tianocore

UEFI & EDK II Training

Platform Build Lab Up Xtreme - Windows

tianocore.org

Copy and Paste see Readme.md

tianocore

PLATFORM BUILD LABS

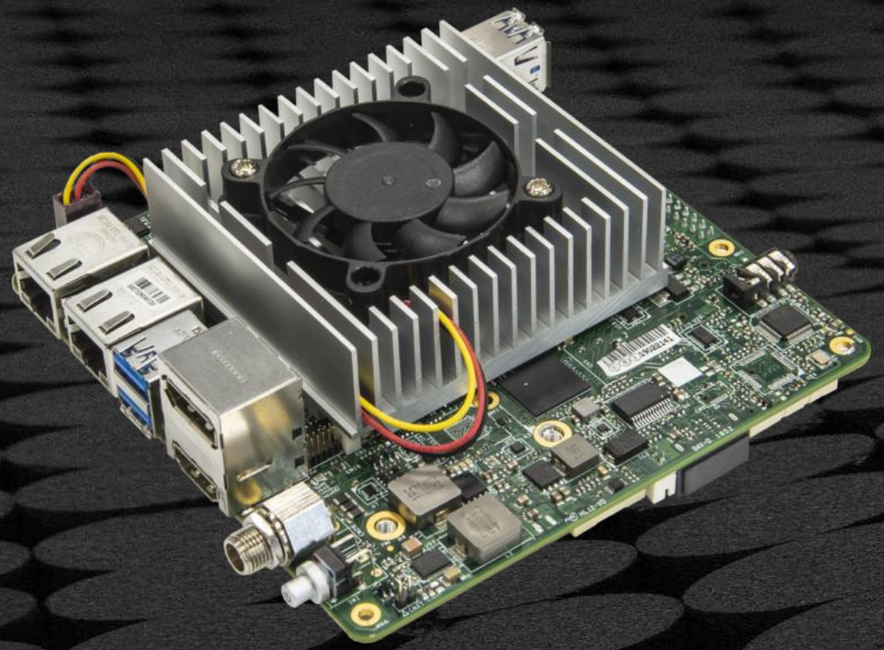
- Download Minplatform Using Git Bash
- Build a EDK II Platform using Up Xtreme Aaeon board

