

# WSL上でのOpenMXのインストール

本資料ではWindows 10/11のPCにOpenMX Ver. 3.9をインストールする手順を説明する。WSL、Intel one API Base Toolkit 及び HPC Toolkitを導入した後にOpenMXをインストールする。Windows 11の場合も基本的に同じであるが、WSLの導入はWindows 10と比較して容易である (Windows 11のWSL導入は本資料では議論しない)。

尾崎泰助  
東京大学物性研究所

2022年4月

## 備考

- 2022年4月の時点での情報に基づいており、最新 and/or 異なる環境ではうまくいかない場合もあるかも知れませんが、ご容赦下さい。
- 本資料は情報共有のため、公開するものです。インストール作業による障害に関しては自己責任でお願い致します。
- 本資料は様々なWeb上の情報に基づき、試行錯誤の上で成功した手順を記載しています。各Webの情報提供者の皆様に感謝致します。

# インストール作業の一覧

以下の手順により、Windows 10 PCにOpenMXをインストールできる

## 1. PCのスペックとディスク空き容量を確認する

後述するIntel one API Base Toolkit 及び HPC Toolkitのインストールにはディスクの空容量が**30 GB程度**、必要となる。十分なディスク空き容量があるか確認する。また、一般にCPUのクロック数が高いものほど、OpenMXの実行速度は高速になる。1.8GHz以上が推奨される。またOpenMXの実行速度はディスクの書き込み速度にも左右される。

## 2. WSLを導入する

WSL (Windows Subsystem for Linux) により、Windows PC上でLinux環境が実現できる。OpenMXの実行性能はWSLのバージョンに左右される。またVcXsrvによるXサーバの導入も推奨される。

## 3. Intel one API Base Toolkit 及び HPC Toolkitのインストール

one API Base ToolkitとHPC Toolkitにより、Intelコンパイラ、Math Kernel Library (MKL), Intel MPIが利用可能となる。

## 4. OpenMX Ver. 3.9のインストール

作業1-3の終了後にOpenMX Ver. 3.9 with patch3.9.9をインストールする。

# テストに用いたPC環境

## PC1

### PC

デバイス名: LAPTOP-OS9EEKFG  
プロセッサ: Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz  
実装RAM: 16.0 GB  
ディスク: SSD 最大読込速度 3000MB/s, 最大書込速度 1900MB/s

### Windows 10

エディション: Windows 10 Pro  
バージョン: 21H1  
システムの種類: 64ビット

## PC2

### PC

デバイス名: LAPTOP-OS9EEKFG  
プロセッサ: Intel(R) Core(TM) i7-10750H CPU @ 2.60GHz  
実装RAM: 16.0 GB  
ディスク: SSD 最大読込速度 3000MB/s, 最大書込速度 1900MB/s

### Windows 10

エディション: Windows 10 Home  
バージョン: 21H1  
システムの種類: 64ビット

# PCのスペックとディスク空き容量を確認する

後述するIntel one API Base Toolkit 及び HPC Toolkitのインストールにはディスクの空容量が30 GB程度、必要となる。十分なディスク空き容量があるか確認する。一般にCPUのクロック数が高いものほど、OpenMXの実行速度は高速になる。1.8GHz以上が推奨される。またOpenMXの実行速度はディスクの書き込み速度にも大きく左右される。

Intel one API Base Toolkitのインストールに必要なシステム要件は以下webに記載されているので、ご確認頂きたい。

<https://software.intel.com/content/www/us/en/develop/articles/intel-oneapi-base-toolkit-system-requirements.html>

Intel one API HPC Toolkitのインストールに必要なシステム要件は以下webに記載されているので、ご確認頂きたい。

<https://software.intel.com/content/www/us/en/develop/articles/intel-oneapi-hpc-toolkit-system-requirements.html>

# WSLを導入する #1

- WSL (Windows Subsystem for Linux) により、Windows PC上でLinux環境が実現できる。
- インストールの手順はマイクロソフトの公式サイトに記載されているので、そちらをご参照頂きたい。

<https://docs.microsoft.com/ja-jp/windows/wsl/install-win10>

- 上記のWebでは「Windows Insider 用の簡略化されたインストール」と「**手動インストールの手順**」の二つの方法が説明されており、私は**後者の方法**でインストールした。
- またgoogle等で「wsl インストール」と検索すれば、多数のサイトでインストール手順が解説されている。
- OpenMXの実行速度はWSLのバージョンに依存している。WSL1とWSL2を導入し、ベンチマーク計算を実施した上で適切なバージョンを選択して頂きたい。
- Linux ディストリビューションとしてOpenMXの安定稼働が確認できている**Ubuntu 18.04 LTSを推奨する**。(インストールで設定したパスワードは今後、使用するため、メモしておく)

# WSLを導入する #2

- WSLを導入後にコンソールwindowを開いて、.bashrcをエディタを用いて修正する。(右例ではemacsを使用しているが、自身の使い慣れたエディタをする)

```
t-ozaki@LAPTOP-OS9EEKFG: ~  
t-ozaki@LAPTOP-OS9EEKFG: ~$ emacs -nw .bashrc
```

