

UEFI & EDK II Training

Platform Build Lab Up Xtreme - Windows

tianocore.org

Copy and Paste see [Lab Guide.md](#)

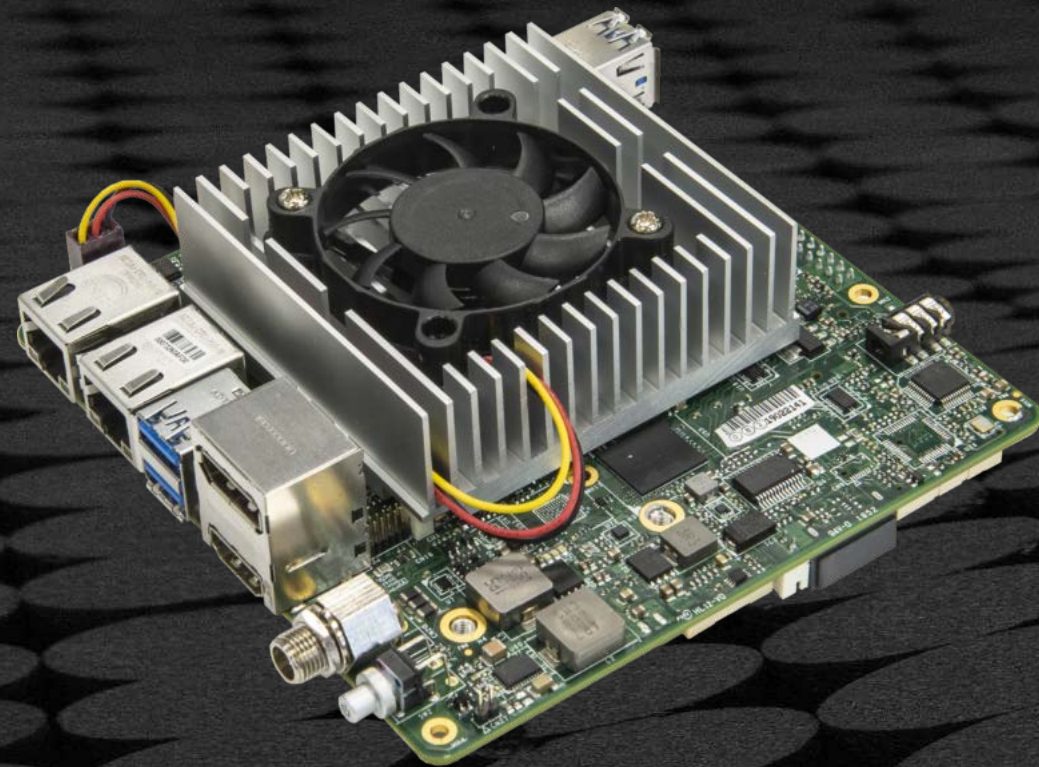
PLATFORM BUILD LABS

- ✿ Download Minplatform Using Git Bash
- ✿ Build a EDK II Platform using Up Xtreme Aaeon board

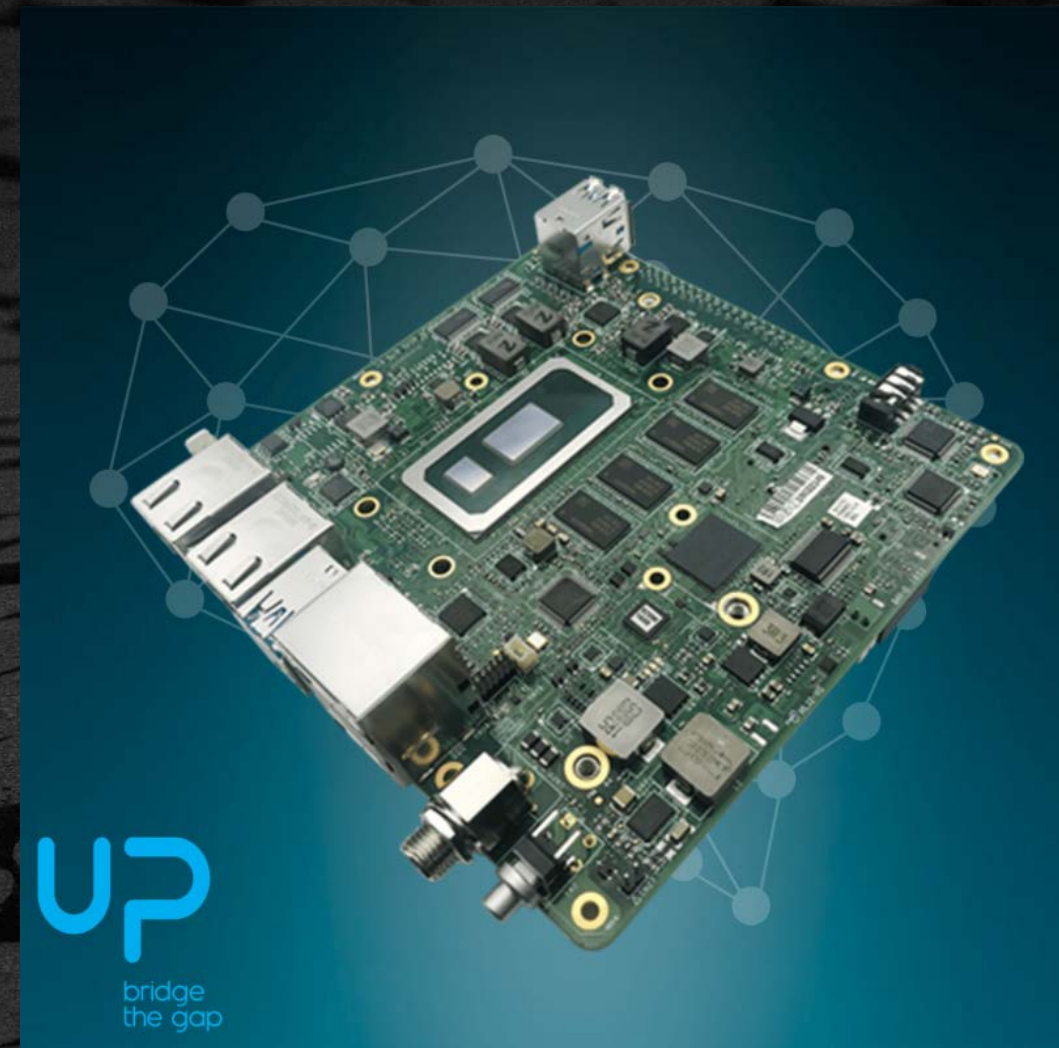
DOWNLOAD MINPLATFORM

Use Git Bash to download EDK II and MinPlatform

EDK II Platform – Up Xtreme by Aaeon



8th Generation Intel® Core™
U-Series processors
(Formerly Whiskey Lake)



[UP Board products](#)
[Up Shop](#)

Git Bash



Open “Git Bash”

Linux like commands “/” for dirs.