tianocore

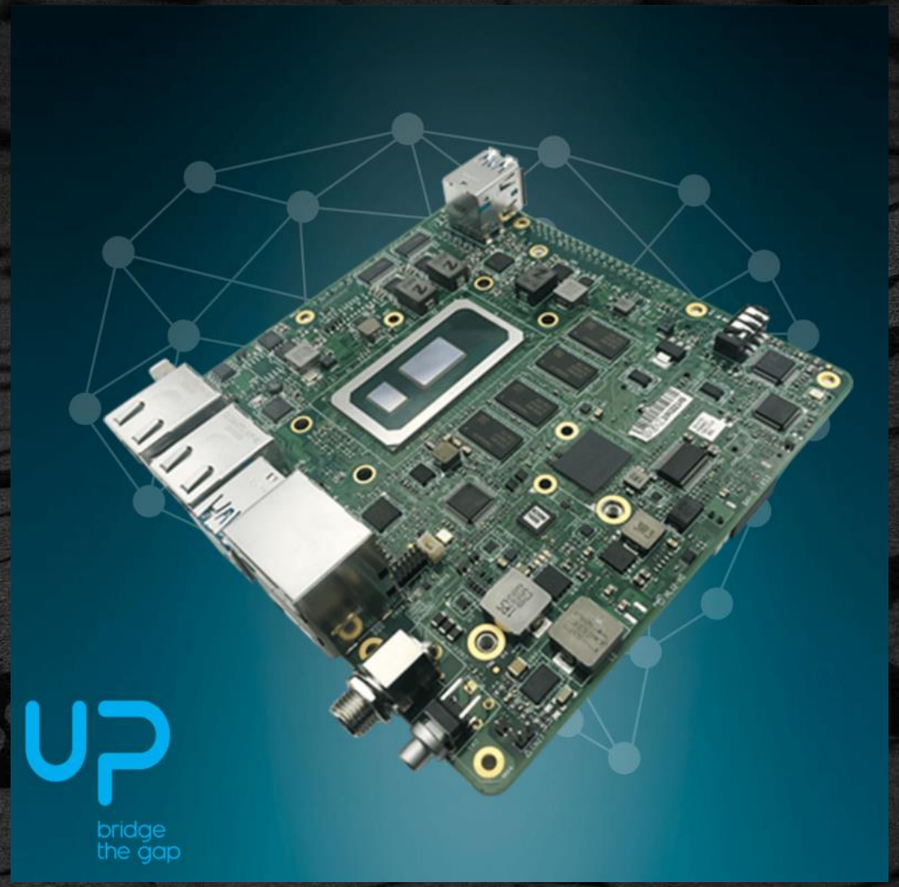
DOWNLOAD MINPLATFORM

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

tianocore

Git Bash

Open “Git Bash”

- Linux like commands “/” for dirs.
- Use “/c” to go to C: in Windows, etc.

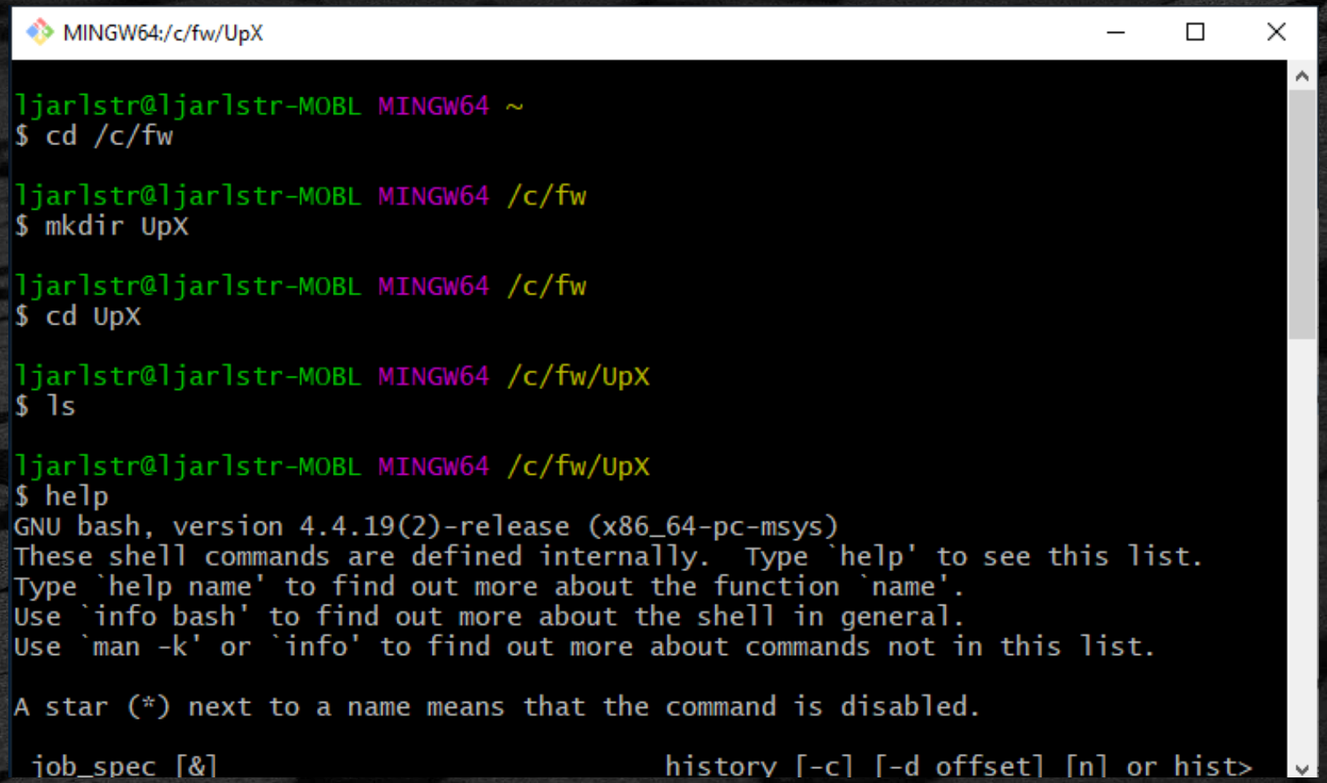
Cd to the Work Space:

```
$ cd /c/fw

$ mkdir UpX

$ help

$ cd UpX
```