- .bashrcの最後に下記行を付け加えて、セーブする。

## WSL1の場合

```
export DISPLAY=:0.0  
export LIBGL_ALWAYS_INDIRECT=1  
alias xterm='xterm -rv -fn 12x24 &'  
alias uxterm='uxterm -fa "DejaVu Sans Mono:style=Book" -fd IPAGothic:style=Regular -fs 12 &'  
alias cdh='cd /mnt/c/Users/Taisuke Ozaki/work/'
```

WSLとWindowsのファイルシステムは別管理されているため、WSL上で作業する際にもWindowsファイルシステムに移動した方が作業が簡単になる。aliasでcdhを設定しておくことで、windowsファイルシステムに簡単に移動できる。

## WSL2の場合

```
export DISPLAY=$(awk '/nameserver / {print $2; exit}' /etc/resolv.conf 2>/dev/null):0  
export LIBGL_ALWAYS_INDIRECT=1  
alias xterm='xterm -rv -fn 12x24 &'  
alias uxterm='uxterm -fa "DejaVu Sans Mono:style=Book" -fd IPAGothic:style=Regular -fs 12 &'  
alias cdh='cd /mnt/c/Users/Taisuke Ozaki/work/'
```

この部分をご自身の環境に合わせて変更

この部分をご自身の環境に合わせて変更

# WSLを導入する #3

- Intel one API Base Toolkit 及び HPC Toolkitのインストールの際にXサーバが利用されているようである。そこで「Xサーバ」を導入する。
- Xサーバを導入することにより、Linux上の様々なGUIソフトが利用できる。またCUIであるX-terminalを複数、開いて作業することが可能となり、作業効率が向上する。
- Xサーバを導入するにはVcXsrv、Cygwin/X、Xmingなど複数のソフトウェアが提供されているが、私はVcXsrvを使用している。下記サイトにVcXsrvのインストール方法が記載されている。

<https://atmarkit.itmedia.co.jp/ait/articles/1812/06/news040.html>

<https://sourceforge.net/projects/vcxsrv/>

Xサーバの導入後にWSLのコンソールwindowから、xtermやuxterm (日本語使用可)を立ち上げることが出来る。注意: 前頁にて説明した.bashrcへのalias設定により、下記コマンドで立ち上がる。

```
t-ozaki@LAPTOP-OS9EEKFG: ~
```

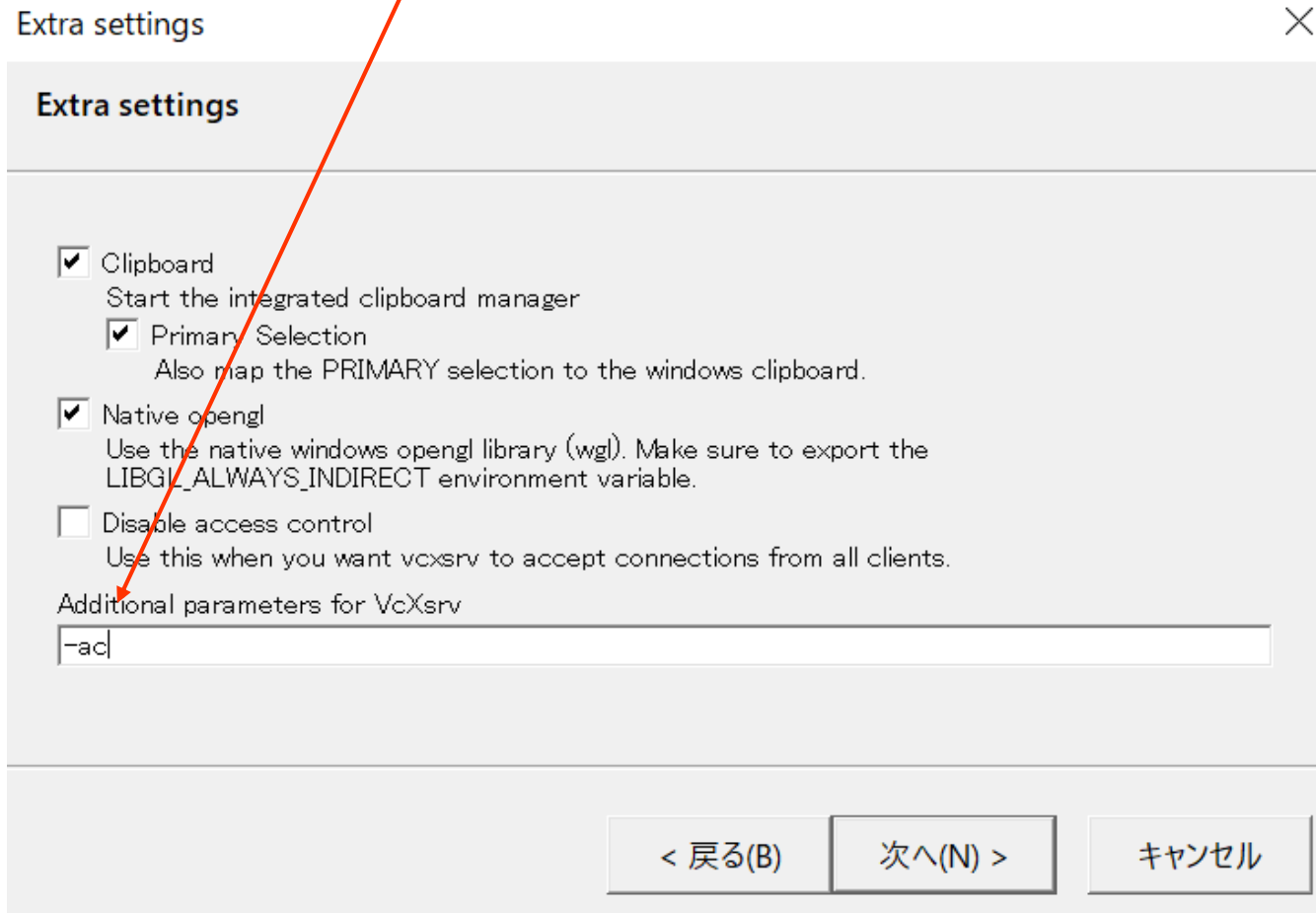
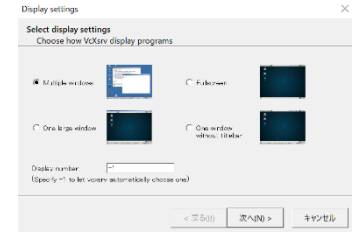
```
t-ozaki@LAPTOP-OS9EEKFG:~$ xterm
```

```
t-ozaki@LAPTOP-OS9EEKFG: ~
```

```
t-ozaki@LAPTOP-OS9EEKFG:~$ uxterm
```

