAN\vakken\ti_cp\DigitaleSignaalbewerking\practicum\software\opdracht4\Executable\x86_64
7>------ Erstellen gestartet: Projekt: ALL_BUILD, Konfiguration: Debug x64 ------
7>Building Custom Rule D:/HAN/vakken/ti_cp/DigitaleSignaalbewerking/practicum/software/opdracht4/CMakeLists.txt
8>------ Erstellen übersprungen: Projekt: INSTALL, Konfiguration: Debug x64 ------
8>Für diese Projektmappenkonfiguration wurde kein zu erstellendes Projekt ausgewählt.
9>------ Erstellen übersprungen: Projekt: PACKAGE, Konfiguration: Debug x64 ------
9>Für diese Projektmappenkonfiguration wurde kein zu erstellendes Projekt ausgewählt.
========== Erstellen: 7 erfolgreich, 0 fehlerhaft, 0 aktuell, 2 übersprungen ==========
```

main.cpp Dateieigenschaften

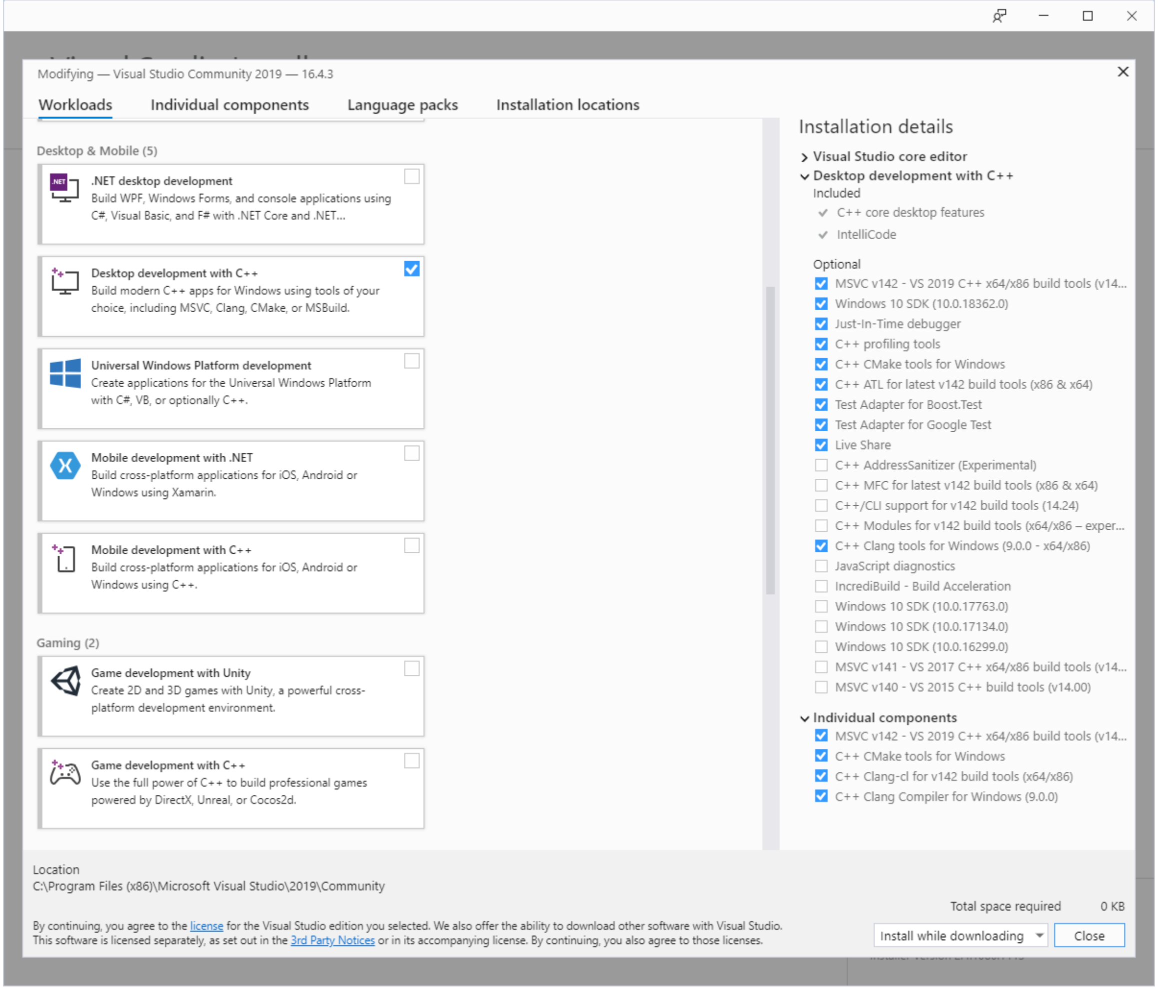
Sonstiges

| | |
|---|---|
| (Name) | main.cpp |
| Dateityp | C/C++-Code |
| Inhalt | False |
| Relativer Pfad | ..\main.cpp |
| Vollständiger Pfad | D:\HAN\vakken\ti_cp\DigitaleSig |
| Zu Projekt hinzufügen | True |

(Name)
Benennt das Dateiobjekt.

*Microsoft Visual Studio Community 2019*

An installer for Visual Studio is named VS Community. The installation follows after starting this installer by choosing the components (for ESE) used:



Modifying — Visual Studio Community 2019 — 16.4.3

Workloads | Individual components | Language packs | Installation locations

Desktop & Mobile (5)

□ .NET desktop development
Build WPF, Windows Forms, and console applications using C#, Visual Basic, and F# with .NET Core and .NET...

☑ Desktop development with C++
Build modern C++ apps for Windows using tools of your choice, including MSVC, Clang, CMake, or MSBuild.

□ Universal Windows Platform development
Create applications for the Universal Windows Platform with C#, VB, or optionally C++.

□ Mobile development with .NET
Build cross-platform applications for iOS, Android or Windows using Xamarin.

□ Mobile development with C++
Build cross-platform applications for iOS, Android or Windows using C++.

Gaming (2)

□ Game development with Unity
Create 2D and 3D games with Unity, a powerful cross-platform development environment.

□ Game development with C++
Use the full power of C++ to build professional games powered by DirectX, Unreal, or Cocos2d.

Installation details

> Visual Studio core editor

⌄ Desktop development with C++