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-osi

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

- FSP

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

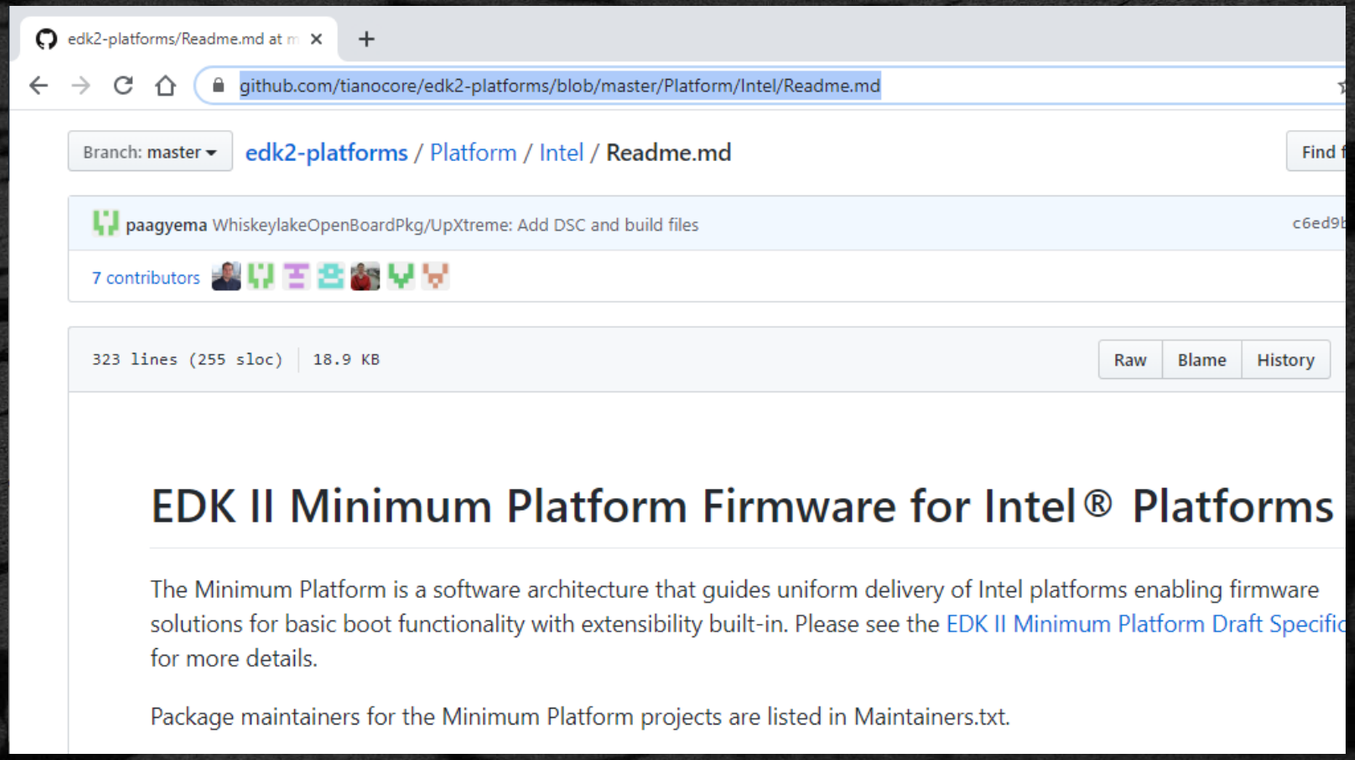
Takes about 6 minutes

tianocore

BUILD UP XTREME

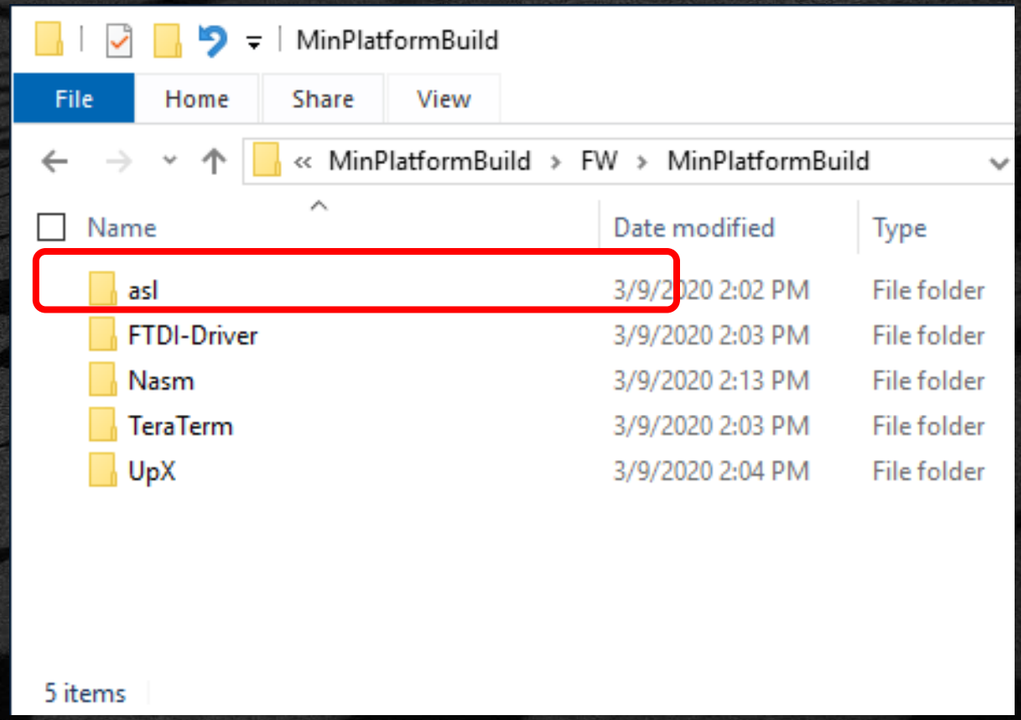
Where to get Open Source Up Xtreme