# WSLを導入する #4

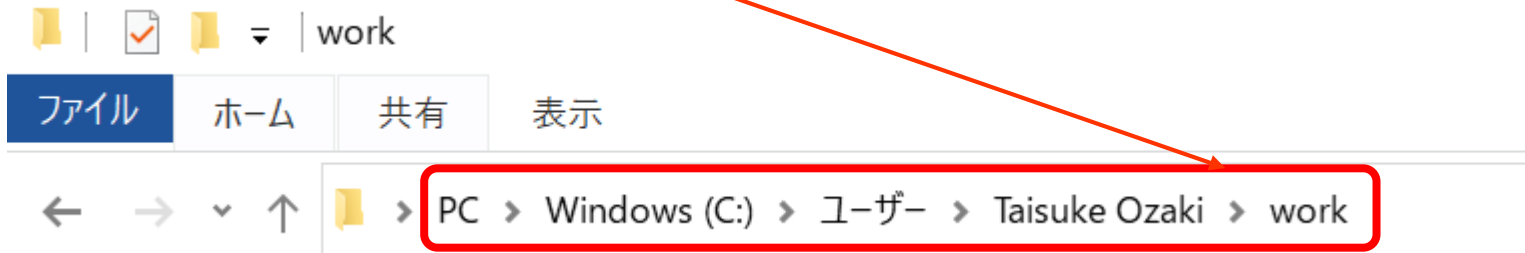
VcXsrvを用いてXサーバを導入し、かつWSL2を使用している場合には、Xサーバの「Display setting」中(右図)にて「Additional parameters for VcXsrv」に **-ac** を指定する。





# WSLを導入する #5

Windows エクスプローラでWSL用の作業ディレクトリを作成しておく  
私の場合には以下のようにworkを作成してある。



これに対応して、.bashrcに以下のaliasが設定してある。

```
alias cdh='cd /mnt/c/Users/Taisuke Ozaki/work/'
```

.bashrcに関しては「WSLを導入する #2」のスライドを参照のこと

.bashrcはWSLが管理するホームディレクトリに存在していることに注意。WSLのコマンドプロンプトから右のコマンドで確認できる。

```
t-ozaki@LAPTOP-OS9EEKFG: ~  
t-ozaki@LAPTOP-OS9EEKFG:~$ cd  
t-ozaki@LAPTOP-OS9EEKFG:~$ ls -A .bashrc  
.bashrc  
t-ozaki@LAPTOP-OS9EEKFG:~$ tail .bashrc  
fi  
fi  
export DISPLAY=$(awk '/nameserver / {print $2; exit}' /etc/resolv.conf 2>/dev/null):0  
export LIBGL_ALWAYS_INDIRECT=1  
alias xterm='xterm -rv -fn 12x24 &'  
alias uxterm='uxterm -fa "DejaVu Sans Mono:style=Book" -fd IPAGothic:style=Regular -fs 12 &'  
alias cdh='cd /mnt/c/Users/Taisuke Ozaki/work/'  
source /opt/intel/oneapi/setvars.sh  
t-ozaki@LAPTOP-OS9EEKFG:~$
```

# WSLを導入する #6

WSLのコンソールウィンドで次の**必須**コマンドを実行する。

```
sudo apt update
sudo apt upgrade
sudo apt install make
sudo apt install gcc
sudo apt install g++
sudo apt install libnss3
```

適宜、必要なアプリをインストールする

```
sudo apt install gnuplot
sudo apt install gnuplot-x11
sudo apt install emacs
sudo apt install vim
```