Included

- ✓ C++ core desktop features
- ✓ IntelliCode

Optional

- ☑ MSVC v142 - VS 2019 C++ x64/x86 build tools (v14...
- ☑ Windows 10 SDK (10.0.18362.0)
- ☑ Just-In-Time debugger
- ☑ C++ profiling tools
- ☑ C++ CMake tools for Windows
- ☑ C++ ATL for latest v142 build tools (x86 & x64)
- ☑ Test Adapter for Boost.Test
- ☑ Test Adapter for Google Test
- ☑ Live Share
- □ C++ AddressSanitizer (Experimental)
- □ C++ MFC for latest v142 build tools (x86 & x64)
- □ C++/CLI support for v142 build tools (14.24)
- □ C++ Modules for v142 build tools (x64/x86 – exper...
- ☑ C++ Clang tools for Windows (9.0.0 - x64/x86)
- □ JavaScript diagnostics
- □ IncrediBuild - Build Acceleration
- □ Windows 10 SDK (10.0.17763.0)
- □ Windows 10 SDK (10.0.17134.0)
- □ Windows 10 SDK (10.0.16299.0)
- □ MSVC v141 - VS 2017 C++ x64/x86 build tools (v14...
- □ MSVC v140 - VS 2015 C++ build tools (v14.00)

⌄ Individual components

- ☑ MSVC v142 - VS 2019 C++ x64/x86 build tools (v14...
- ☑ C++ CMake tools for Windows
- ☑ C++ Clang-cl for v142 build tools (x64/x86)
- ☑ C++ Clang Compiler for Windows (9.0.0)

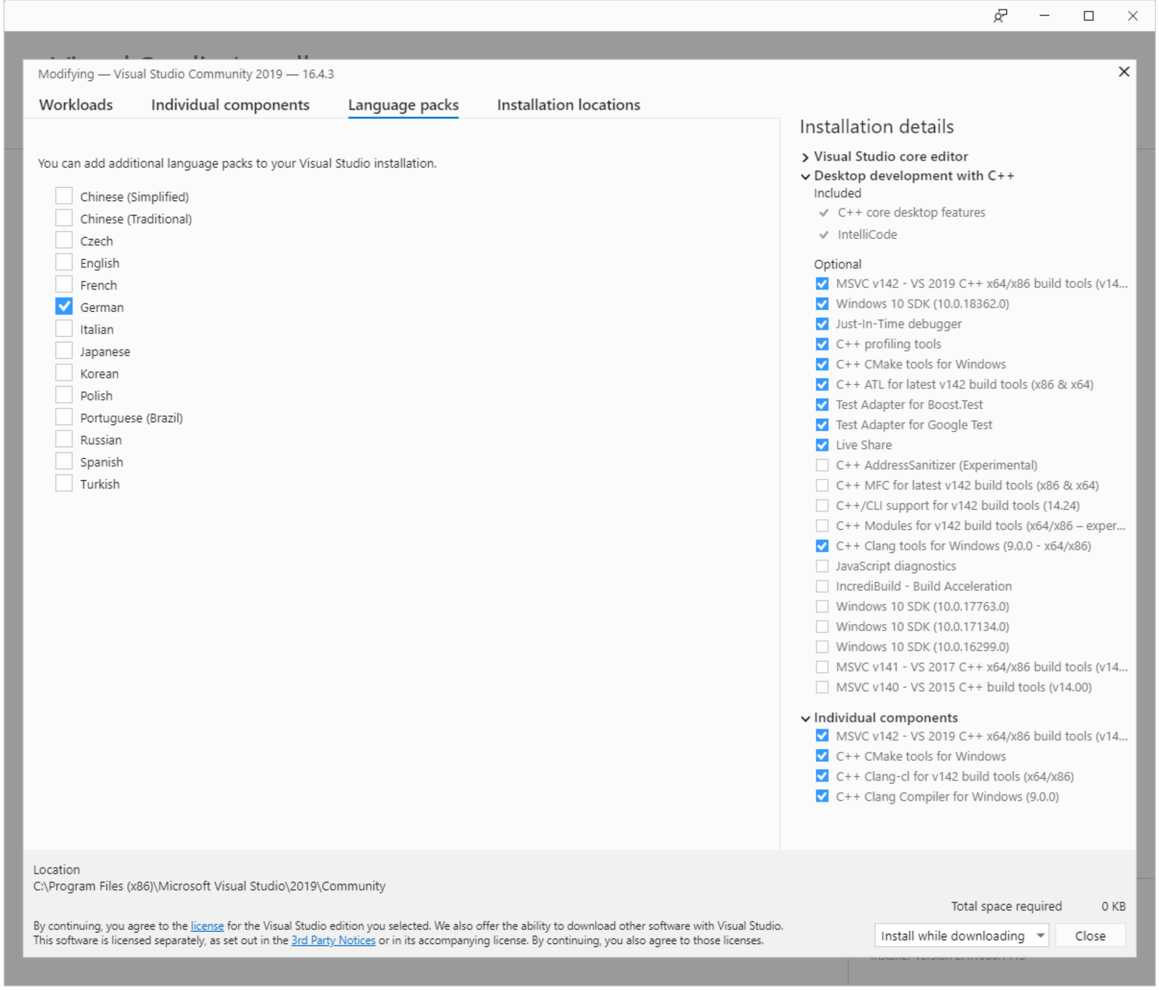
Location
C:\Program Files (x86)\Microsoft Visual Studio\2019\Community

Total space required 0 KB

By continuing, you agree to the license for the Visual Studio edition you selected. We also offer the ability to download other software with Visual Studio. This software is licensed separately, as set out in the 3rd Party Notices or in its accompanying license. By continuing, you also agree to those licenses.

Choose "Desktop development with C ++" and also check CMake and CLang tools. Other options are less relevant.

As can be seen in the image above from Visual Studio, a language other than English can be chosen. Those who like this can choose from a number of languages packs.


Modifying — Visual Studio Community 2019 — 16.4.3
Workloads
Individual components
Language packs
Installation locations
Install while downloading
Close

You can add additional language packs to your Visual Studio installation.

- ☐ Chinese (Simplified)
- ☐ Chinese (Traditional)
- ☐ Czech
- ☐ English
- ☐ French
- ☑ German
- ☐ Italian
- ☐ Japanese
- ☐ Korean
- ☐ Polish
- ☐ Portuguese (Brazil)
- ☐ Russian
- ☐ Spanish
- ☐ Turkish

Installation details

> Visual Studio core editor