How to Download & Build: Open Source MinPlatform Readme.md



Preparing to Build

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

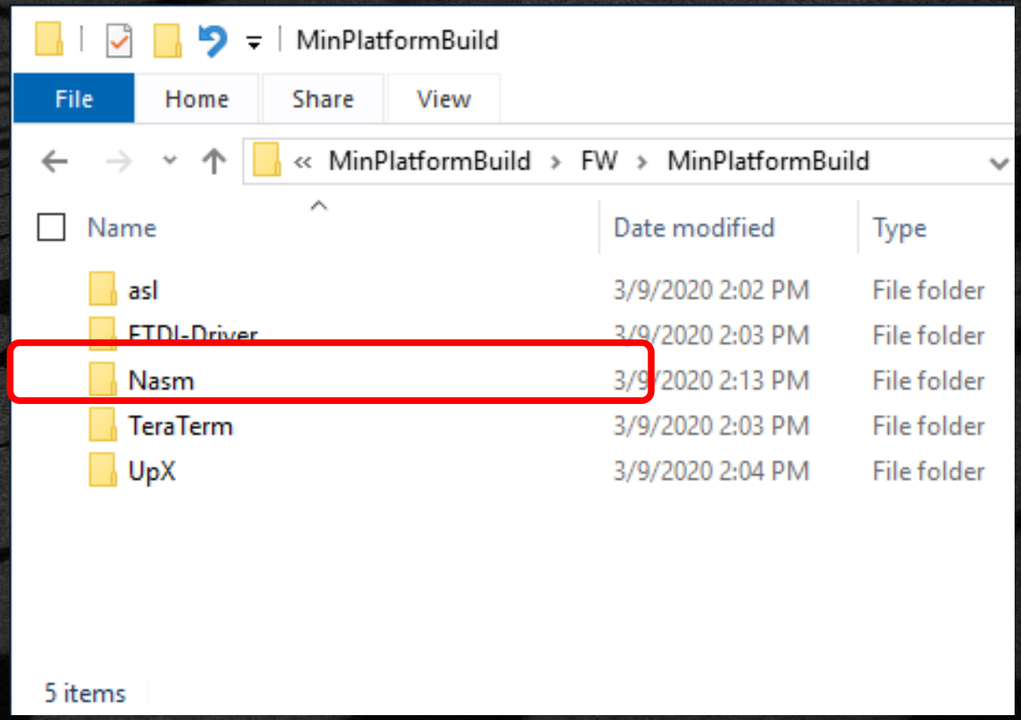


1. Copy \asl Folder to C:\

Note: Download Asl compiler described in the Readme.txt

Preparing to Build

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-platform   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