等

# Intel one API Base Toolkitのインストール #1

Intel one API Base Toolkitには以下のライブラリ等が含まれる。

Intel® oneAPI Collective Communications Library

Intel® oneAPI Data Analytics Library

Intel® oneAPI Deep Neural Network Library

Intel® oneAPI DPC++/C++ Compiler

Intel® oneAPI DPC++ Library

Intel® oneAPI Math Kernel Library

Intel® oneAPI Threading Building Blocks

Intel® oneAPI Video Processing Library

Intel® Advisor

Intel® Distribution for GDB\*

Intel® Distribution for Python\*

Intel® DPC++ Compatibility Tool

Intel® Integrated Performance Primitives

Intel® VTune™ Profiler

Optional: Intel® FPGA Add-on for oneAPI Base Toolkit

Base ToolkitはEnd User License Agreementの規約に則り、その使用が認められている。詳細は下記URLをご確認頂きたい。

<https://software.intel.com/content/www/us/en/develop/articles/end-user-license-agreement.html>

# Intel one API Base Toolkitのインストール #2

- (1) Intel web上からBase Kit (I\_BaseKit\_p\_2021.3.0.3219\_offline.sh) をダウンロードする。

[https://registrationcenter-download.intel.com/akdlm/irc\\_nas/17977/I\\_BaseKit\\_p\\_2021.3.0.3219\\_offline.sh](https://registrationcenter-download.intel.com/akdlm/irc_nas/17977/I_BaseKit_p_2021.3.0.3219_offline.sh)

サイズが3.4GBあるので、ダウンロードにはそれなりに時間(4分程度)が掛かる。最新版のBase ToolKitを用いてinstallする場合には以下、versionを読みかえて下さい。

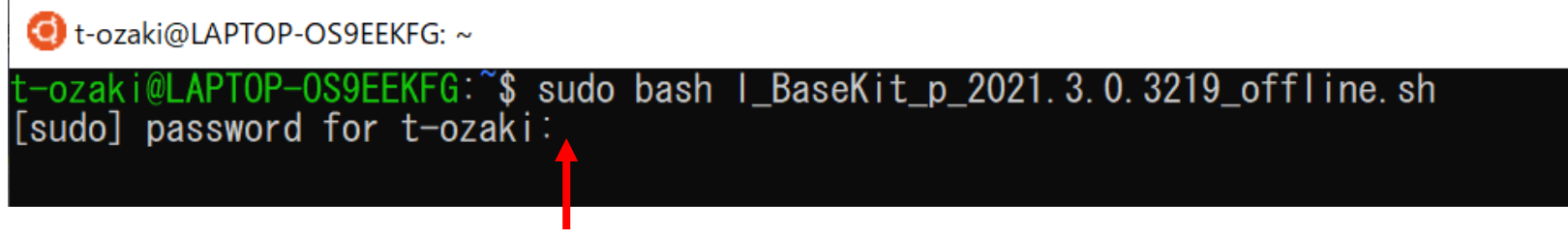
- (2) "I\_BaseKit\_p\_2021.3.0.3219\_offline.sh"を自身の"work"ディレクトリにコピーする。

# Intel one API Base Toolkitのインストール #3

(3) WSLのコマンドプロンプトから次のコマンドを実行する

```
sudo bash I_BaseKit_p_2021.3.0.3219_offline.sh
```

```
t-ozaki@LAPTOP-OS9EEKFG: ~  
t-ozaki@LAPTOP-OS9EEKFG:~$ sudo bash I_BaseKit_p_2021.3.0.3219_offline.sh  
[sudo] password for t-ozaki:
```



Ubuntuをインストールした際に設定したpasswordを入力し、しばらく待っているとSystem requirementのチェックやEnd User License Agreementの承諾に関する質問(ポップアップ画面)があり、承諾するとインストールが始まる。途中でいくつかの質問事項がある。

```
t-ozaki@LAPTOP-OS9EEKFG: /mnt/c/Users/Taisuke Ozaki/work  
t-ozaki@LAPTOP-OS9EEKFG:/mnt/c/Users/Taisuke Ozaki/work$ sudo bash I_BaseKit_p_2021.3.0.3219_offline.sh  
[sudo] password for t-ozaki:  
Extract I_BaseKit_p_2021.3.0.3219_offline to /mnt/c/Users/Taisuke Ozaki/work/I_BaseKit_p_2021.3.0.3219_offline...  
[#####]  
Extract I_BaseKit_p_2021.3.0.3219_offline completed!  
XStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-root'
```



正常に終了すると、  
/opt/intel/oneapi/  
以下にDPC++/C++  
Compilerや Math Kernel  
Libraryなどがインストールされる。

# Intel one API HPC Toolkitのインストール #1

Intel one API HPC Toolkitには以下のライブラリ等が含まれる。

Intel® oneAPI DPC++/C++ Compiler

Intel® oneAPI Fortran Compiler

Intel® C++ Compiler Classic

Intel® Cluster Checker

Intel® Inspector

Intel® MPI Library

Intel® Trace Analyzer and Collector

HPC ToolkitのBase Toolkitへの依存性のため、Base Toolkitをインストールした後にIntel one API HPC Toolkitをインストールしてください。

Base ToolkitはEnd User License Agreementの規約に則り、その使用が認められている。詳細は下記URLをご確認頂きたい。

<https://software.intel.com/content/www/us/en/develop/articles/end-user-license-agreement.html>

