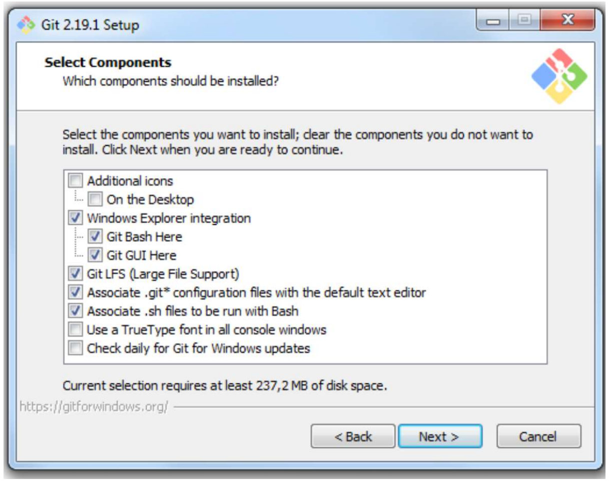
AFF3CT SIMULATOR

Specifications:
Operating System : Windows 10
Reference User Manual:
https://aff3ct.readthedocs.io/en/latest/user/introduction/introduction.html

SECTION 1: INSTALLATION GUIDE

1.1 Git Installation

This project use Git as the version-control system to execute the source code. Download git from https://git-scm.com/downloads. During installation perform selection of settings as shown in figure 1, figure2, figure3.



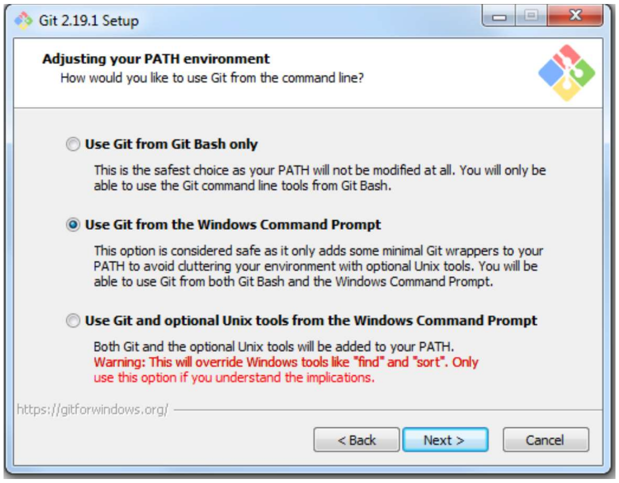
Select Components
Which components should be installed?

Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.

- □ Additional icons
  - □ On the Desktop
- ☑ Windows Explorer integration
  - ☑ Git Bash Here
  - ☑ Git GUI Here
- ☑ Git LFS (Large File Support)
- ☑ Associate .git* configuration files with the default text editor
- ☑ Associate .sh files to be run with Bash
- □ Use a TrueType font in all console windows
- □ Check daily for Git for Windows updates

Current selection requires at least 237,2 MB of disk space.

Figure1: Select Components



Adjusting your PATH environment
How would you like to use Git from the command line?

○ **Use Git from Git Bash only**

This is the safest choice as your PATH will not be modified at all. You will only be able to use the Git command line tools from Git Bash.

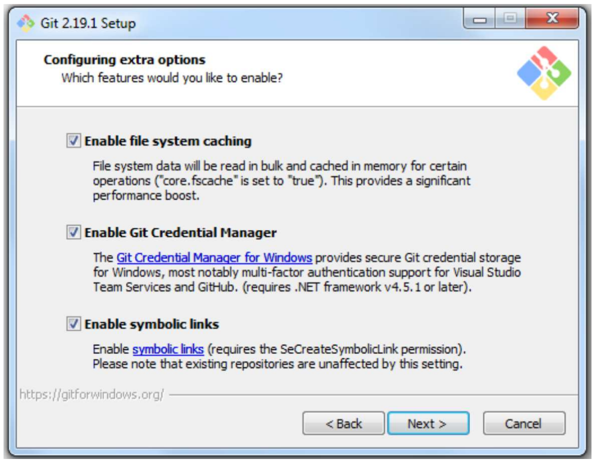
● **Use Git from the Windows Command Prompt**

This option is considered safe as it only adds some minimal Git wrappers to your PATH to avoid cluttering your environment with optional Unix tools. You will be able to use Git from both Git Bash and the Windows Command Prompt.

○ **Use Git and optional Unix tools from the Windows Command Prompt**

Both Git and the optional Unix tools will be added to your PATH.
Warning: This will override Windows tools like "find" and "sort". Only use this option if you understand the implications.

Figure 2: Adding Path Environment



Configuring extra options

Which features would you like to enable?

☑ **Enable file system caching**

File system data will be read in bulk and cached in memory for certain operations ("core.fscache" is set to "true"). This provides a significant performance boost.