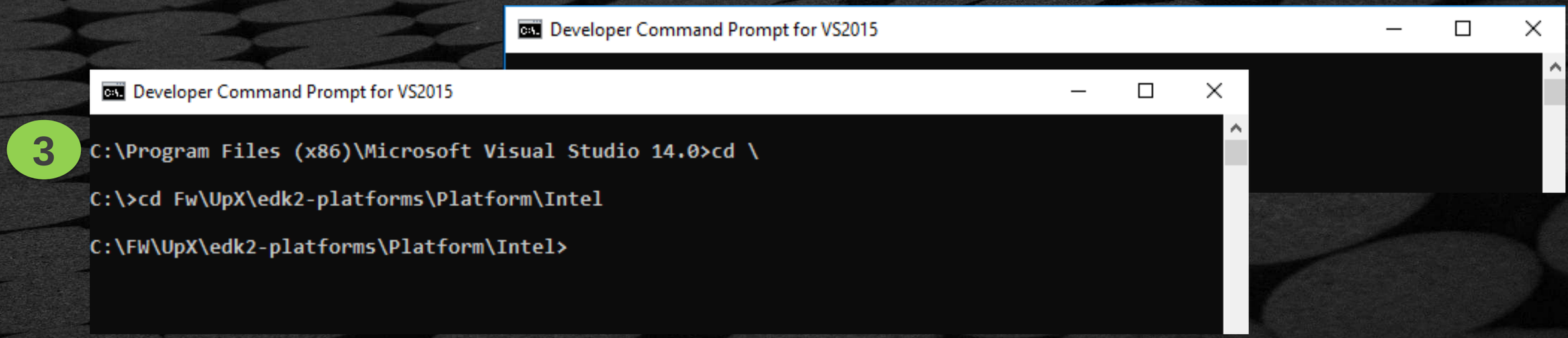
Platform DSC & FDF here

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-platform\Platform\Intel
```



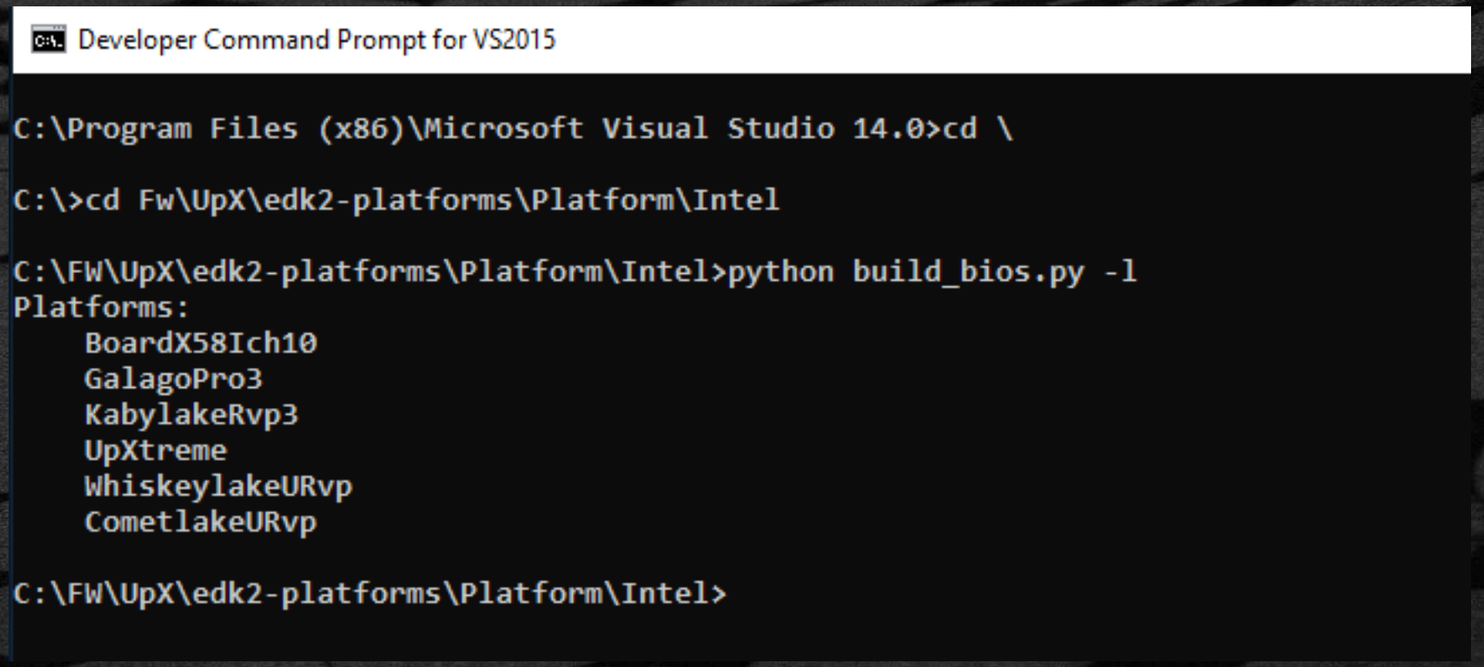
Build Environment

Check if Python okay