# Intel one API HPC Toolkitのインストール #2

- (1) Intel web上からHPC Kit (I\_HPCKit\_p\_2021.3.0.3230\_offline.sh) をダウンロードする。

[https://registrationcenter-download.intel.com/akdlm/irc\\_nas/17912/I\\_HPCKit\\_p\\_2021.3.0.3230\\_offline.sh](https://registrationcenter-download.intel.com/akdlm/irc_nas/17912/I_HPCKit_p_2021.3.0.3230_offline.sh)

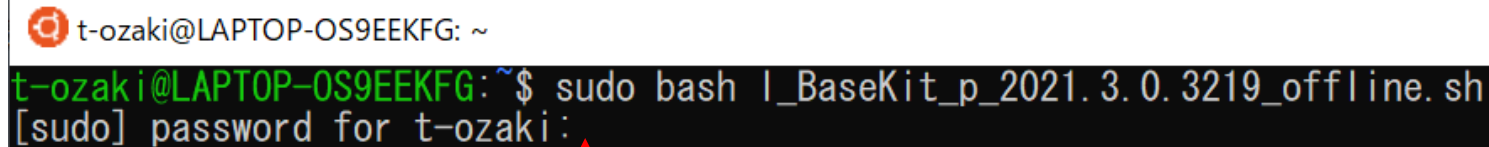
サイズが1.25GBあるので、ダウンロードにはそれなりに時間(2分程度)が掛かる。最新版のBase HPCKitを用いてinstallする場合には以下、versionを読みかえて下さい。

- (2) “I\_HPCKit\_p\_2021.3.0.3230\_offline.sh”を自身の  
“work”ディレクトリにコピーする。

# Intel one API HPC Toolkitのインストール #3

(6) WSLのコマンドプロンプトから次のコマンドを実行する

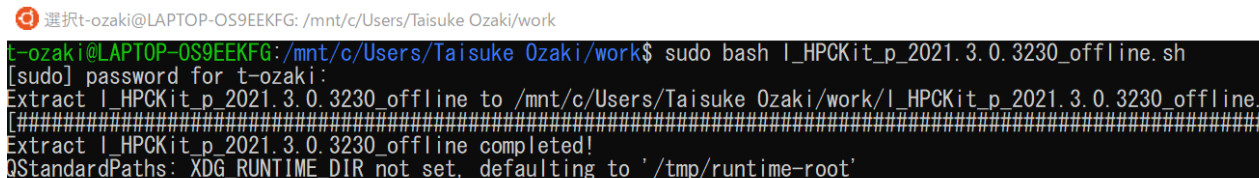
```
sudo bash I_HPCKit_p_2021.3.0.3230_offline.sh
```



```
t-ozaki@LAPTOP-OS9EEKFG: ~  
t-ozaki@LAPTOP-OS9EEKFG: ~$ sudo bash I_BaseKit_p_2021.3.0.3219_offline.sh  
[sudo] password for t-ozaki:
```

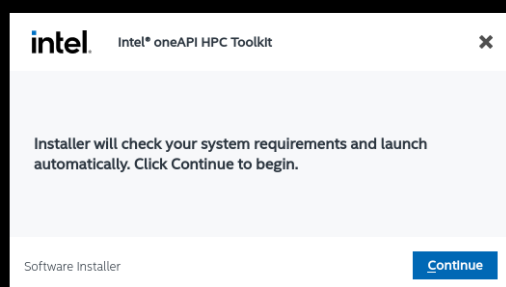
A red arrow points to the password prompt line in the terminal screenshot.

Ubuntuをインストールした際に設定したpasswordを入力し、しばらく待っているとSystem requirementのチェックやEnd User License Agreementの承諾に関する質問(ポップアップ画面)があり、承諾するとインストールが始まる。途中でいくつかの質問事項がある。



```
t-ozaki@LAPTOP-OS9EEKFG: /mnt/c/Users/Taisuke Ozaki/work  
t-ozaki@LAPTOP-OS9EEKFG: /mnt/c/Users/Taisuke Ozaki/work$ sudo bash I_HPCKit_p_2021.3.0.3230_offline.sh  
[sudo] password for t-ozaki:  
Extract I_HPCKit_p_2021.3.0.3230_offline to /mnt/c/Users/Taisuke Ozaki/work/I_HPCKit_p_2021.3.0.3230_offline..  
#####  
Extract I_HPCKit_p_2021.3.0.3230_offline completed!  
StandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-root'
```