∨ Desktop development with C++

Included

- ✓ C++ core desktop features
- ✓ IntelliCode

Optional

- ☑ MSVC v142 - VS 2019 C++ x64/x86 build tools (v14...
- ☑ Windows 10 SDK (10.0.18362.0)
- ☑ Just-In-Time debugger
- ☑ C++ profiling tools
- ☑ C++ CMake tools for Windows
- ☑ C++ ATL for latest v142 build tools (x86 & x64)
- ☑ Test Adapter for Boost.Test
- ☑ Test Adapter for Google Test
- ☑ Live Share
- ☐ C++ AddressSanitizer (Experimental)
- ☐ C++ MFC for latest v142 build tools (x86 & x64)
- ☐ C++/CLI support for v142 build tools (14.24)
- ☐ C++ Modules for v142 build tools (x64/x86 – exper...
- ☑ C++ Clang tools for Windows (9.0.0 - x64/x86)
- ☐ JavaScript diagnostics
- ☐ IncrediBuild - Build Acceleration
- ☐ Windows 10 SDK (10.0.17763.0)
- ☐ Windows 10 SDK (10.0.17134.0)
- ☐ Windows 10 SDK (10.0.16299.0)
- ☐ MSVC v141 - VS 2017 C++ x64/x86 build tools (v14...
- ☐ MSVC v140 - VS 2015 C++ build tools (v14.00)

∨ Individual components

- ☑ MSVC v142 - VS 2019 C++ x64/x86 build tools (v14...
- ☑ C++ CMake tools for Windows
- ☑ C++ Clang-cl for v142 build tools (x64/x86)
- ☑ C++ Clang Compiler for Windows (9.0.0)

Location
C:\Program Files (x86)\Microsoft Visual Studio\2019\Community

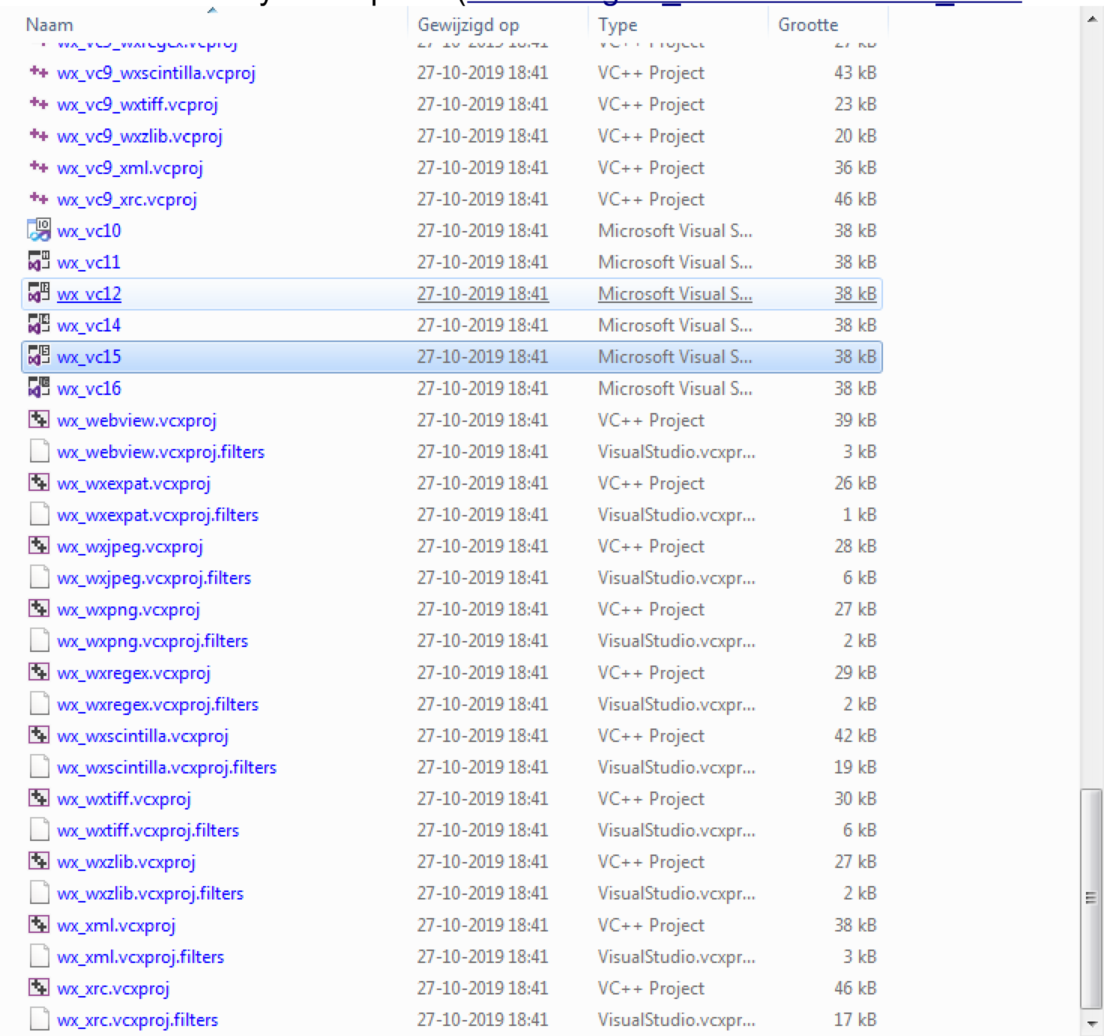
By continuing, you agree to the license for the Visual Studio edition you selected. We also offer the ability to download other software with Visual Studio. This software is licensed separately, as set out in the 3rd Party Notices or in its accompanying license. By continuing, you also agree to those licenses.

Total space required 0 KB

After installing Visual Studio, it is advisable to link it to a Microsoft account (for example, OneDrive).

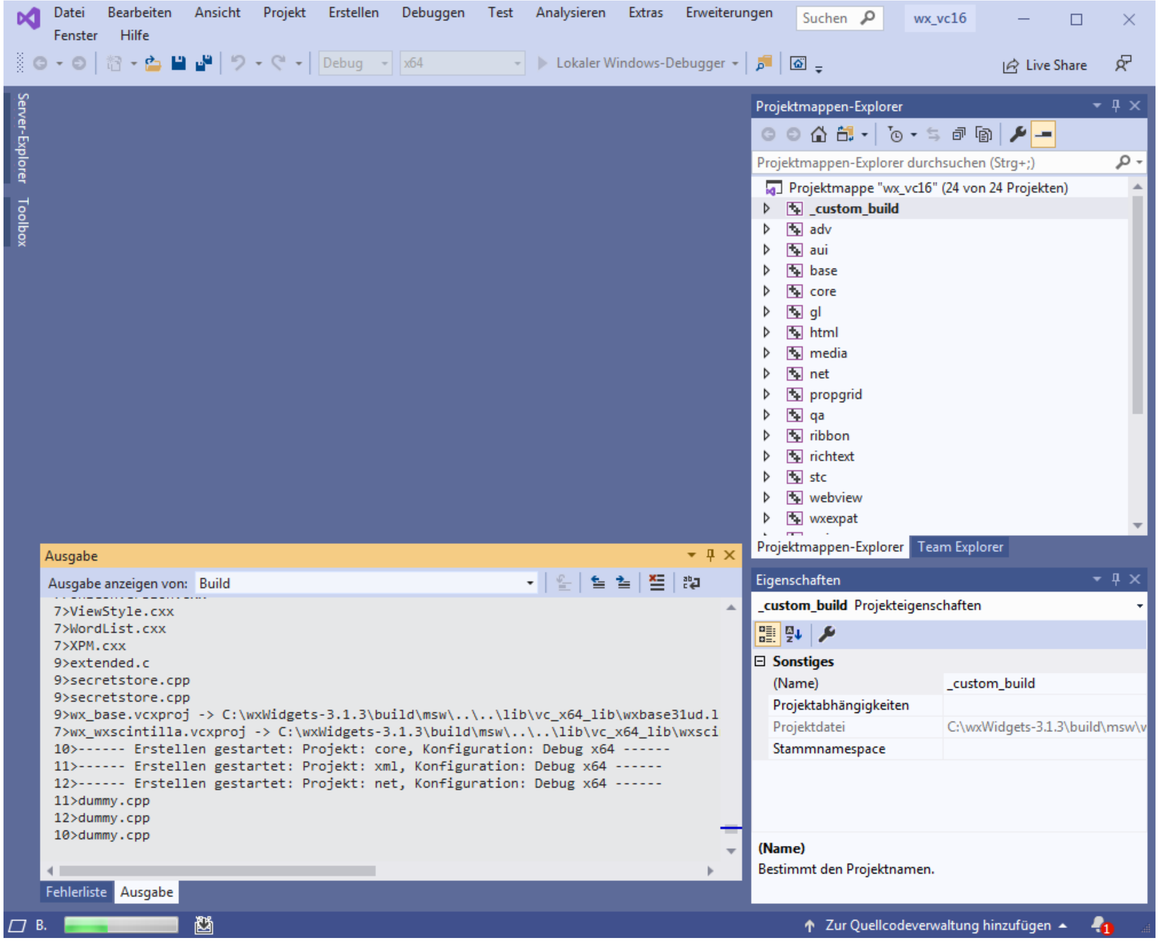
### 1.2.1 Installation of wxWidgets on MS Windows.

1. Download the wxWidgets win32 installer from the wxWidgets website to the standard location (C:/wxWidgets_x.xx).
2. Start Visual Studio en navigate to the directory where a vc_15/vc_16 solution is at your disposal (C:/wxWidgets_x.xx/build/msw/wx_vc15 of 16):



| Naam | Gewijzigd op | Type | Grootte |
|---|---|---|---|
| wx_vc9_wxregex.vcproj | 27-10-2019 18:41 | VC++ Project | 27 kB |
| wx_vc9_wxscintilla.vcproj | 27-10-2019 18:41 | VC++ Project | 43 kB |
| wx_vc9_wxtiff.vcproj | 27-10-2019 18:41 | VC++ Project | 23 kB |