☑ **Enable Git Credential Manager**

The Git Credential Manager for Windows provides secure Git credential storage for Windows, most notably multi-factor authentication support for Visual Studio Team Services and GitHub. (requires .NET framework v4.5.1 or later).

☑ **Enable symbolic links**

Enable symbolic links (requires the SeCreateSymbolicLink permission). Please note that existing repositories are unaffected by this setting.

https://gitforwindows.org/

Figure 3: Configuring extra options

Once the installation is complete make sure to add GIT to the environment path of user account. Refer to following step for adding variables:
Start -> Type ‘Edit environment variables for your account’ -> Select PATH -> Edit -> Paste the path of Git where it is installed.

1.1.1 Clone AFF3CT from GitHub

1. Extract the source code from GitHub using following command:

```
git clone --recursive https://github.com/aff3ct/aff3ct.git
cd aff3ct
```

The AFF3CT repository contains some dependencies to other repositories. Technically those dependencies are managed by the Git submodule feature. By default the submodules are not downloaded during the `git clone` process this is why the `--recursive` option has been added.

Manually update the AFF3CT submodules, as per below command.

```
git submodule update --init --recursive
```

1.2 Compilation with a Visual Studio 2019 solution:

Install the community version of Windows Visual studio 2019 from https://visualstudio.microsoft.com/downloads/ and install all packages. After downloading all the packages are to be selected for installation. This is done as AFF3CT requires additional libraries to be run.

1. To generate AFF3CT solution, open the `$AFF3CT_ROOT` folder from the IDE.

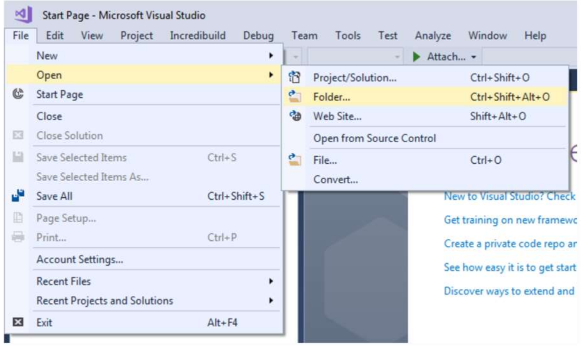
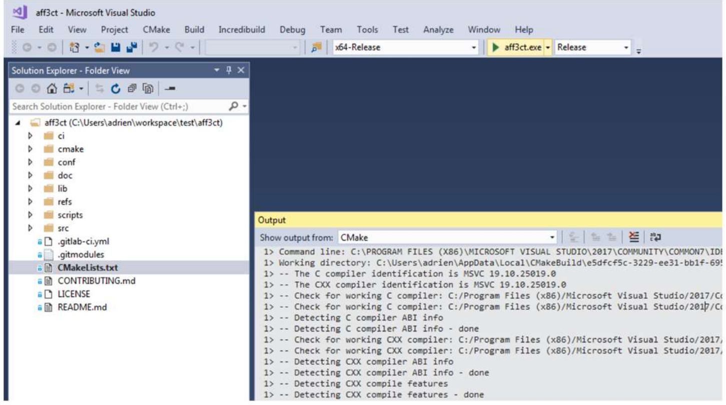

Start Page - Microsoft Visual Studio
File
Edit
View
Project
Incredibuild
Debug
Team
Tools
Test
Analyze
Window
Help
New
Open
Start Page
Close
Close Solution
Save Selected Items Ctrl+S
Save Selected Items As...
Save All Ctrl+Shift+S
Page Setup...
Print... Ctrl+P
Account Settings...
Recent Files
Recent Projects and Solutions
Exit Alt+F4
Project/Solution... Ctrl+Shift+O
Folder... Ctrl+Shift+Alt+O
Web Site... Shift+Alt+O
Open from Source Control
File... Ctrl+O
Convert...
Attach...
New to Visual Studio? Check
Get training on new framewo
Create a private code repo ar
See how easy it is to get start
Discover ways to extend and

Figure 4: To generate solution

2. Select the *Release* target and press the green play button aff3ct.exe to start the compilation.



```
1> Command line: C:\PROGRAM FILES (X86)\MICROSOFT VISUAL STUDIO\2017\COMMUNITY\COMMON7\IDE
1> Working directory: C:\Users\adrien\AppData\Local\CMakeBuild\e5dfcf5c-3229-ee31-bb1f-695
1> -- The C compiler identification is MSVC 19.10.25019.0
1> -- The CXX compiler identification is MSVC 19.10.25019.0
1> -- Check for working C compiler: C:/Program Files (x86)/Microsoft Visual Studio/2017/Co
1> -- Check for working C compiler: C:/Program Files (x86)/Microsoft Visual Studio/2017/Co
1> -- Detecting C compiler ABI info
1> -- Detecting C compiler ABI info - done
1> -- Check for working CXX compiler: C:/Program Files (x86)/Microsoft Visual Studio/2017/
1> -- Check for working CXX compiler: C:/Program Files (x86)/Microsoft Visual Studio/2017/
1> -- Detecting CXX compiler ABI info
1> -- Detecting CXX compiler ABI info - done
1> -- Detecting CXX compile features
1> -- Detecting CXX compile features - done
```