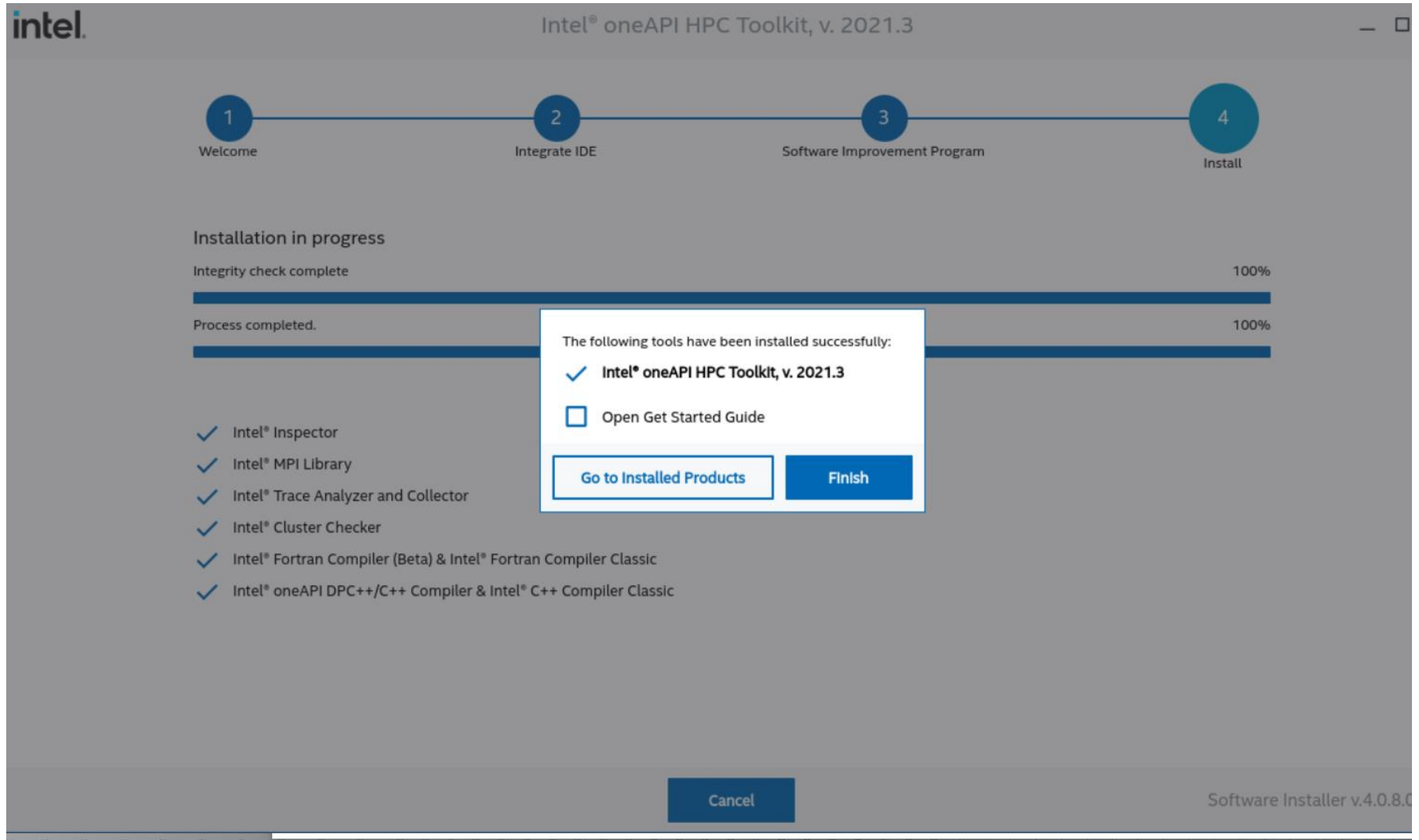
正常に終了すると、  
/opt/intel/oneapi/  
以下にFortranコンパイラ  
や MPI Libraryなどがイン  
ストールされる。





# Intel one API HPC Toolkitのインストール #4

正常にインストールできた際の最終画面



# Intel one API Base Toolkit 及び HPC Toolkitのインストールの確認 #1

(1) WSLのコマンドプロンプトより、次のコマンドを実行する。

```
source /opt/intel/oneapi/setvars.sh
```

(2) エディタを用いて.bashrcの最終行に  
source /opt/intel/oneapi/setvars.sh を付け加える。

```
t-ozaki@LAPTOP-OS9EEKFG: ~  
t-ozaki@LAPTOP-OS9EEKFG:~$ emacs -nw .bashrc
```

.bashrcの中身

```
export DISPLAY=$(awk '/nameserver / {print $2; exit}' /etc/resolv.conf 2>/dev/null):0  
export LIBGL_ALWAYS_INDIRECT=1  
alias xterm='xterm -rv -fn 12x24 &'  
alias uxterm='uxterm -fa "DejaVu Sans Mono:style=Book" -fd IPAGothic:style=Regular -fs 12 &'  
alias cdh='cd /mnt/c/Users/Taisuke¥ Ozaki/work/'  
source /opt/intel/oneapi/setvars.sh
```

付け加えた一行

## Intel one API Base Toolkit 及び HPC Toolkitのインストールの確認 #2

(3) WSLのコマンドプロンプトより、次のコマンドを実行する。バージョン情報が表示されるなら、インストールが正常に行われている。

```
t-ozaki@LAPTOP-OS9EEKFG: ~  
t-ozaki@LAPTOP-OS9EEKFG:~$ icc --version  
icc (ICC) 2021.3.0 20210609  
Copyright (C) 1985-2021 Intel Corporation. All rights reserved.  
  
t-ozaki@LAPTOP-OS9EEKFG:~$ ifort --version  
ifort (IFORT) 2021.3.0 20210609  
Copyright (C) 1985-2021 Intel Corporation. All rights reserved.  
  
t-ozaki@LAPTOP-OS9EEKFG:~$ mpirun --version  
Intel(R) MPI Library for Linux* OS, Version 2021.3 Build 20210601 (id: 6f90181f1)  
Copyright 2003-2021, Intel Corporation.  
t-ozaki@LAPTOP-OS9EEKFG:~$
```