| wx_vc9_wxzlib.vcproj | 27-10-2019 18:41 | VC++ Project | 20 kB |
| wx_vc9_xml.vcproj | 27-10-2019 18:41 | VC++ Project | 36 kB |
| wx_vc9_xrc.vcproj | 27-10-2019 18:41 | VC++ Project | 46 kB |
| wx_vc10 | 27-10-2019 18:41 | Microsoft Visual S... | 38 kB |
| wx_vc11 | 27-10-2019 18:41 | Microsoft Visual S... | 38 kB |
| wx_vc12 | 27-10-2019 18:41 | Microsoft Visual S... | 38 kB |
| wx_vc14 | 27-10-2019 18:41 | Microsoft Visual S... | 38 kB |
| wx_vc15 | 27-10-2019 18:41 | Microsoft Visual S... | 38 kB |
| wx_vc16 | 27-10-2019 18:41 | Microsoft Visual S... | 38 kB |
| wx_webview.vcxproj | 27-10-2019 18:41 | VC++ Project | 39 kB |
| wx_webview.vcxproj.filters | 27-10-2019 18:41 | VisualStudio.vcxpr... | 3 kB |
| wx_wxexpat.vcxproj | 27-10-2019 18:41 | VC++ Project | 26 kB |
| wx_wxexpat.vcxproj.filters | 27-10-2019 18:41 | VisualStudio.vcxpr... | 1 kB |
| wx_wxjpeg.vcxproj | 27-10-2019 18:41 | VC++ Project | 28 kB |
| wx_wxjpeg.vcxproj.filters | 27-10-2019 18:41 | VisualStudio.vcxpr... | 6 kB |
| wx_wxpng.vcxproj | 27-10-2019 18:41 | VC++ Project | 27 kB |
| wx_wxpng.vcxproj.filters | 27-10-2019 18:41 | VisualStudio.vcxpr... | 2 kB |
| wx_wxregex.vcxproj | 27-10-2019 18:41 | VC++ Project | 29 kB |
| wx_wxregex.vcxproj.filters | 27-10-2019 18:41 | VisualStudio.vcxpr... | 2 kB |
| wx_wxscintilla.vcxproj | 27-10-2019 18:41 | VC++ Project | 42 kB |
| wx_wxscintilla.vcxproj.filters | 27-10-2019 18:41 | VisualStudio.vcxpr... | 19 kB |
| wx_wxtiff.vcxproj | 27-10-2019 18:41 | VC++ Project | 30 kB |
| wx_wxtiff.vcxproj.filters | 27-10-2019 18:41 | VisualStudio.vcxpr... | 6 kB |
| wx_wxzlib.vcxproj | 27-10-2019 18:41 | VC++ Project | 27 kB |
| wx_wxzlib.vcxproj.filters | 27-10-2019 18:41 | VisualStudio.vcxpr... | 2 kB |
| wx_xml.vcxproj | 27-10-2019 18:41 | VC++ Project | 38 kB |
| wx_xml.vcxproj.filters | 27-10-2019 18:41 | VisualStudio.vcxpr... | 3 kB |
| wx_xrc.vcxproj | 27-10-2019 18:41 | VC++ Project | 46 kB |
| wx_xrc.vcxproj.filters | 27-10-2019 18:41 | VisualStudio.vcxpr... | 17 kB |

3. Build the solution in 64 bit mode (not in 32 bit mode!) In static debug and optional static release mode. DLL mode is not required :



Projektmappe "wx_vc16" (24 von 24 Projekten)
- _custom_build
- adv
- aui
- base
- core
- gl
- html
- media
- net
- propgrid
- qa
- ribbon
- richtext
- stc
- webview
- wxexpat