Use “/c” to go to C: in Windows, etc.

Cd to the Workspace:

```
$ cd /c/fw
```

```
$ mkdir UpX
```

```
$ help
```

```
$ cd UpX
```

```
MINGW64:/c/fw/UpX

ljarlstr@ljarlstr-MOBL MINGW64 ~
$ cd /c/fw

ljarlstr@ljarlstr-MOBL MINGW64 /c/fw
$ mkdir UpX

ljarlstr@ljarlstr-MOBL MINGW64 /c/fw
$ cd UpX

ljarlstr@ljarlstr-MOBL MINGW64 /c/fw/UpX
$ ls

ljarlstr@ljarlstr-MOBL MINGW64 /c/fw/UpX
$ help
GNU bash, version 4.4.19(2)-release (x86_64-pc-msys)
These shell commands are defined internally. Type 'help' to see this list.
Type 'help name' to find out more about the function 'name'.
Use 'info bash' to find out more about the shell in general.
Use 'man -k' or 'info' to find out more about commands not in this list.

A star (*) next to a name means that the command is disabled.

job_spec [&]                                history [-c] [-d offset] [n] or hist>
```


Download the source for Edk II, MinPlatform and FSP

In the Git Bash command line window Do the following:

- Edk2

```
$ git clone --recursive https://github.com/tianocore/edk2
```

- Edk2-platforms

```
$ git clone https://github.com/tianocore/edk2-platforms.git
```

- Edk2-non-os

```
$ git clone https://github.com/tianocore/edk2-non-os.git
```

- FSP

```
$ git clone https://github.com/IntelFsp/FSP.git
```

Set PROXYS FIRST

```
$ git config --global https.proxy=proxy.hf.intel.com:911
```

```
$ git config --global http.proxy=proxy.hf.intel.com:911
```



Takes
about 6
minutes

Download MinPlatform Lab Material

Download the PlatformBuildLab_MinPlatform_FW.zip from :  [github.com](https://github.com/tianocore-training/PlatformBuildLab2_FW.zip)
[PlatformBuildLab2_FW.zip](https://github.com/tianocore-training/PlatformBuildLab2_FW.zip)

OR

Use `git clone` to download the PlatformBuildLab_MinPlatform_FW

```
C:/> git clone https://github.com/tianocore-training/PlatformBuildLab_MinPlatform_FW.git
```

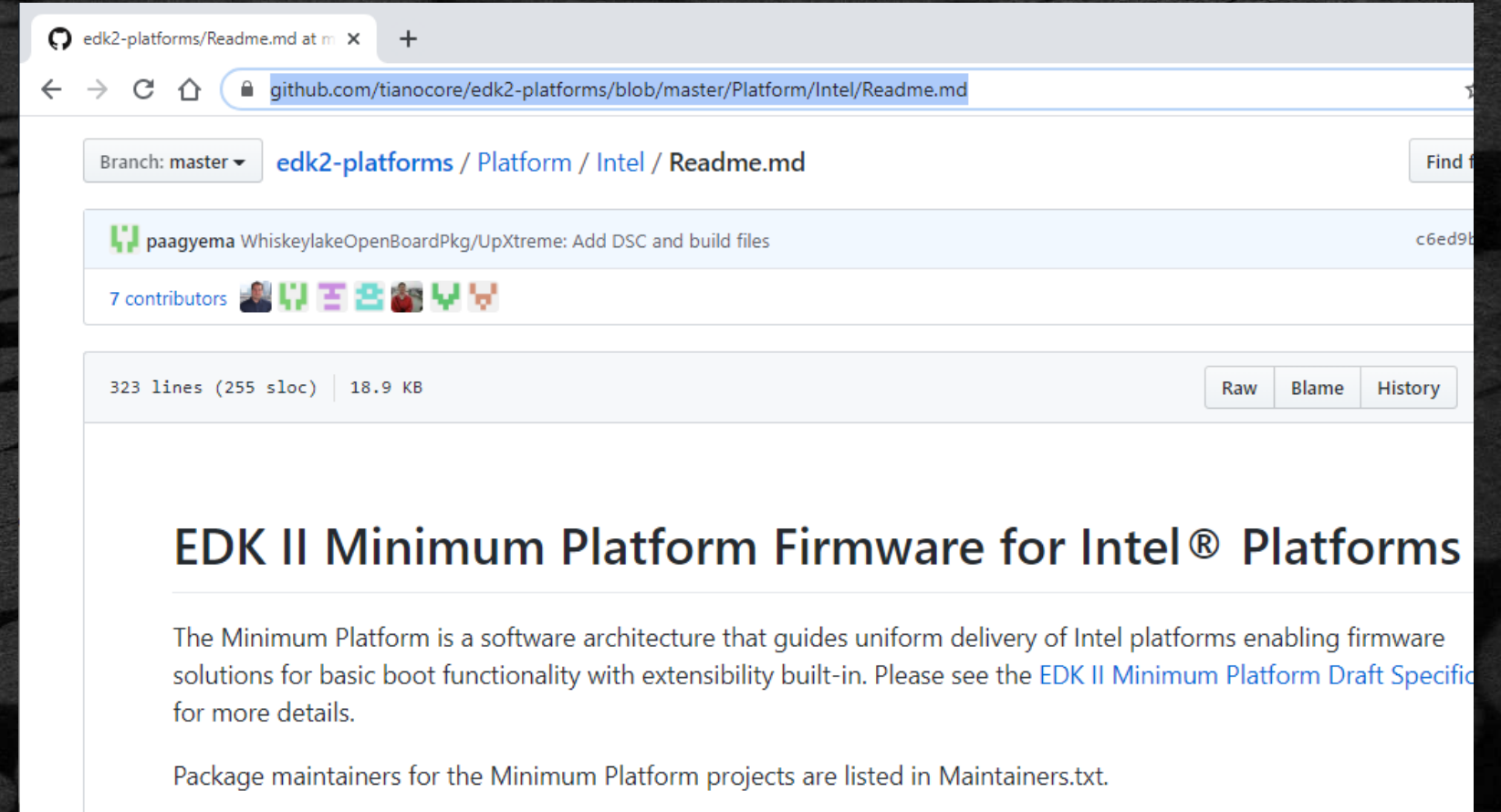
Directory PlatformBuildLab_MinPlatform_FW will be created

```
/FW
/MinPlatformBuild
- asl                - Asl Compiler                - Readme has download info
- FTDI-Driver        - Serial / USB cable         - Readme has download info
- UpX_Lab            - Lab Material and Lab Guide
- TeraTerm           - Terminal app               - Readme has download info
- Nasm               - Nasm Assembler            - Readme has download info
```