```
$>  python --version
Python 3.7.2
```

Check for available MinPlatform Boards

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



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

Takes about 16 minutes

```
Developer Command Prompt for VS201

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

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

execute command "nmake all" in directory C:\FW\UpX\
Microsoft (R) Program Maintenance Utility Version 1
Copyright (C) Microsoft Corporation.  All rights re

execute command "nmake all" in directory C:\FW\UpX\
Microsoft (R) Program Maintenance Utility Version 1
Copyright (C) Microsoft Corporation.  All rights re

######################
# 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
Copyright (C) Microsoft Corporation.  All rights re

######################
# Build executables
```

```
Developer Command Prompt for VS2015 - python  build_bios.py -p UpXtreme
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\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      = DEBUG
Toolchain         = VS2015

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

Platform Build Scripts

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	Build Mode
TARGET_ARCH	= IA32 X64	CPU Architecture
TOOL_CHAIN_TAG	= VS2015	VS Tool Chain
ACTIVE_PLATFORM	= ... \WhiskylakeOpenBoardPkg\ UpXtreme\OpenBoardPkg.dsc	Platform DSC file
Report file created (via python script)	= BuildReport.log	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\UpXtreme OpenBoardPkgPcd.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

4

Invoke the Python Build script for Up Xtreme

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

Takes about 16 minutes

```
Developer Command Prompt for VS2015 - p

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 righ


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

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

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-p
\RebaseFspBinBaseAddress.py C:\FW\UpX\edk2-pla
Xtreme\Include\Fdf\FlashMapInclude.fdf C:\FW\U
```

```
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
treme\OpenBoardPkg.dsc
```