# OpenMX Ver. 3.9のインストール #1

<http://www.openmx-square.org/>

Welcome to OpenMX  
Open source package for Material explorer

## (1) OpenMXのWebサイトから

openmx3.9.tar.gz  
patch3.9.9.tar.gz

をダウンロードする。

“Download”をクリックすると以下のページに遷移する。

openmx3.9.tar.gz    patch3.9.9.tar.gz



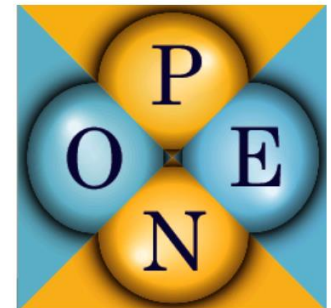
それぞれ、以下をクリックすることでダウンロードできる。

## Download of OpenMX

Available packages in terms of **GNU-GPL Version 3 (GPLv3)**

- |   |  |                            |
|---|--|----------------------------|
| • <a href="#">openmx3.9</a> (release date: 03/Dec./2019, 158 MB)  | + <a href="#">patch</a> (21/Aug./2021) | <a href="#">README.txt</a> |
| • <a href="#">openmx3.8</a> (release date: 03/April/2016, 136 MB) | + <a href="#">patch</a> (12/June/2018) | <a href="#">README.txt</a> |
| • <a href="#">openmx3.7</a> (release date: 23/May/2013, 112 MB)   | + <a href="#">patch</a> (21/Feb./2015) | <a href="#">README.txt</a> |

- **What's new**
  - [Patch3.9.6 to OpenMX Ver 3.9 \(Aug. 21, 2021\)](#)
  - [Patch3.9.2 to OpenMX Ver 3.9 \(Feb. 11, 2020\)](#)
- **What is OpenMX?**
- **Download**
- **Manual of Ver. 3.9**
- **Manual of Ver. 3.8**
- **Technical Notes**
- **Video Lectures**
- **Publications**
- **OpenMX Forum**
- **OpenMX Viewer**
- **Workshop**
- **Database of Results**
- **Database of VPS and PAO**
  - [Ver. 2019](#)
  - [Ver. 2019 for core excitations](#)
- **ADPACK**
- **Miscellaneous informations**
- **Contributors**



# OpenMX Ver. 3.9のインストール #2

(2) openmx3.9.tar.gzとpatch3.9.9.tar.gzを"work"ディレクトリにコピーする。 .bashrc中の"alias cdh"で指定したディレクトリ下に置くと良い。

(3) WSLのコンソールにて以下を実行する。

```
$ tar zxvf openmx3.9.tar.gz
$ cp ./patch3.9.9.tar.gz openmx3.9/source
$ cd openmx3.9/source
$ tar zxvfmp patch3.9.9.tar.gz
$ mv kpoint.in ../work/
```

(4) openmx3.9/sourceのmakefile中で以下の様に指定する(エディタを用いて)。

```
MKLROOT = /opt/intel/oneapi/mkl/latest
CC = mpiicc -O3 -xHOST -ip -no-prec-div -qopenmp -I${MKLROOT}/include -I${MKLROOT}/include/fftw
FC = mpiifort -O3 -xHOST -ip -no-prec-div -qopenmp
LIB= -L${MKLROOT}/lib/intel64 -lmkl_scalapack_lp64 -lmkl_intel_lp64 -lmkl_intel_thread -lmkl_core -lifcore -lmkl_blacs_intelmpi_lp64 -liomp5 -lpthread -lm -ldl
```

**LIBの行は一行であることに注意**

(5) WSLのコンソールにて以下を実行する。

```
$ make all          インストールが正常に終了すると"openmx3.9/work"中
$ make install      に行うファイル:openmxが生成される。
```

# OpenMX Ver. 3.9のベンチマーク計算 #1

14個の入力ファイルに対して自動的に計算を実行し、過去の計算結果と比較するruntestを行った。二つのWindows 10 PCに対して計算を実施。

## PC1

### PC

デバイス名: LAPTOP-OS9EEKFG  
プロセッサ: Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz  
実装RAM: 16.0 GB  
ディスク: SSD 最大読込速度 3000MB/s, 最大書込速度 1900MB/s

### Windows 10

エディション: Windows 10 Pro  
バージョン: 21H1  
システムの種類: 64ビット

## PC2

### PC

デバイス名: LAPTOP-OS9EEKFG  
プロセッサ: Intel(R) Core(TM) i7-10750H CPU @ 2.60GHz  
実装RAM: 16.0 GB  
ディスク: SSD 最大読込速度 3000MB/s, 最大書込速度 1900MB/s

### Windows 10

エディション: Windows 10 Home  
バージョン: 21H1  
システムの種類: 64ビット

runtestに関してはwebを参照のこと。

[http://www.openmx-square.org/openmx\\_man3.9jp/node17.html](http://www.openmx-square.org/openmx_man3.9jp/node17.html)