BUILD UP XTREME

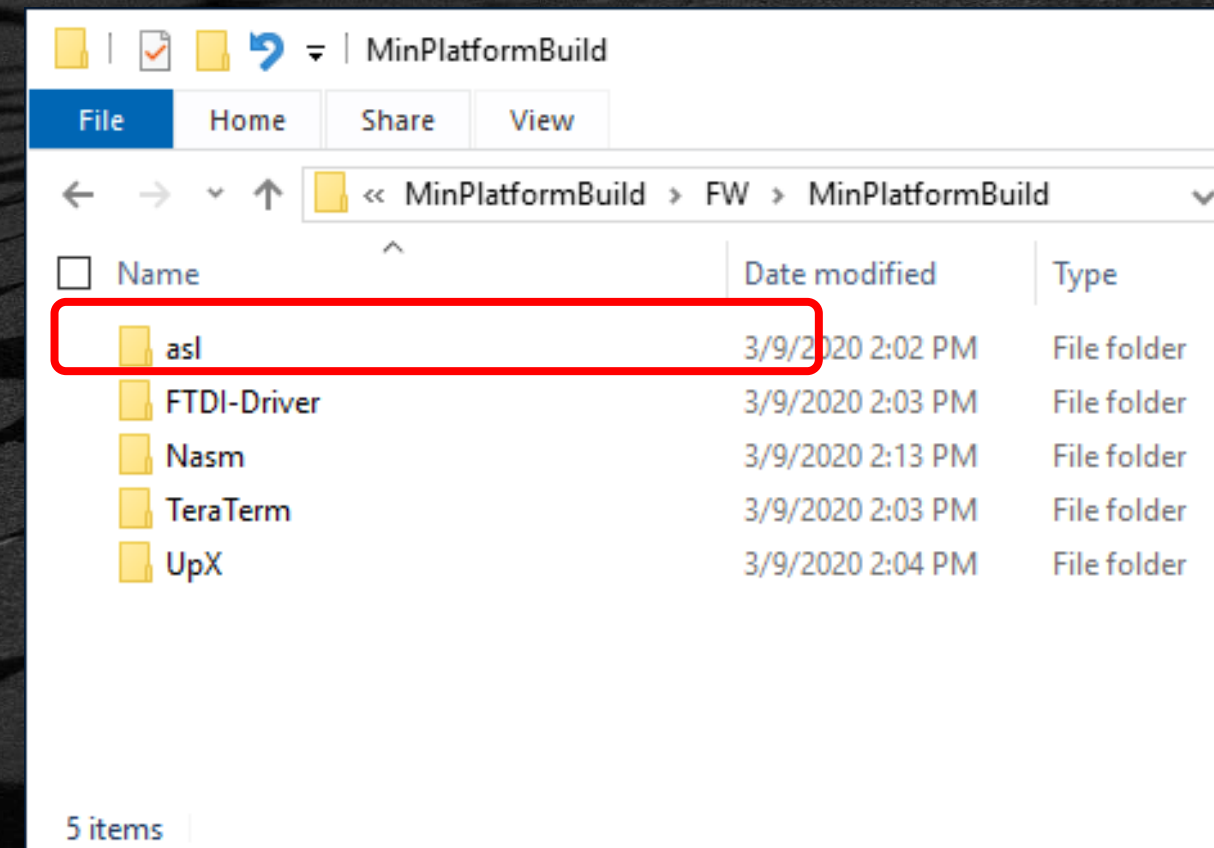
Where to get Open Source Up Xtreme

How to Download & Build: Open Source MinPlatform [Readme.md](#)