```
7>ViewStyle.cxx
7>WordList.cxx
7>XPM.cxx
9>extended.c
9>secretstore.cpp
9>secretstore.cpp
9>wx_base.vcxproj -> C:\wxWidgets-3.1.3\build\msw\..\..\lib\vc_x64_lib\wxbase31ud.l
7>wx_wxscintilla.vcxproj -> C:\wxWidgets-3.1.3\build\msw\..\..\lib\vc_x64_lib\wxsci
10>------ Erstellen gestartet: Projekt: core, Konfiguration: Debug x64 ------
11>------ Erstellen gestartet: Projekt: xml, Konfiguration: Debug x64 ------
12>------ Erstellen gestartet: Projekt: net, Konfiguration: Debug x64 ------
11>dummy.cpp
12>dummy.cpp
10>dummy.cpp
```

_custom_build Projekteigenschaften

Sonstiges

| | |
|---|---|
| (Name) | _custom_build |
| Projektabhängigkeiten | |
| Projektdatei | C:\wxWidgets-3.1.3\build\msw\v |
| Stammnamespace | |

**(Name)**
Bestimmt den Projektnamen.

After the construction is finished, everything is ready - leave the construction folder with all obj files, this is the wxWidgets toolkit for use in the practicum.

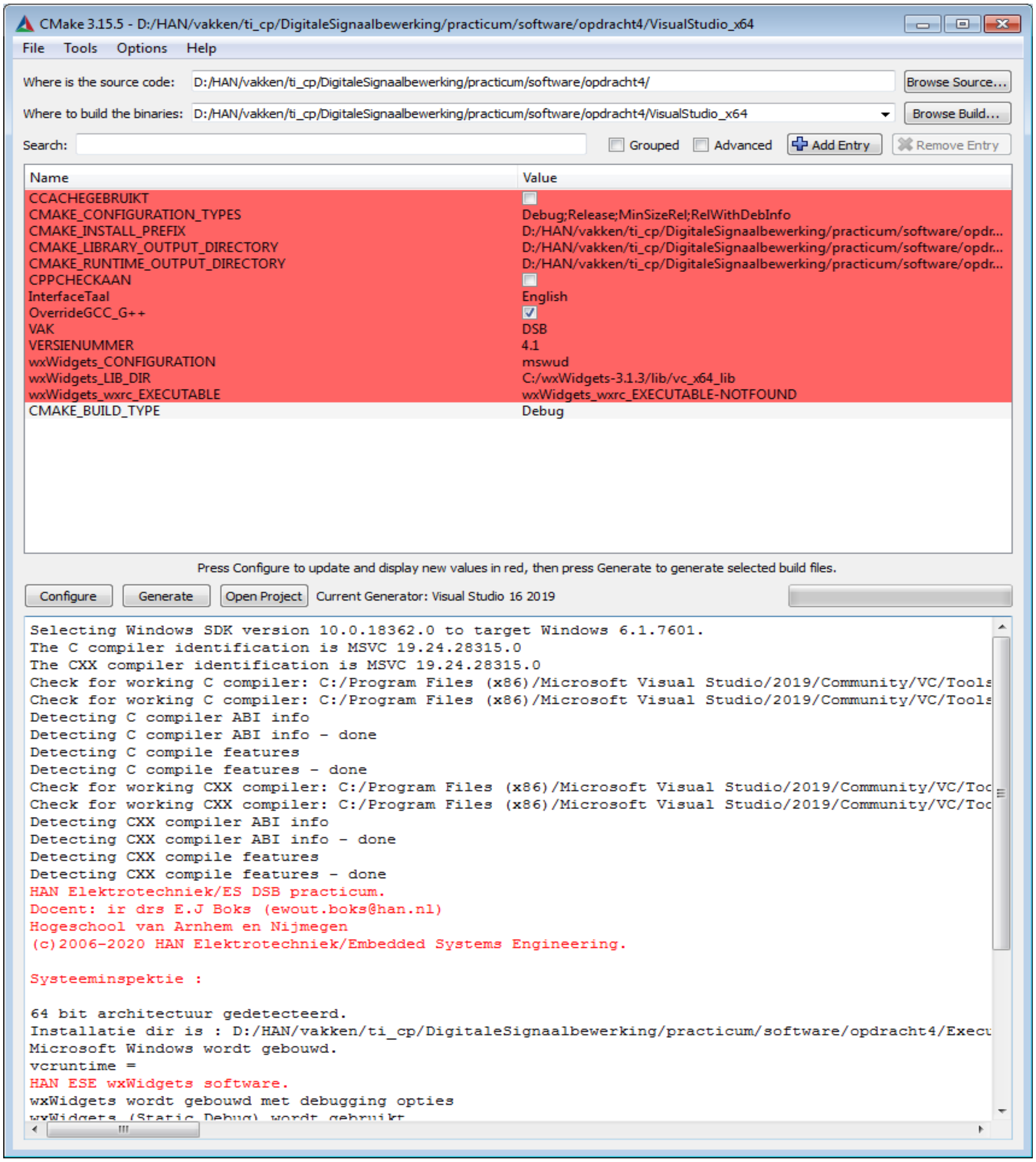
### 1.2.2 Installation of CMake

It is advisable to install CMake  in addition to Visual Studio as well, to independently generate Visual Studio Solution projects from CMake project files.

### 1.2.3 Usage of CMake with Visual Studio

It is possible to use the practical projects in Visual Studio in two ways:

- The traditional way. A VS Solution is constructed with the CMake generator, which is then loaded into Visual Studio:



Where is the source code: D:/HAN/vakken/ti_cp/DigitaleSignaalbewerking/practicum/software/opdracht4/

Where to build the binaries: D:/HAN/vakken/ti_cp/DigitaleSignaalbewerking/practicum/software/opdracht4/VisualStudio_x64

Search:

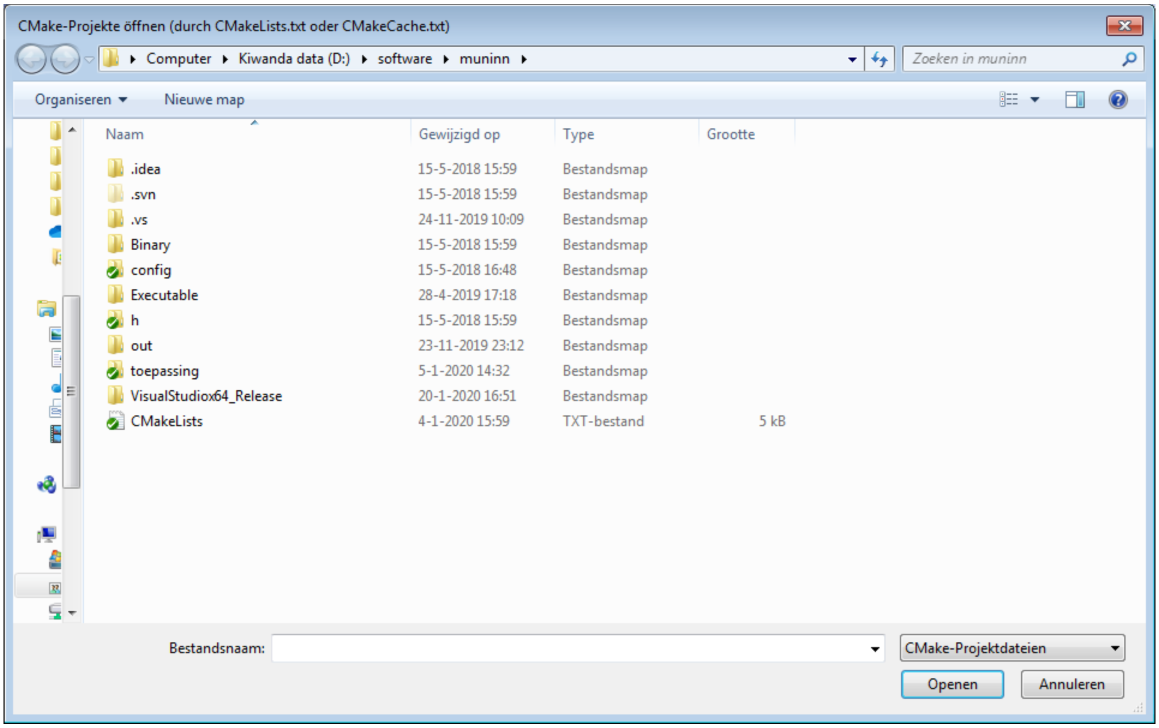
| Name | Value |
|---|---|
| CCACHEGEBRUIKT | □ |
| CMAKE_CONFIGURATION_TYPES | Debug;Release;MinSizeRel;RelWithDebInfo |
| CMAKE_INSTALL_PREFIX | D:/HAN/vakken/ti_cp/DigitaleSignaalbewerking/practicum/software/opdr... |
| CMAKE_LIBRARY_OUTPUT_DIRECTORY | D:/HAN/vakken/ti_cp/DigitaleSignaalbewerking/practicum/software/opdr... |
| CMAKE_RUNTIME_OUTPUT_DIRECTORY | D:/HAN/vakken/ti_cp/DigitaleSignaalbewerking/practicum/software/opdr... |
| CPPCHECKAAN | □ |
| InterfaceTaal | English |
| OverrideGCC_G++ | ☑ |
| VAK | DSB |
| VERSIENUMMER | 4.1 |
| wxWidgets_CONFIGURATION | mswud |
| wxWidgets_LIB_DIR | C:/wxWidgets-3.1.3/lib/vc_x64_lib |
| wxWidgets_wxrc_EXECUTABLE | wxWidgets_wxrc_EXECUTABLE-NOTFOUND |
| CMAKE_BUILD_TYPE | Debug |

Press Configure to update and display new values in red, then press Generate to generate selected build files.

Current Generator: Visual Studio 16 2019