# OpenMX Ver. 3.9のベンチマーク計算 #2

## PC1によるruntestの結果の一例

**WSL1:** mpirun -np 4 ./openmx -runtest -nt 1

1 input_example/Benzene.dat	Elapsed time(s)= 10.35	diff Utot= 0.000000000031	diff Force= 0.000000000003
2 input_example/C60.dat	Elapsed time(s)= 48.24	diff Utot= 0.000000000000	diff Force= 0.000000000001
3 input_example/CO.dat	Elapsed time(s)= 23.61	diff Utot= 0.000000000096	diff Force= 0.000000000262
4 input_example/Cr2.dat	Elapsed time(s)= 21.58	diff Utot= 0.000000000944	diff Force= 0.000000000294
5 input_example/Crys-MnO.dat	Elapsed time(s)= 84.86	diff Utot= 0.000000000003	diff Force= 0.000000000004
6 input_example/GaAs.dat	Elapsed time(s)= 99.17	diff Utot= 0.000000000000	diff Force= 0.000000000000
7 input_example/Glycine.dat	Elapsed time(s)= 11.67	diff Utot= 0.000000000001	diff Force= 0.000000000000
8 input_example/Graphite4.dat	Elapsed time(s)= 7.99	diff Utot= 0.000000000006	diff Force= 0.000000000127
9 input_example/H2O-EF.dat	Elapsed time(s)= 8.70	diff Utot= 0.000000000000	diff Force= 0.000000000002
10 input_example/H2O.dat	Elapsed time(s)= 8.13	diff Utot= 0.000000000000	diff Force= 0.000000000011
11 input_example/HMn.dat	Elapsed time(s)= 29.00	diff Utot= 0.000000000131	diff Force= 0.000000000021
12 input_example/Methane.dat	Elapsed time(s)= 6.54	diff Utot= 0.000000000016	diff Force= 0.000000000001
13 input_example/Mol_MnO.dat	Elapsed time(s)= 17.34	diff Utot= 0.000000000370	diff Force= 0.000000000118
14 input_example/Ndia2.dat	Elapsed time(s)= 8.71	diff Utot= 0.000000000001	diff Force= 0.000000000000

Total elapsed time (s) 385.90

runtestに関しては以下のwebを参照のこと。

[http://www.openmx-square.org/openmx\\_man3.9jp/node17.html](http://www.openmx-square.org/openmx_man3.9jp/node17.html)

# OpenMX Ver. 3.9のベンチマーク計算 #3

## PC2によるruntestの結果の一例

**WSL1:** mpirun -np 6 ./openmx -runtest -nt 1

1 input_example/Benzene.dat	Elapsed time(s)=	5.45	diff Utot=	0.000000000040	diff Force=	0.000000000012
2 input_example/C60.dat	Elapsed time(s)=	24.39	diff Utot=	0.000000000000	diff Force=	0.000000000000
3 input_example/CO.dat	Elapsed time(s)=	13.21	diff Utot=	0.000000000096	diff Force=	0.000000000261
4 input_example/Cr2.dat	Elapsed time(s)=	9.92	diff Utot=	0.000000000907	diff Force=	0.000000000172
5 input_example/Crys-MnO.dat	Elapsed time(s)=	24.60	diff Utot=	0.000000000003	diff Force=	0.000000000005
6 input_example/GaAs.dat	Elapsed time(s)=	38.56	diff Utot=	0.000000000000	diff Force=	0.000000000000
7 input_example/Glycine.dat	Elapsed time(s)=	6.48	diff Utot=	0.000000000001	diff Force=	0.000000000000
8 input_example/Graphite4.dat	Elapsed time(s)=	4.32	diff Utot=	0.000000000006	diff Force=	0.000000000124
9 input_example/H2O-EF.dat	Elapsed time(s)=	4.99	diff Utot=	0.000000000000	diff Force=	0.000000000002
10 input_example/H2O.dat	Elapsed time(s)=	3.97	diff Utot=	0.000000000000	diff Force=	0.000000000011
11 input_example/HMn.dat	Elapsed time(s)=	20.27	diff Utot=	0.000000000131	diff Force=	0.000000000021
12 input_example/Methane.dat	Elapsed time(s)=	3.15	diff Utot=	0.000000000011	diff Force=	0.000000000001
13 input_example/Mol_MnO.dat	Elapsed time(s)=	10.51	diff Utot=	0.000000000370	diff Force=	0.000000000118
14 input_example/Ndia2.dat	Elapsed time(s)=	4.98	diff Utot=	0.000000000001	diff Force=	0.000000000000

Total elapsed time (s)      174.79

runtestに関しては以下のwebを参照のこと。

[http://www.openmx-square.org/openmx\\_man3.9jp/node17.html](http://www.openmx-square.org/openmx_man3.9jp/node17.html)



# OpenMX Ver. 3.9のベンチマーク計算 #4

runtestの計算時間を以下にまとめる。

	WSL バージョン	MPI procs.	OMP threads:	計算時間 (秒)
PC1	1	4	1	385.90
	1	2	2	1805.65
	2	4	1	275.30
	2	2	2	275.55
PC2	1	6	1	174.79
	2	6	1	196.43

WSLのバージョンの適切な選択はPCに依存していることが分かる。  
下記Webに掲載されたクラスター計算機の計算時間と比較しても、  
十分な計算速度であることが分かる。

[http://www.openmx-square.org/openmx\\_man3.9jp/node17.html](http://www.openmx-square.org/openmx_man3.9jp/node17.html)

メモリを十分に積んだPCであれば、様々な計算が可能であろう。