Figure 5: To generate AFF3CT.exe

3. Once AFF3CT is compiled you can browse the build by right clicking on `CMakeList.txt` > `Cache` > `Open Cache Folder`.

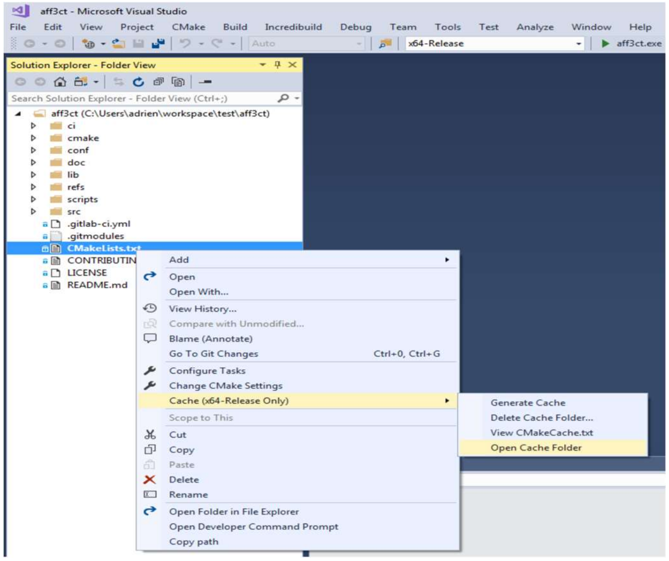


Figure 6: To run Release build

The compilation can also be started from the command line after calling the `%VS_PATH%\VC\Auxiliary\Build\vcvars64.bat` batch script (where `%VS_PATH%` is the location of Visual Studio on your system) and typing following command in prompt:

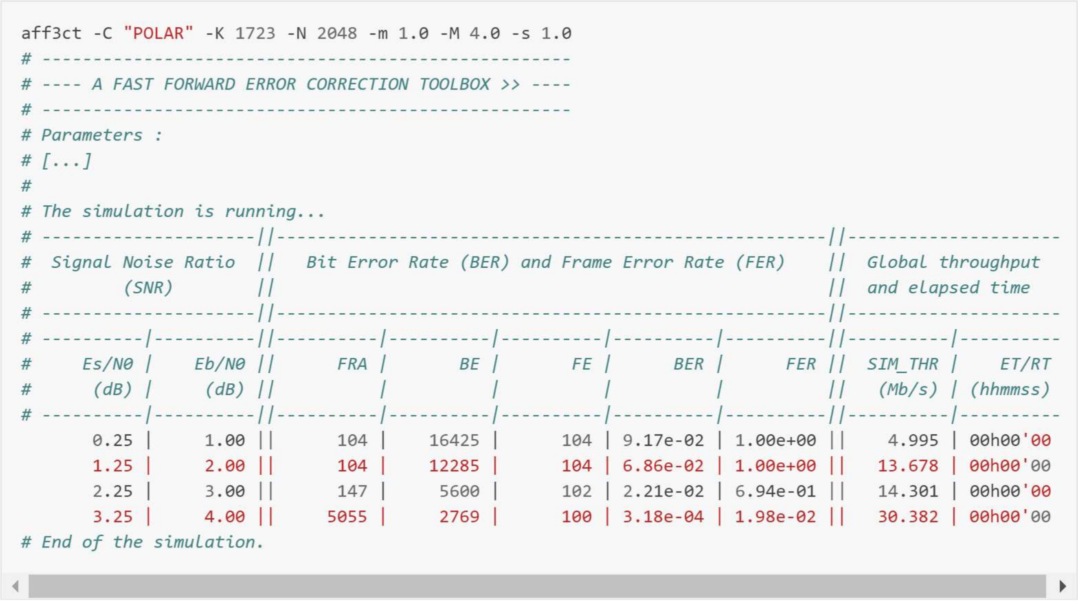
```
devenv /build Release aff3ct.sln
```

1.3 Installation

AFF3CT can be installed on system either by compiling it locally and then installing or by downloading remotely precompiled versions. In this case we have chosen the precompiled version technique as we don't intend to modify the code. Download one of the latest build from "http://aff3ct.github.io/download.html". Precompiled binaries are available for the most common operating systems : Windows, macOS and Linux.