Directory

C:\MinPlatformBuildLab_FW\FW\MinPlatformBuildLab
from Download or zip

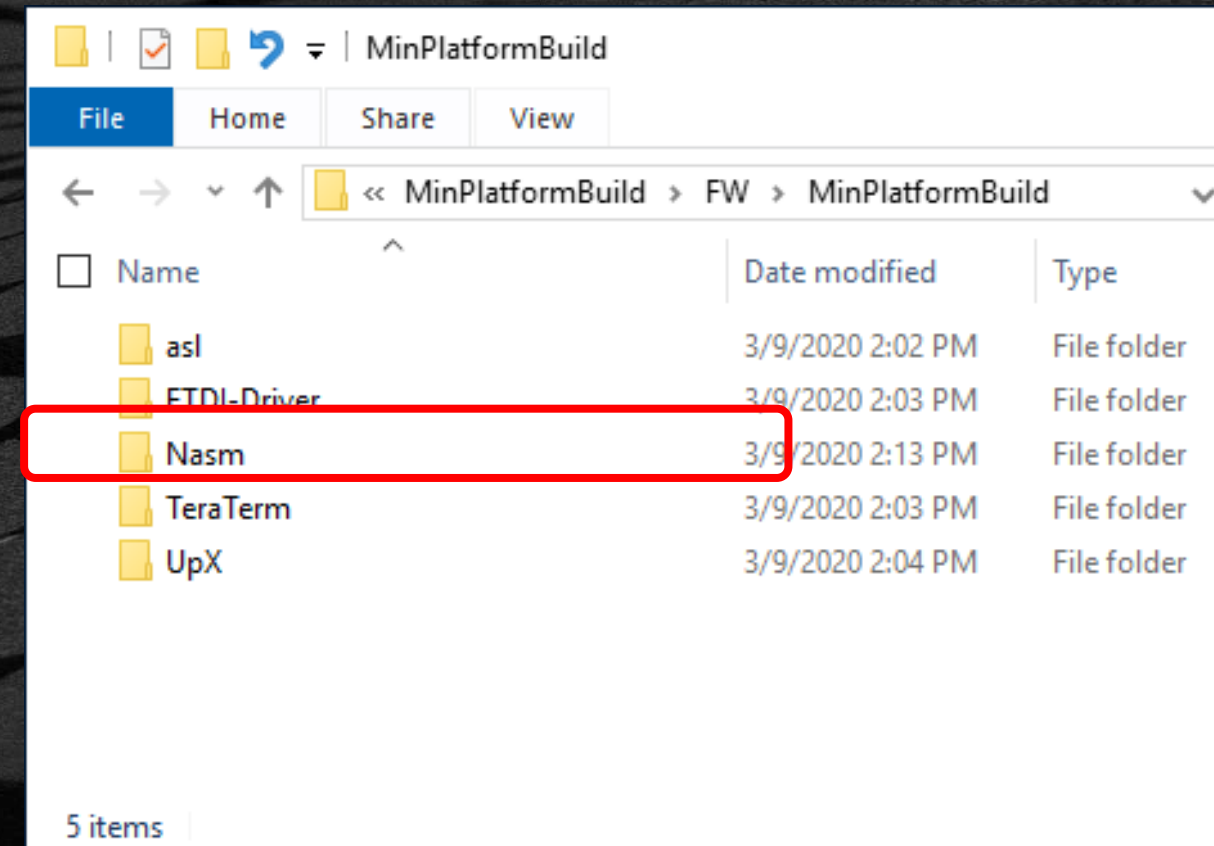


1 Copy \asl Folder to C:\

Note: Download Asl compiler described in the Readme.txt

Directory

C:\MinPlatformBuildLab_FW\FW\MinPlatformBuildLab
from Download or zip



2 Copy \Nasm Folder to C:\

Note: Download Nasm compiler described in the Readme.txt

MinPlatform Open Board Tree Structure

edk2 <https://github.com/tianocore/edk2>

edk2-platforms <https://github.com/tianocore/edk2-platforms>

Platform/

Intel/

BoardModulePkg

WhiskeylakeOpenBoardPkg

UpXtreme

MinPlatformPkg

Silicon/

Intel/

CoffeelakeSiliconPkg

. . .

Features/Intel

AdvancedFeaturePkg

edk2-non-osi <https://github.com/tianocore/edk2-non-osi>

Silicon/

Intel/

CoffeelakeSiliconBinPkg

FSP <https://github.com/IntelFsp/FSP>

CoffeelakeFspBinPkg

Invoke the Build .py from here

Platform DSC & FDF here

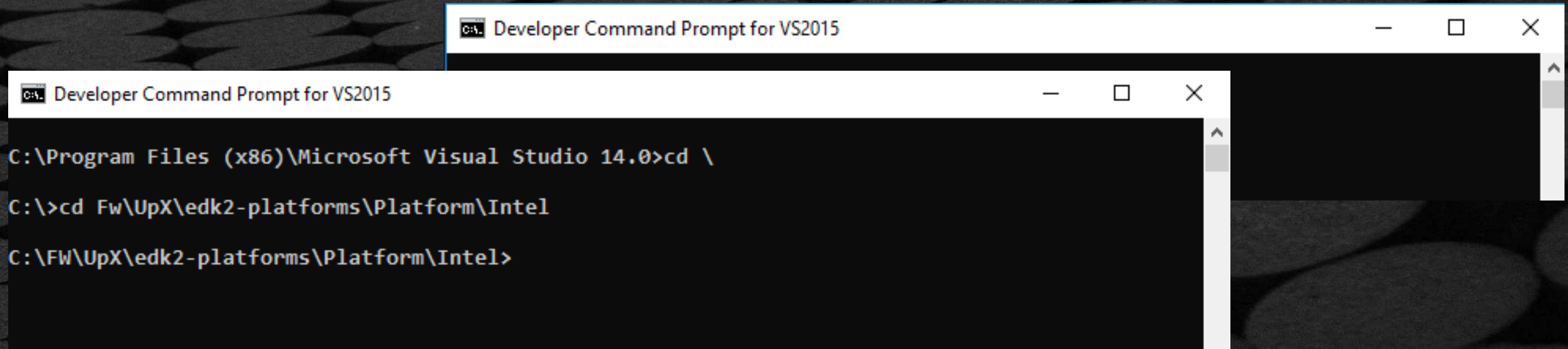
Open a VS Command Prompt

Follow Steps from [here](#) to Pin the Visual Studio Command Prompt to the Windows Task Bar

Open a Visual Studio Command Prompt &

```
> cd C:\FW\UpX\edk2-platforms\Platform\Intel
```

3



```
C:\Program Files (x86)\Microsoft Visual Studio 14.0>cd \  
C:\>cd Fw\UpX\edk2-platforms\Platform\Intel  
C:\FW\UpX\edk2-platforms\Platform\Intel>
```


Check if Python okay (may also need to set PYTHON_HOME)