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

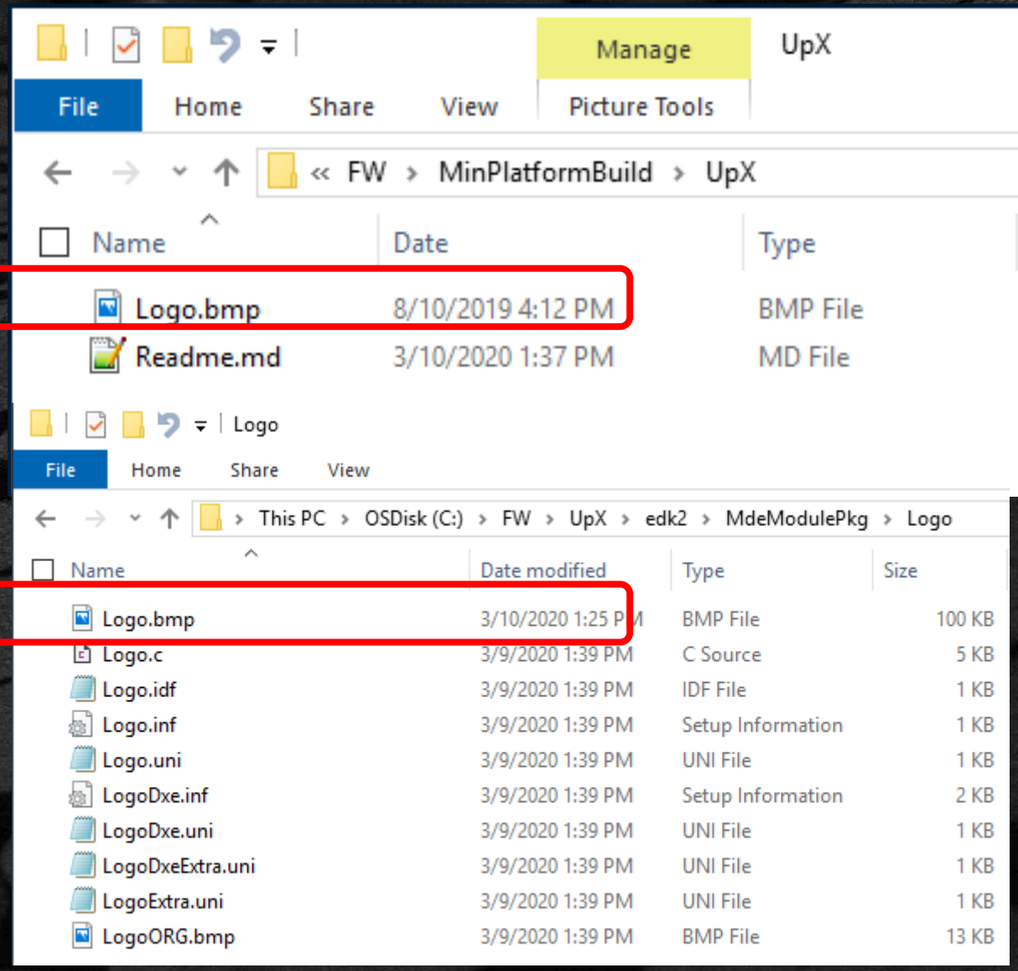
Make a Change

Directory `C:\MinPlatformBuildLab_FW\FW\MinPlatformBuildLab\UpX`

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

Or create a .BMP with Windows Paint

See . . . `WhiskeylakeOpenBoardPkg\UpXtreme\OpenBoardPkg.fdf` line 285

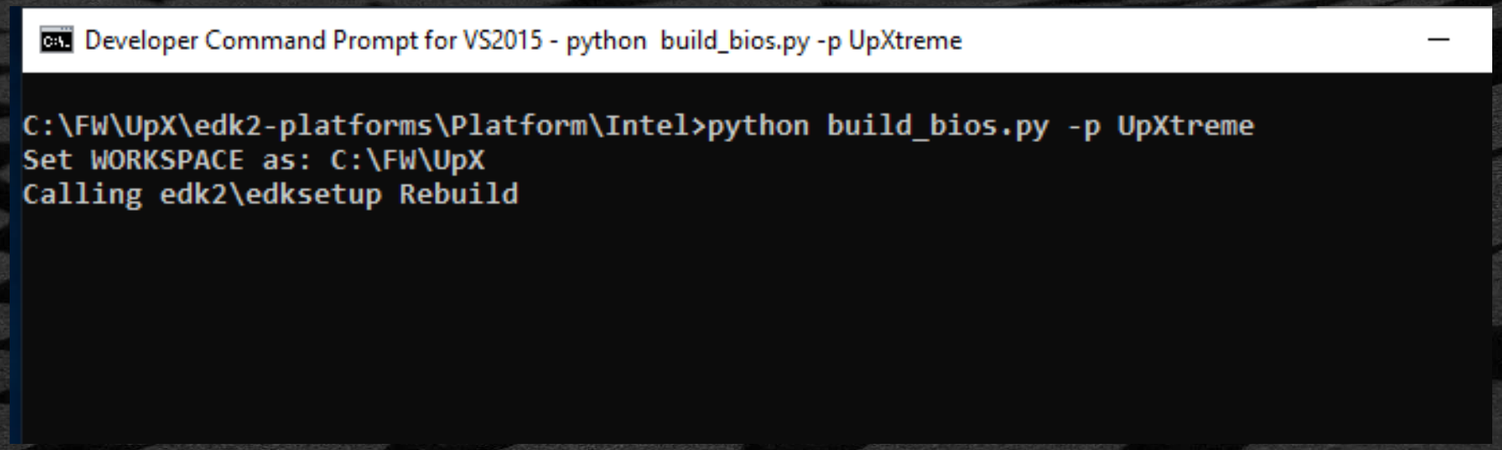


Build with new logo

Invoke the Python Build script for Up Xtreme