```
Selecting Windows SDK version 10.0.18362.0 to target Windows 6.1.7601.
The C compiler identification is MSVC 19.24.28315.0
The CXX compiler identification is MSVC 19.24.28315.0
Check for working C compiler: C:/Program Files (x86)/Microsoft Visual Studio/2019/Community/VC/Tools
Check for working C compiler: C:/Program Files (x86)/Microsoft Visual Studio/2019/Community/VC/Tools
Detecting C compiler ABI info
Detecting C compiler ABI info - done
Detecting C compile features
Detecting C compile features - done
Check for working CXX compiler: C:/Program Files (x86)/Microsoft Visual Studio/2019/Community/VC/Toc
Check for working CXX compiler: C:/Program Files (x86)/Microsoft Visual Studio/2019/Community/VC/Toc
Detecting CXX compiler ABI info
Detecting CXX compiler ABI info - done
Detecting CXX compile features
Detecting CXX compile features - done
HAN Elektrotechniek/ES DSB practicum.
Docent: ir drs E.J Boks (ewout.boks@han.nl)
Hogeschool van Arnhem en Nijmegen
(c)2006-2020 HAN Elektrotechniek/Embedded Systems Engineering.

Systeeminspektie :

64 bit architectuur gedetecteerd.
Installatie dir is : D:/HAN/vakken/ti_cp/DigitaleSignaalbewerking/practicum/software/opdracht4/Execu
Microsoft Windows wordt gebouwd.
vcruntime =
HAN ESE wxWidgets software.
wxWidgets wordt gebouwd met debugging opties
wxWidgets (Static Debug) wordt gebruikt
```

- The "new" way. Microsoft has since VS2017 introduced a method with which CMake projects can be loaded directly into Visual Studio. At File / Open go to the opening option “CMake”, where after the file dialog appears with which a CMakeLists.txt file can be chosen:



CMake-Projekte öffnen (durch CMakeLists.txt oder CMakeCache.txt)

Computer ▸ Kiwanda data (D:) ▸ software ▸ muninn ▸

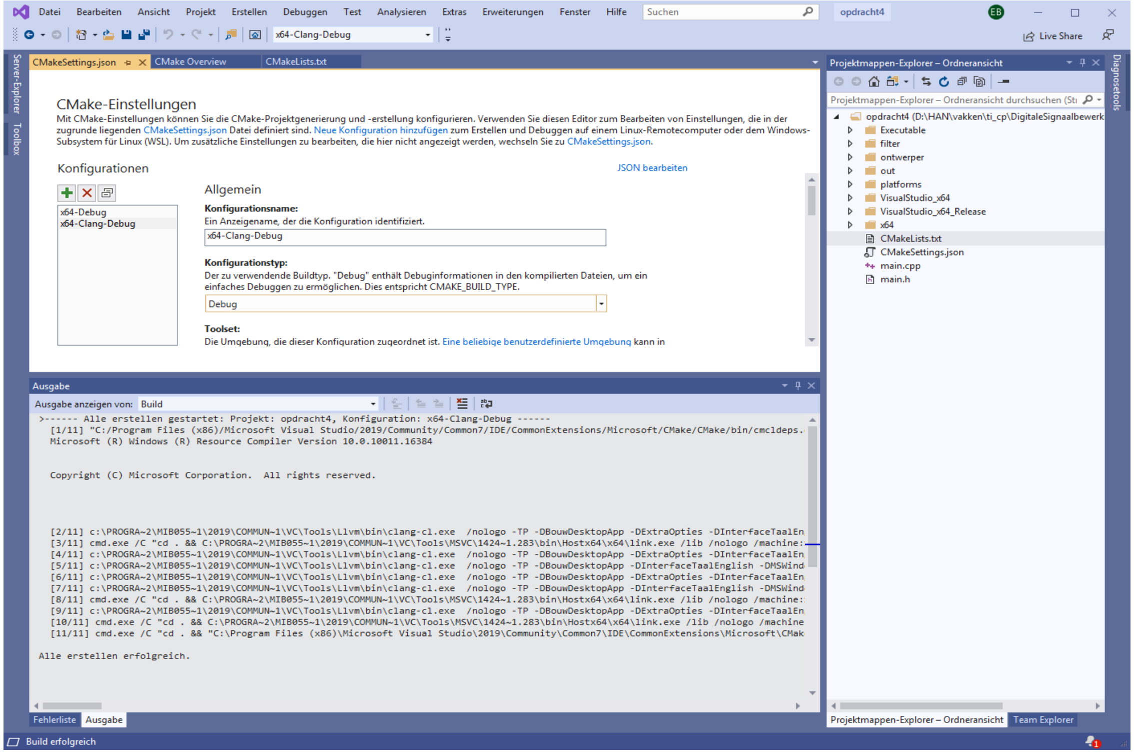
| Naam | Gewijzigd op | Type | Grootte |
|---|---|---|---|
| .idea | 15-5-2018 15:59 | Bestandsmap | |