SECTION 2: SIMULATION

The AFF3CT toolbox has a dedicated simulator for communication channels focused on the channel coding level. The simulator output will generate a table as shown in figure 7.



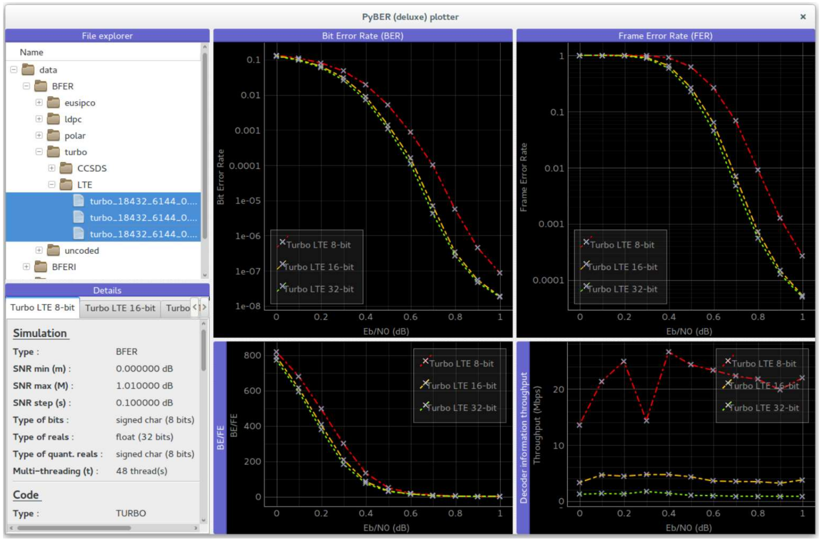
```
aff3ct -C "POLAR" -K 1723 -N 2048 -m 1.0 -M 4.0 -s 1.0
# ----------------------------------------------------
# ---- A FAST FORWARD ERROR CORRECTION TOOLBOX >> ----
# ----------------------------------------------------
# Parameters :
# [...]
#
# The simulation is running...
# ---------------------||------------------------------------------------------||---------------------
#  Signal Noise Ratio  ||   Bit Error Rate (BER) and Frame Error Rate (FER)    ||  Global throughput
#         (SNR)        ||                                                      ||  and elapsed time
# ---------------------||------------------------------------------------------||---------------------
# ----------|----------||----------|----------|----------|----------|----------||----------|----------
#     Es/N0 |    Eb/N0 ||      FRA |       BE |       FE |      BER |      FER ||  SIM_THR |    ET/RT
#      (dB) |     (dB) ||          |          |          |          |          ||   (Mb/s) | (hhmmss)
# ----------|----------||----------|----------|----------|----------|----------||----------|----------
       0.25 |     1.00 ||      104 |    16425 |      104 | 9.17e-02 | 1.00e+00 ||    4.995 | 00h00'00
       1.25 |     2.00 ||      104 |    12285 |      104 | 6.86e-02 | 1.00e+00 ||   13.678 | 00h00'00
       2.25 |     3.00 ||      147 |     5600 |      102 | 2.21e-02 | 6.94e-01 ||   14.301 | 00h00'00
       3.25 |     4.00 ||     5055 |     2769 |      100 | 3.18e-04 | 1.98e-02 ||   30.382 | 00h00'00
# End of the simulation.
```

Figure 7: Polar code for Information bits 1723

Following commands can be used in cmd prompt after navigating to the folder where it is stored:

1. ./bin/aff3ct -C "POLAR" -K 1723 -N 2048 -m 1.0 -M 4.0 -s 1.0
2. ./bin/aff3ct-2.3.5.exe --sim-type "BFER" -C "TURBO" -K "2048" -m "0.0" -M "2.0" -s "0.1"

PyBER is the Python GUI developed by AFF3CT developers to display the AFF3CT outputs. This will display graphs of BER and FER versus Eb/No as shown in figure 8 for various block lengths.



Simulation

Type : BFER
SNR min (m) : 0.000000 dB
SNR max (M) : 1.010000 dB
SNR step (s) : 0.100000 dB
Type of bits : signed char (8 bits)
Type of reals : float (32 bits)
Type of quant. reals : signed char (8 bits)
Multi-threading (t) : 48 thread(s)

Code

Type : TURBO

Figure 8: Output of PyBER

1. Installation Path: https://github.com/aff3ct/PyBER.
2. Install Anaconda from https://www.anaconda.com/download/ (tested on Anaconda3-4.2.0-Windows-x86_64).
3. Keep clicking install and then Finish.
4. Once it is installed add Python to environment variable path for your account.
5. Run PyBER from terminal by typing following command:

```
python PyBER.py
```

6. A dialog box appears similar to figure 8.