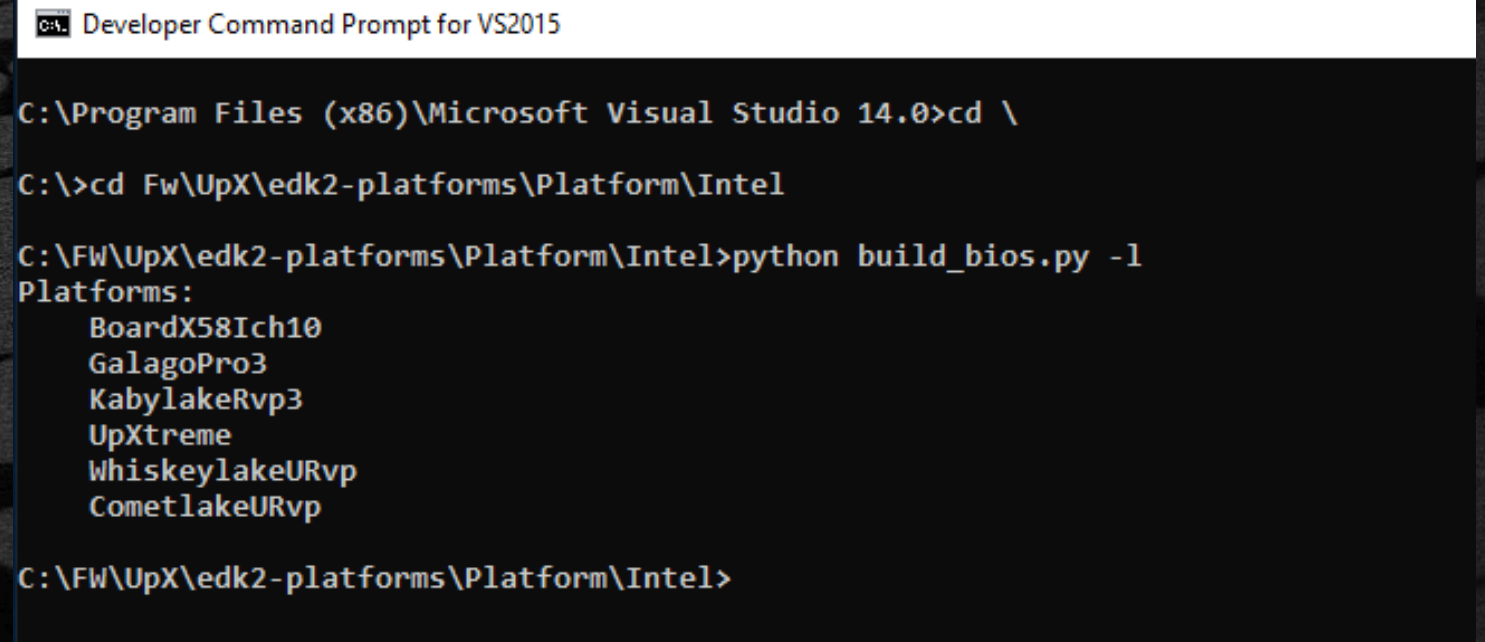
```
$> python --version  
Python 3.8.2
```

Check for PYTHON_HOME Variable and set if not declared (note Python v 3.8.n)

```
$> set PYTHON_HOME=%USERPROFILE%\AppData\Local\Programs\Python\Python38-32
```

Check for available MinPlatform Boards

```
$> python build_bios.py -l
```



```
Developer Command Prompt for VS2015  
C:\Program Files (x86)\Microsoft Visual Studio 14.0>cd \  
C:\>cd Fw\UpX\edk2-platforms\Platform\Intel  
C:\FW\UpX\edk2-platforms\Platform\Intel>python build_bios.py -l  
Platforms:  
BoardX58Ich10  
GalagoPro3  
KabylakeRvp3  
UpXtreme  
WhiskeylakeURvp  
CometlakeURvp  
C:\FW\UpX\edk2-platforms\Platform\Intel>
```


Invoke the Build

4

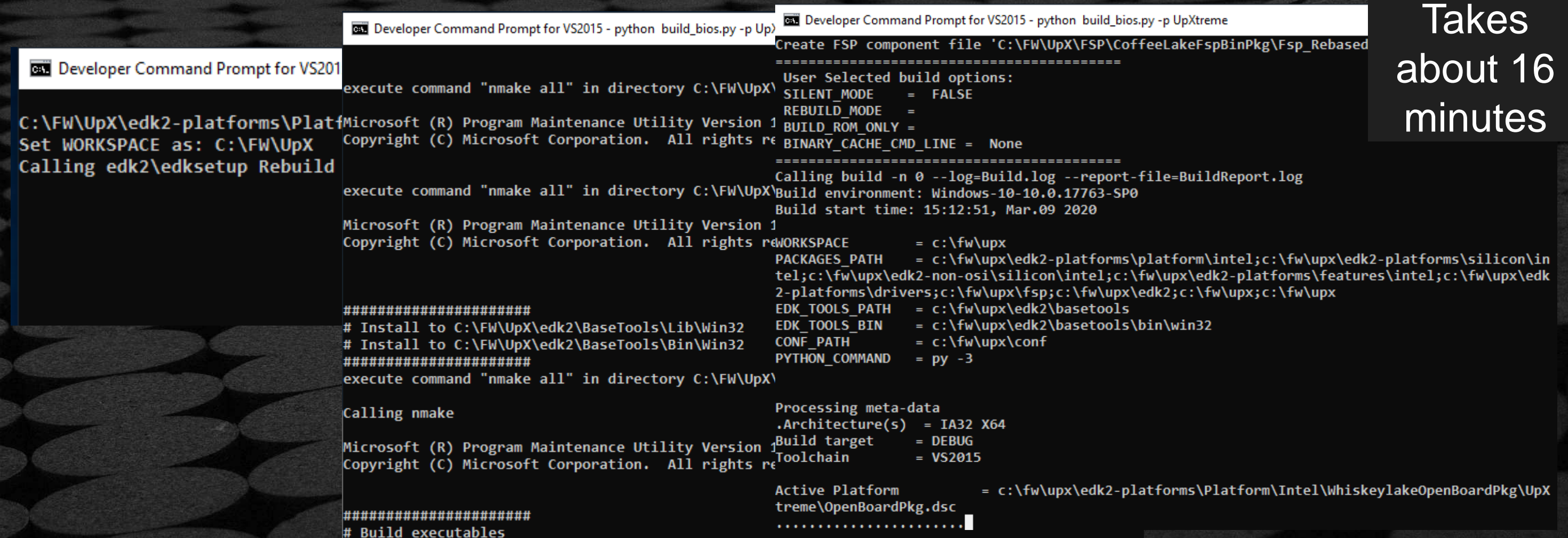
Invoke the Python Build script for Up Xtreme

```
$> python build_bios.py -p UpXtreme -t VS20XX
```