| .svn | 15-5-2018 15:59 | Bestandsmap | |
| .vs | 24-11-2019 10:09 | Bestandsmap | |
| Binary | 15-5-2018 15:59 | Bestandsmap | |
| config | 15-5-2018 16:48 | Bestandsmap | |
| Executable | 28-4-2019 17:18 | Bestandsmap | |
| h | 15-5-2018 15:59 | Bestandsmap | |
| out | 23-11-2019 23:12 | Bestandsmap | |
| toepassing | 5-1-2020 14:32 | Bestandsmap | |
| VisualStudiox64_Release | 20-1-2020 16:51 | Bestandsmap | |
| CMakeLists | 4-1-2020 15:59 | TXT-bestand | 5 kB |

Bestandsnaam:

CMake-Projektdateien

Visual Studio will load here after the project and set to the standard construction method.

In addition to this standard method, other building configurations can be chosen. A configuration based on the LLVM clang compiler suite (the standard compiler for Apple Mac OS X and FreeBSD) is now also possible; the miracles are not out of the world yet.....:



CMake-Einstellungen

Mit CMake-Einstellungen können Sie die CMake-Projektgenerierung und -erstellung konfigurieren. Verwenden Sie diesen Editor zum Bearbeiten von Einstellungen, die in der zugrunde liegenden CMakeSettings.json Datei definiert sind. Neue Konfiguration hinzufügen zum Erstellen und Debuggen auf einem Linux-Remotecomputer oder dem Windows-Subsystem für Linux (WSL). Um zusätzliche Einstellungen zu bearbeiten, die hier nicht angezeigt werden, wechseln Sie zu CMakeSettings.json.

Konfigurationen

JSON bearbeiten

x64-Debug
x64-Clang-Debug

Allgemein

**Konfigurationsname:**
Ein Anzeigename, der die Konfiguration identifiziert.
x64-Clang-Debug

**Konfigurationstyp:**
Der zu verwendende Buildtyp. "Debug" enthält Debuginformationen in den kompilierten Dateien, um ein einfaches Debuggen zu ermöglichen. Dies entspricht CMAKE_BUILD_TYPE.
Debug

**Toolset:**
Die Umgebung, die dieser Konfiguration zugeordnet ist. Eine beliebige benutzerdefinierte Umgebung kann in

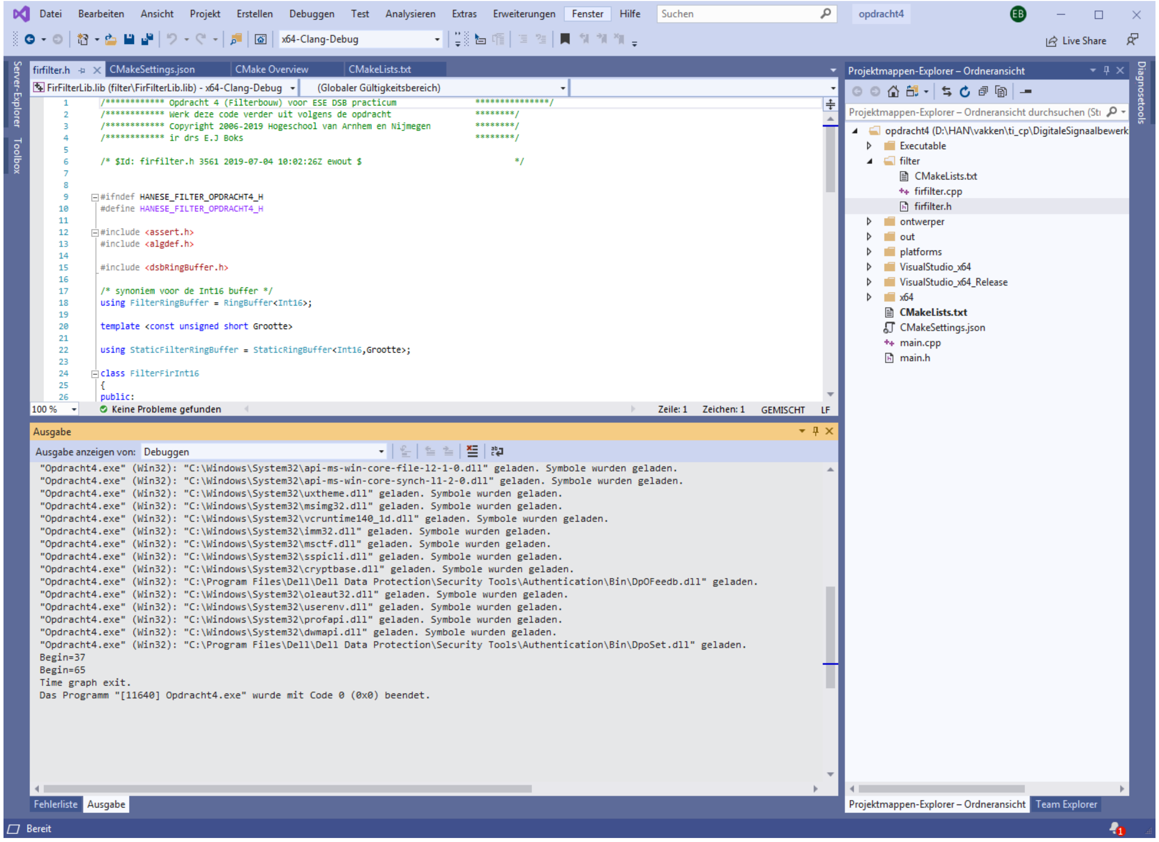
Ausgabe

Ausgabe anzeigen von: Build

