OpenEXR 2.1.0 (Win64, icl_14.0, vc10) BUILD Journal
(updated: 01/02/2014, Mike Wong@artixels)

(http://www.openexr.com/downloads.html)

IlmBase

It's important to expand the archive into a hierarchy similar to the below:
`c:\_OpenSource\source\OpenEXR_V2.1_vc10_icc\openexr_2.1.0\ilmbase_2.1.0`

and the solution should prepare everything and have them built into:
`c:\_OpenSource\source\OpenEXR_V2.1_vc10_icc\Deploy`

1. Create vc10 solution

Convert the included vc8 solution into vs2010.

2. Use Intel C++ compiler

Converted to icc environment inside visual studio using "Use Intel C++"

3. Build createDLL

`createDLL.exe` now resides in 2 locations:
`c:\_OpenSource\source\OpenEXR_V2.1_vc9_icc\Deploy\bin\x64\Release\createDLL.exe`
`C:\_OpenSource\source\OpenEXR_V2.1_vc9_icc\openexr_2.1.0\ilmbase-2.1.0\vc\vc8\IlmBase\x64\Release\createDLL.exe`

4. Intel C++ compiler libraries

Necessary to copy the following Intel compiler libraries (`C:\Program Files (x86)\Intel\Composer XE 2011 SP1\compiler\lib\intel64`) to

`C:\_OpenSource\source\OpenEXR_V2.1_vc9_icc\openexr_2.1.0\ilmbase-2.1.0\vc\vc8\IlmBase\x64\Release`

- `libdecimal.lib`
- `libirc.lib`
- `libmmd.lib`
- `svml_dispmd.lib`

5. Half and HalfTest

should be okay

6. Iex

`IexExport.h` in this 2.1.0 controls whether to build library or not, in order to build, we need both `OPENEXR_DLL` and `IEX_EXPORTS` both declared in order to build a working library, and then we need to add `PLATFORM_WINDOWS` in order to avoid a build error. So do the following:

Via Properties -> C/C++ -> Preprocessor, add `PLATFORM_WINDOWS` and `OPENEXR_DLL` to the Preprocessor

Definitions.

7. IexTest

requires addition of the following folder to 'Properties -> C/C++ -> Additional Include Directories':

```
..\..\..\..\config.windows
```

8. IlmThread

should be okay

9. Imath

requires addition of the following folder to 'Properties -> C/C++ -> Additional Include Directories':

```
..\..\..\..\config.windows
```

also, we need to comment out ALL 'IMATH_EXPORT" inside ImathVec.cpp:

9. ImathTest

requires addition of the following folder to 'Properties -> C/C++ -> Additional Include Directories':

```
..\..\..\..\config.windows
```

Via Properties -> C/C++ -> Preprocessor, add `__INTEL_COMPILER` to the Preprocessor Definitions.

We have to add the"`/Qrestrict`" flag to projects `Imath` and `ImathTest` instruct Intel compiler to acknowledge the use of 'restrict' keyword. "`/Qrestrict`" is added via "C++ -> Command line -> additional options"

[ilmBase ported OKAY 22/1/2014]

OpenEXR library

1. Create vc10 solution

Convert the included vc8 solution in our VS2010 (vc10)

2. Use Intel C++ compiler

Converted to icc environment inside visual studio using "Use Intel C++"

3. Prepare zlib (1.2.7)

`zlib.h` and `zconf.h` should be copied to `Deploy/include`
`zlib.lib` should be copied to `Deploy/lib/x64/Release`

Necessary to modify the 'input' of Linker to the IlmImf project:

```
..\..\..\..\..\..\Deploy\lib\$(IntDir)\zlibwapi.lib ->
```

..\..\..\..\..\..\Deploy\lib\$(IntDir)\zlib.lib

4. Intel C++ compiler libraries

Necessary to copy the following Intel compiler libraries (C:\Program Files (x86)\Intel\Composer XE 2011 SP1\compiler\lib\intel64) to

C:_OpenSource\source\OpenEXR_V2.1_vc9_icc\openexr_2.1.0\openexr-2.1.0\vc\vc8\OpenEXR\x64\Release:

- libdecimal.lib
- libirc.lib
- libmmd.lib
- svml_dispmd.lib

5. [new] Missing halfExport.h in Deploy

copy halfExport.h to Deploy\include

6. missing .cpp files for the IlmImf project:

ImfCompositeDeepScanLine.cpp

ImfDeepCompositing.cpp
ImfDeepFramebuffer.cpp
ImfDeepImageStateAttribute.cpp

ImfDeepScanLineInputFile.cpp
ImfDeepScanLineInputPart.cpp
ImfDeepScanLineOutputFile.cpp
ImfDeepScanLineOutputPart.cpp

ImfDeepTiledInputFile.cpp
ImfDeepTiledInputPart.cpp
ImfDeepTiledOutputFile.cpp
ImfDeepTiledOutputPart.cpp

ImfGenericInputFile.cpp
ImfGenericOutputFile.cpp

ImfInput*.cpp
ImfInputPartData.cpp

ImfMultiPartInputFile.cpp
ImfMultiPartOutputFile.cpp

ImfOutput*.cpp
ImfOutputPartData.cpp

ImfPartType.cpp
ImfTiled*.cpp

7. IlmImfTest project has MANY MISSING *.cpp:

```
	testCompositeDeepScanLine.cpp
	testDeep*.cpp
	testInputPart.cpp
	testMultiPart*.cpp
	testMultiScanlinePartThreading.cpp
	testMultiTiledPartThreading.cpp

	testBackwardCompatibility.cpp
	testCopyDeep*.cpp
	testCopyMultiPartFile.cpp

2.1.0:
	testOptimized.cpp
	testOptimizedInterleavePatterns.cpp
	testPartHelper.cpp
	testBadTypeAttributes.cpp
	testFutureProofing.cpp
```

8. IlmImfTest project requires Pre-processor definition of 'WIN32':

Via Properties -> C/C++ -> Preprocessor, add `WIN32` to the Preprocessor Definitions.