Where XX is 15 or 17 or 19



Takes
about 16
minutes



```

C:\FW\UpX\edk2-platforms\Platform\Intel\WhiskeylakeOpenBoardPkg> python build_bios.py -p UpXtreme -t VS2015
Set WORKSPACE as: C:\FW\UpX
Calling edk2\edksetup Rebuild
execute command "nmake all" in directory C:\FW\UpX\
Microsoft (R) Program Maintenance Utility Version 1.0.0
Copyright (C) Microsoft Corporation. All rights reserved.

#####
# Install to C:\FW\UpX\edk2\BaseTools\Lib\Win32
# Install to C:\FW\UpX\edk2\BaseTools\Bin\Win32
#####
execute command "nmake all" in directory C:\FW\UpX\
Calling nmake
Microsoft (R) Program Maintenance Utility Version 1.0.0
Copyright (C) Microsoft Corporation. All rights reserved.

#####
# Build executables
#####

Create FSP component file 'C:\FW\UpX\FSP\CoffeeLakeFspBinPkg\Fsp_Rebased
=====
User Selected build options:
SILENT_MODE      = FALSE
REBUILD_MODE     =
BUILD_ROM_ONLY   =
BINARY_CACHE_CMD_LINE = None
=====
Calling build -n 0 --log=Build.log --report-file=BuildReport.log
Build environment: Windows-10-10.0.17763-SP0
Build start time: 15:12:51, Mar.09 2020
WORKSPACE       = c:\fw\upx
PACKAGES_PATH   = c:\fw\upx\edk2-platforms\platform\intel;c:\fw\upx\edk2-platforms\silicon\intel;c:\fw\upx\edk2-non-osi\silicon\intel;c:\fw\upx\edk2-platforms\features\intel;c:\fw\upx\edk2-platforms\drivers;c:\fw\upx\fsp;c:\fw\upx\edk2;c:\fw\upx;c:\fw\upx
EDK_TOOLS_PATH  = c:\fw\upx\edk2\basetools
EDK_TOOLS_BIN   = c:\fw\upx\edk2\basetools\bin\win32
CONF_PATH       = c:\fw\upx\conf
PYTHON_COMMAND  = py -3

Processing meta-data
Architecture(s) = IA32 X64
Build target     = DEBUG
Toolchain        = VS2015

Active Platform  = c:\fw\upx\edk2-platforms\Platform\Intel\WhiskeylakeOpenBoardPkg\UpXtreme\OpenBoardPkg.dsc
.....
  
```


Platform Config

Many Platforms have a bash, bat or Python script file to pre or post process the EDK II build process

For MinPlatform platform specific config

Build processing:

Build_config.cfg – Lists directories required for the build and build settings

Link to Up Xtreme [Build_config.cfg](#)

Examine Build Parameters

```
Python build_bios.py -p UpXtreme
```

...

```
Calling build -n 0 --log=Build.log --report-file=BuildReport.log  
and from UpX\conf\target.txt
```

| | |
|--|---|
| TARGET | = DEBUG |
| TARGET_ARCH | = IA32 X64 |
| TOOL_CHAIN_TAG | = VS2015 |
| ACTIVE_PLATFORM | = ... \\WhiskylakeOpenBoardPkg\ UpXtreme\OpenBoardPkg.dsc |
| Report file created (via python script) | = BuildReport.log |

Build Mode

CPU Architecture

VS Tool Chain

Platform DSC file

PCDs, Libs, etc.

Platform Build and PCD Parameters

Platform Parameters

Many Platform Parameters are defined in a top .DSC file that controls PCD and build switches

For Up Xtreme : edk2-platforms\Platform\Intel\WhiskeylakeOpenBoardPkg\UpXtremeOpenBoardPkgPcd.dsc and OpenBoardPkgBuildOption.dsc

Example:

```
# Define Build Options both for EDK and EDKII drivers.
```

```
DEFINE DSC_S3_BUILD_OPTIONS =  
DEFINE DSC_CSM_BUILD_OPTIONS =
```

```
!if gSiPkgTokenSpaceGuid.PcdAcpiEnable == TRUE  
  DEFINE DSC_ACPI_BUILD_OPTIONS = -DACPI_SUPPORT=1  
!else  
  DEFINE DSC_ACPI_BUILD_OPTIONS =  
!endif
```

```
DEFINE BIOS_GUARD_BUILD_OPTIONS =  
DEFINE OVERCLOCKING_BUILD_OPTION =
```


Build Process for RELEASE Target

Invoke the Python Build script for Up Xtreme

```
$> python build_bios.py -p UpXtreme -r -t VS20XX
```



Takes
about 16
minutes

```

C:\FW\UpX\edk2-platforms\Platform
Set WORKSPACE as: C:\FW\UpX
Calling edk2\edksetup Rebuild

Developer Command Prompt for VS2015 - python build_bios.py
Calling nmake
Microsoft (R) Program Maintenance Utility Vers
Copyright (C) Microsoft Corporation. All right

#####
# Build executables
#####
Building FitGen

Microsoft (R) Program Maintenance Utility Vers
Copyright (C) Microsoft Corporation. All right

FitGen built successfully (all)

=====
BIOS_SIZE_OPTION = -DBIOS_SIZE_OPTION=SIZE
EFI_SOURCE        = edk2
TARGET            = RELEASE
TARGET_ARCH       = IA32 X64
TOOL_CHAIN_TAG    = VS2015
WORKSPACE         = C:\FW\UpX
WORKSPACE_CORE    = edk2
EXT_BUILD_FLAGS  =

Calling C:\Python37-32\python C:\FW\UpX\edk2-plat
\RebaseFspBinBaseAddress.py C:\FW\UpX\edk2-plat
Xtreme\Include\Fdf\FlashMapInclude.fdf C:\FW\UpX\

Developer Command Prompt for VS2015 - python build_bios.py -p UpXtreme -r
Create FSP component file 'C:\FW\UpX\FSP\CoffeeLakeFspBinPkg\Fsp_Rebase
=====
User Selected build options:
SILENT_MODE      = FALSE
REBUILD_MODE     =
BUILD_ROM_ONLY   =
BINARY_CACHE_CMD_LINE = None
=====
Calling build -n 0 --log=Build.log --report-file=BuildReport.log
Build environment: Windows-10-10.0.17763-SP0
Build start time: 15:35:03, Mar.09 2020

Workspace          = c:\fw\upx
Packages_Path      = c:\fw\upx\edk2-platforms\platform\intel;c:\fw\upx\edk2-platforms\silicon\in
tel;c:\fw\upx\edk2-non-osi\silicon\intel;c:\fw\upx\edk2-platforms\features\intel;c:\fw\upx\edk
2-platforms\drivers;c:\fw\upx\fsp;c:\fw\upx\edk2;c:\fw\upx;c:\fw\upx
Edk_Tools_Path     = c:\fw\upx\edk2\basetools
Edk_Tools_Bin      = c:\fw\upx\edk2\basetools\bin\win32
Conf_Path          = c:\fw\upx\conf
Python_Command     = py -3

Processing meta-data .
Architecture(s)   = IA32 X64
Build target      = RELEASE
Toolchain         = VS2015

Active Platform    = c:\fw\upx\edk2-platforms\Platform\Intel\WhiskeylakeOpenBoardPkg\UpX

