MKL latest Building record

Here are three key points to make MKL run.

- Correct OpenCL SDK **include** directories (not set through install)
  - This two directories should contain sycl/sycl.hpp and CL/sycl.hpp, (for build C++ only. If only C language support is used, then it is not required)
- Correct Environment variable: **MKLROOT** and **CMPLR_ROOT**, which should direct containing lib include and bin, for using .cmake to find modules. If CMakeList.txt is not in used, then it is not required.
- The directory which contain the MKL lib and bin (which are mkl_intel_ilp64_dll.lib, mkl_intel_thread_dll.lib, mkl_intel_thread.2.dll, mkl_core_dll.lib, mkl_core.2.dll, libiomp5md.lib) have to exist on **PATH** (set cmake command of link_directory is not work :-(

By default installation on windows, the setting variables will be:

add to **PATH**:

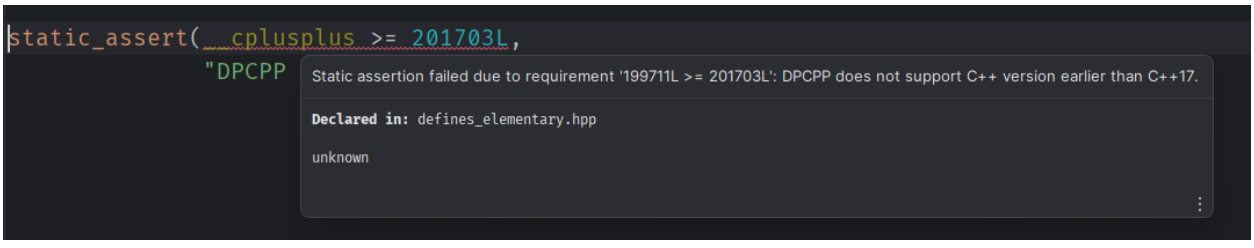
```
C:\Program Files (x86)\Intel\oneAPI\mkl\latest\include
C:\Program Files (x86)\Intel\oneAPI\mkl\latest\bin
C:\Program Files (x86)\Intel\oneAPI\mkl\latest\lib
C:\Program Files (x86)\Intel\oneAPI\compiler\latest\bin
C:\Program Files (x86)\Intel\oneAPI\compiler\latest\include
C:\Program Files (x86)\Intel\oneAPI\compiler\latest\lib
```

add to system environment variables:

MKLROOT	C:\Program Files (x86)\Intel\oneAPI\mkl\latest
CMPLR_ROOT	C:\Program Files (x86)\Intel\oneAPI\compiler\latest

Building fail using MSVC with cl.exe

The MSVC will always set __**cplusplus** marco to 199711L, which will make openCL SDK build fail through static_assert:



However, by the end of 2018, MicroSoft finally fix this bug through provided additional compiler flag.
https://devblogs.microsoft.com/cppblog/msvc-now-correctly-reports-__cplusplus/

Add this on CMakeList.txt will fix the problem:

```
if(MSVC)
set(CMAKE_CXX_FLAGS "${CMAKE_CXX_FLAGS} /Zc:__cplusplus")
# for passing correct __cplusplus macro to the compiler
endif()
```

Working demo:

Reference CMakeList.txt (work well on MSVC2022, with MSBuild Tool, MKL latest SDK)

```
cmake_minimum_required(VERSION 3.26)
project(mkl_helloword_p)

set(CMAKE_CXX_STANDARD 17)
if(MSVC)
set(CMAKE_CXX_FLAGS "${CMAKE_CXX_FLAGS} /Zc:__cplusplus")
# for passing correct __cplusplus macro to the compiler
endif()
find_package(MKL CONFIG REQUIRED)
#message(STATUS "${MKL_IMPORTED_TARGETS}") #Provides available list of targets based on input
add_executable(mkl_helloword main.cpp)
set(CL_INCLUDE "$ENV{CMPLR_ROOT}/include" "$ENV{CMPLR_ROOT}/include/sycl/")

target_compile_options(mkl_helloword PUBLIC $<TARGET_PROPERTY:MKL::MKL,INTERFACE_COMPILE_OPTIONS>)
target_include_directories(mkl_helloword PUBLIC ${CL_INCLUDE} $<TARGET_PROPERTY:MKL::MKL,INTERFACE_INCLUDE_DIRECTORIES>)
target_link_libraries(mkl_helloword PUBLIC $<LINK_ONLY:MKL::MKL>)
```

```
#include<stdio.h>
#include<oneapi/mkl.hpp>

int main()
{
    float* A, * B;
    int a = 1, b = 1;
    int n = 5;
    A = (float*)mkl_malloc(n * 1 * sizeof(float), 64);
    B = (float*)mkl_malloc(n * 1 * sizeof(float), 64);
    printf("The 1st vector is ");
    for (int i = 0; i < n; i++) {
        A[i] = i;
        printf("%2.0f", A[i]);
    }
    printf("\n");
    printf("The 2st vector is ");
    for (int i = 0; i < n; i++) {
        B[i] = i + 1;
        printf("%2.0f", B[i]);
    }
    printf("\n");

    cblas_saxpby(n, a, A, 1, b, B, 1);
    printf("The a*A+b*B is ");
    for (int i = 0; i < n; i++) {
        printf("%2.0f", B[i]);
    }
    printf("\n");
    mkl_free(A);
    mkl_free(B);
    getchar();
```

```
    return 0;
}
```

Result:

```
The 1st vector is  0 1 2 3 4
The 2st vector is  1 2 3 4 5
The a*A+b*B is  1 3 5 7 9
```