```
>------ Alle erstellen gestartet: Projekt: opdracht4, Konfiguration: x64-Clang-Debug ------
  [1/11] "C:/Program Files (x86)/Microsoft Visual Studio/2019/Community/Common7/IDE/CommonExtensions/Microsoft/CMake/CMake/bin/cmcldeps.
  Microsoft (R) Windows (R) Resource Compiler Version 10.0.10011.16384


  Copyright (C) Microsoft Corporation.  All rights reserved.




  [2/11] c:\PROGRA~2\MIB055~1\2019\COMMUN~1\VC\Tools\Llvm\bin\clang-cl.exe  /nologo -TP -DBouwDesktopApp -DExtraOpties -DInterfaceTaalEn
  [3/11] cmd.exe /C "cd . && C:\PROGRA~2\MIB055~1\2019\COMMUN~1\VC\Tools\MSVC\1424~1.283\bin\Hostx64\x64\link.exe /lib /nologo /machine:
  [4/11] c:\PROGRA~2\MIB055~1\2019\COMMUN~1\VC\Tools\Llvm\bin\clang-cl.exe  /nologo -TP -DBouwDesktopApp -DExtraOpties -DInterfaceTaalEn
  [5/11] c:\PROGRA~2\MIB055~1\2019\COMMUN~1\VC\Tools\Llvm\bin\clang-cl.exe  /nologo -TP -DBouwDesktopApp -DInterfaceTaalEnglish -DMSWind
  [6/11] c:\PROGRA~2\MIB055~1\2019\COMMUN~1\VC\Tools\Llvm\bin\clang-cl.exe  /nologo -TP -DBouwDesktopApp -DExtraOpties -DInterfaceTaalEn
  [7/11] c:\PROGRA~2\MIB055~1\2019\COMMUN~1\VC\Tools\Llvm\bin\clang-cl.exe  /nologo -TP -DBouwDesktopApp -DInterfaceTaalEnglish -DMSWind
  [8/11] cmd.exe /C "cd . && C:\PROGRA~2\MIB055~1\2019\COMMUN~1\VC\Tools\MSVC\1424~1.283\bin\Hostx64\x64\link.exe /lib /nologo /machine:
  [9/11] c:\PROGRA~2\MIB055~1\2019\COMMUN~1\VC\Tools\Llvm\bin\clang-cl.exe  /nologo -TP -DBouwDesktopApp -DExtraOpties -DInterfaceTaalEn
  [10/11] cmd.exe /C "cd . && C:\PROGRA~2\MIB055~1\2019\COMMUN~1\VC\Tools\MSVC\1424~1.283\bin\Hostx64\x64\link.exe /lib /nologo /machine
  [11/11] cmd.exe /C "cd . && "C:\Program Files (x86)\Microsoft Visual Studio\2019\Community\Common7\IDE\CommonExtensions\Microsoft\CMak

Alle erstellen erfolgreich.
```

Build erfolgreich

After this, the project can be built and executed based on clang!



FirFilterLib.lib (filter\FirFilterLib.lib) - x64-Clang-Debug (Globaler Gültigkeitsbereich)

```
/************ Opdracht 4 (Filterbouw) voor ESE DSB practicum          ***************/
/************ Werk deze code verder uit volgens de opdracht            ********/
/************ Copyright 2006-2019 Hogeschool van Arnhem en Nijmegen    ********/
/************ ir drs E.J Boks                                          ********/

/* $Id: firfilter.h 3561 2019-07-04 10:02:26Z ewout $                           */


#ifndef HANESE_FILTER_OPDRACHT4_H
#define HANESE_FILTER_OPDRACHT4_H

#include <assert.h>
#include <algdef.h>

#include <dsbRingBuffer.h>

/* synoniem voor de Int16 buffer */
using FilterRingBuffer = RingBuffer<Int16>;

template <const unsigned short Grootte>

using StaticFilterRingBuffer = StaticRingBuffer<Int16,Grootte>;

class FilterFirInt16
{
public:
```

100 % Keine Probleme gefunden Zeile: 1 Zeichen: 1 GEMISCHT LF

Ausgabe

Ausgabe anzeigen von: Debuggen

```
"Opdracht4.exe" (Win32): "C:\Windows\System32\api-ms-win-core-file-l2-1-0.dll" geladen. Symbole wurden geladen.
"Opdracht4.exe" (Win32): "C:\Windows\System32\api-ms-win-core-synch-l1-2-0.dll" geladen. Symbole wurden geladen.
"Opdracht4.exe" (Win32): "C:\Windows\System32\uxtheme.dll" geladen. Symbole wurden geladen.
"Opdracht4.exe" (Win32): "C:\Windows\System32\msimg32.dll" geladen. Symbole wurden geladen.
"Opdracht4.exe" (Win32): "C:\Windows\System32\vcruntime140_1d.dll" geladen. Symbole wurden geladen.
"Opdracht4.exe" (Win32): "C:\Windows\System32\imm32.dll" geladen. Symbole wurden geladen.
"Opdracht4.exe" (Win32): "C:\Windows\System32\msctf.dll" geladen. Symbole wurden geladen.
"Opdracht4.exe" (Win32): "C:\Windows\System32\sspicli.dll" geladen. Symbole wurden geladen.
"Opdracht4.exe" (Win32): "C:\Windows\System32\cryptbase.dll" geladen. Symbole wurden geladen.
"Opdracht4.exe" (Win32): "C:\Program Files\Dell\Dell Data Protection\Security Tools\Authentication\Bin\DpOFeedb.dll" geladen.
"Opdracht4.exe" (Win32): "C:\Windows\System32\oleaut32.dll" geladen. Symbole wurden geladen.
"Opdracht4.exe" (Win32): "C:\Windows\System32\userenv.dll" geladen. Symbole wurden geladen.
"Opdracht4.exe" (Win32): "C:\Windows\System32\profapi.dll" geladen. Symbole wurden geladen.
"Opdracht4.exe" (Win32): "C:\Windows\System32\dwmapi.dll" geladen. Symbole wurden geladen.
"Opdracht4.exe" (Win32): "C:\Program Files\Dell\Dell Data Protection\Security Tools\Authentication\Bin\DpoSet.dll" geladen.
Begin=37
Begin=65
Time graph exit.
Das Programm "[11640] Opdracht4.exe" wurde mit Code 0 (0x0) beendet.
```

For questions and comments, contact me! I will try to solve problems or ambiguities as quickly as possible.

Arnhem, 30.01.2020

ir drs E.J Boks (ewout.boks@han.nl)