```


DEBUG & RELEASE Differences

Slower boot because the time it takes to display debug info

Larger image because of debug code & embedded info

Uses the serial port for debug string output

Contains detailed debug strings that show the boot process and various ASSERT/TRACE errors

Directory C:\MinPlatformBuildLab_FW\FW\MinPlatformBuildLab\UpX_Lab

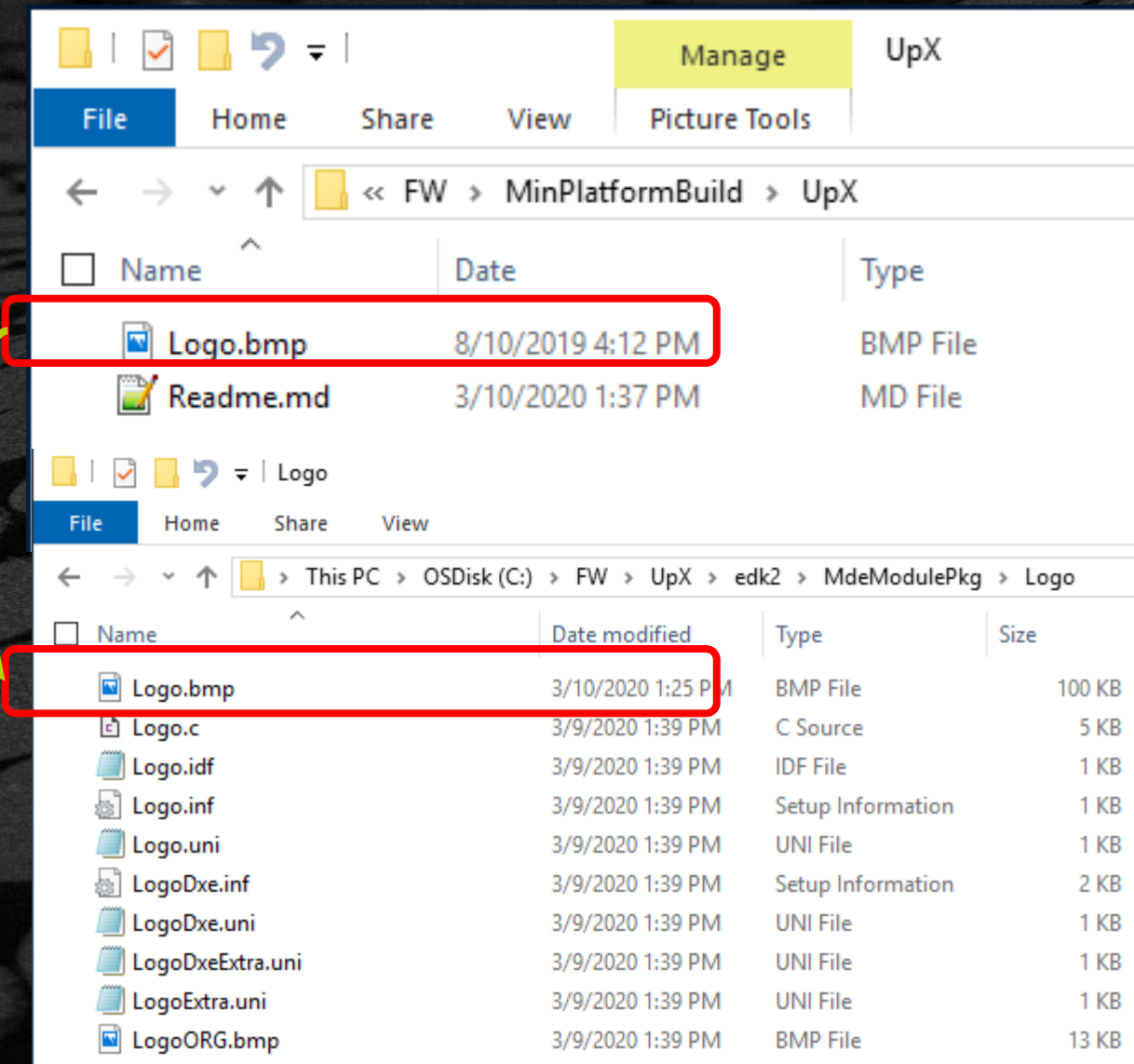
Copy Logo.bmp to
C:\FW\UpX\edk2\MdeModulePkg\Logo

Or create a .BMP with Windows Paint



See . . .

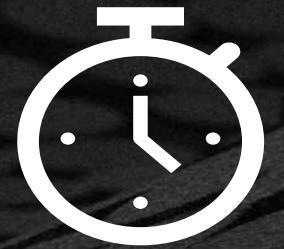
WhiskeylakeOpenBoardPkg\UpXtreme\OpenBoardPkg.fdf line 285



Build with new logo

Invoke the Python Build script for Up Xtreme

```
$> python build_bios.py -p UpXtreme -t VS20XX
```



Takes
about 2
minutes

```
Developer Command Prompt for VS2015 - python build_bios.py -p UpXtreme

C:\FW\UpX\edk2-platforms\Platform\Intel>python build_bios.py -p UpXtreme
Set WORKSPACE as: C:\FW\UpX
Calling edk2\edksetup Rebuild
```


5

Locate the build .fd images

```

C:\> Developer Command Prompt for VS2015

Microcode[0] - (0xffe50060, 0x00018000, 0x0100)
Microcode[1] - (0xffe68060, 0x00018800, 0x0100)
Microcode[2] - (0xffe80860, 0x00018800, 0x0100)

#####
# FIT Table: #
#####
FIT Pointer Offset: 0x40
FIT Table Address: 0xfffffb300
=====
=====)
Index:      Address      Size  Version      Type      C_V  Checksum (Index  Data Width  Bit
Offset)
=====
=====)
00:  2020205f5449465f 000004  0100  00-'_FIT_'  01    1c
01:  00000000ffe50060 000000  0100  01-MICROCODE 00    00
02:  00000000ffe68060 000000  0100  01-MICROCODE 00    00
03:  00000000ffe80860 000000  0100  01-MICROCODE 00    00
=====
=====)
Index:      Address      Size  Version      Type      C_V  Checksum (Index  Data Width  Bit
Offset)
=====
=====)
Done
Fd file can be found at C:\FW\UpX\Build\WhiskeylakeOpenBoardPkg\UpXtreme\RELEASE_VS2015\FV\UPX
TREME.fd
C:\FW\UpX\edk2-platforms\Platform\Intel>

```

The script displays the location of the final .fd files

SUMMARY

- ✿ Download Minplatform Using Git Bash
- ✿ Build a EDK II Platform using Up Xtreme Aaeon board

Questions?



Return to Main Training Page



Return to Training Table of contents for next presentation

[Link](#)



Redistribution and use in source (original document form) and 'compiled' forms (converted to PDF, epub, HTML and other formats) with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code (original document form) must retain the above copyright notice, this list of conditions and the following disclaimer as the first lines of this file unmodified.

Redistributions in compiled form (transformed to other DTDs, converted to PDF, epub, HTML and other formats) must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.


THIS DOCUMENTATION IS PROVIDED BY TIANOCORE PROJECT "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL TIANOCORE PROJECT BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS DOCUMENTATION, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright (c) 2021, Intel Corporation. All rights reserved.

BACKUP

BUILD ERRORS

Error message:



```
Developer Command Prompt for VS2015
EBUG_VS2015x86\X64\ShellPkg\DynamicCommand\TftpDynamicCommand\TftpDynamicCommand\OUTPUT\tftpDynamicCommandhii.rc
'c:\Program' is not recognized as an internal or external command,
operable program or batch file.
"C:\Program Files (x86)\Microsoft Visual Studio 14.0\VC\bin\x86_amd64\link.exe" /OUT:c:\fw\edk2-ws\Build\EmulatorX64
\DEBUG_VS2015x86\X64\MdeModulePkg\Universal\Disk\PartitionDxe\PartitionDxe\DEBUG\PartitionDxe.dll /NOLOGO /NODEFAULTLIB /IGN
ORE:4001 /OPT:REF /OPT:ICF=10 /MAP /ALIGN:32 /SECTION:.xdata,D /SECTION:.pdata,D /Machine:X64 /LTCG /DLL /ENTRY:_ModuleEntry
Point /SUBSYSTEM:EFI_BOOT_SERVICE_DRIVER /SAFESEH:NO /BASE:0 /DRIVER /DEBUG /ALIGN:4096 /FILEALIGN:4096 /SUBSYSTEM:CONSOLE /
EXPORT:InitializeDriver=_ModuleEntryPoint /BASE:0x10000 @c:\fw\edk2-ws\Build\EmulatorX64\DEBUG_VS2015x86\X64\MdeModulePkg\
Universal\Disk\PartitionDxe\PartitionDxe\OUTPUT\static_library_files.lst
NMAKE : fatal error U1077: '"c:\Program Files (x86)\Windows Kits\8.1\bin\x64\rc.exe' : return code '0x1'
Stop.
'c:\Program' is not recognized as an internal or external command,
operable program or batch file.
NMAKE : fatal error U1077: '"c:\Program Files (x86)\Windows Kits\8.1\bin\x64\rc.exe' : return code '0x1'
```

Find where the RC.EXE is located on your VS Installation:

Example (VS 2015): The RC.exe is located on this machine:

C:\Program Files (x86)\Windows Kits\8.1\bin\x64

Edit Conf\tools_def.txt

Build Error- RC.exe Cont.

Edit `Conf\tools_def.txt`

Search for your installation of Visual Studio (2013, 2015, 2017) “RC.EXE”

Probably in path `C:\Program Files (x86)\Windows Kits\`

Update according to the path for where the RC.EXE is found

```
# Microsoft Visual Studio 2013 Professional Edition
DEFINE WINSDK8_BIN      = c:\Program Files\Windows Kits\8.1\bin\x86\
DEFINE WINSDK8x86_BIN   = c:\Program Files (x86)\Windows Kits\8.1\bin\x64

# Microsoft Visual Studio 2015 Professional Edition
DEFINE WINSDK81_BIN     = c:\Program Files\Windows Kits\8.1\bin\x86\
DEFINE WINSDK81x86_BIN  = c:\Program Files (x86)\Windows Kits\8.1\bin\x64

# Microsoft Visual Studio 2017 Professional Edition
DEFINE WINSDK10_BIN     = C:\Program Files (x86)\Windows Kits\10\bin\x86
```

Paths on your
machine



Build Error: fatal error C1041:

Build Error from fatal error C1041: cannot open program database

This Error is usually because the location you are building is being shared by another application in Windows. Example: Syncplicity may cause this

Error Message:

```
k:\fw\edk2\MdePkg\Library\BaseLib\LinkedList.c : fatal error C1041: cannot open program
database
'k:\fw\edk2\build\nt32ia32\debug_vs2013x86\ia32\mdepkg\library\baselib\baselib\vc120.pdb'; if
multiple CL.EXE write to the same .PDB file, please use /FS
NMAKE : fatal error U1077: '"C:\Program Files (x86)\Microsoft Visual Studio
12.0\Vc\bin\cl.exe"' : return code '0x2'
Stop.
```

Solution: Try using a Workspace that is not shared