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

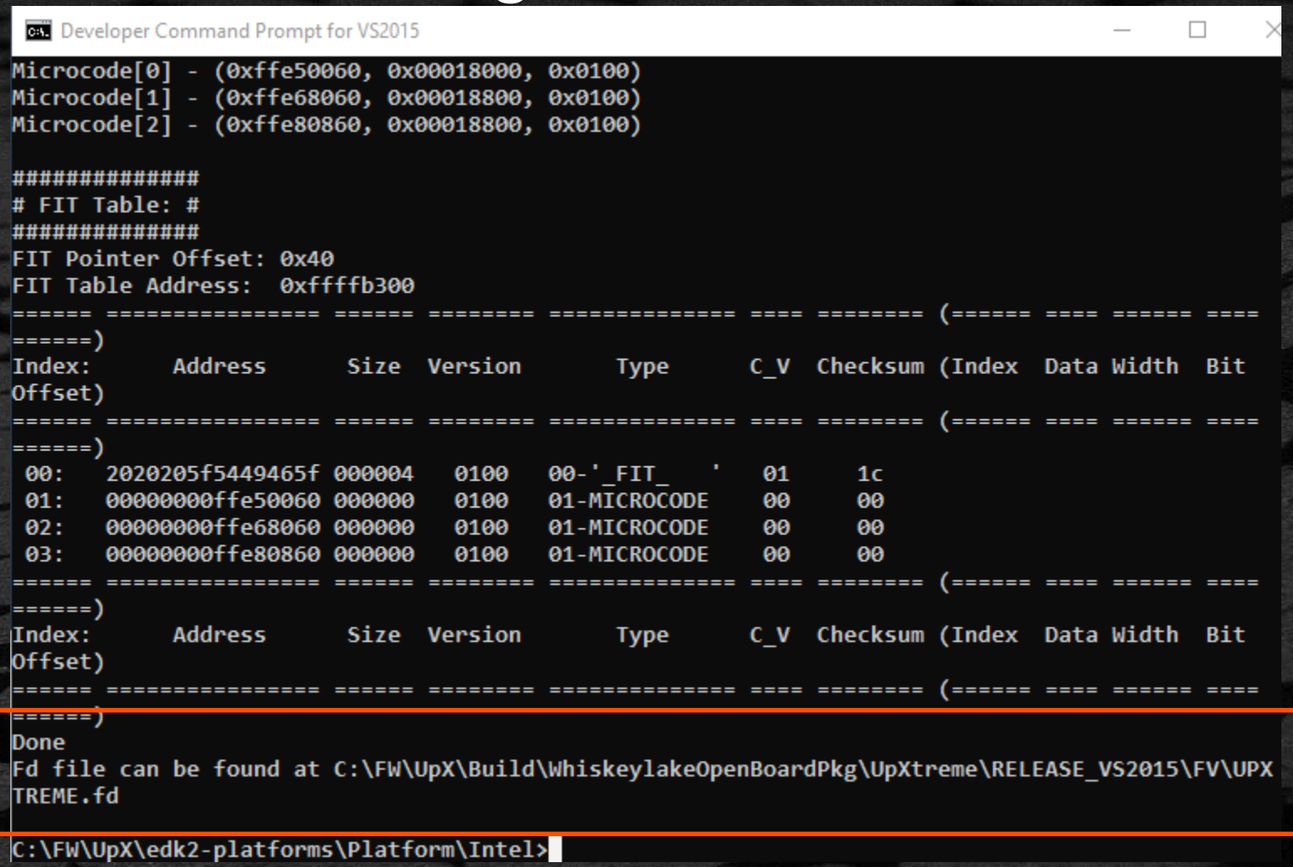
Takes about 2 minutes



Build Process Completed

5

Locate the build .fd images



The script displays the location of the final .fd files

tianocore

SUMMARY

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

tianocore

Questions?

Return to Main Training Page

Return to Training Table of contents for next presentation

Link


tianocore

tianocore

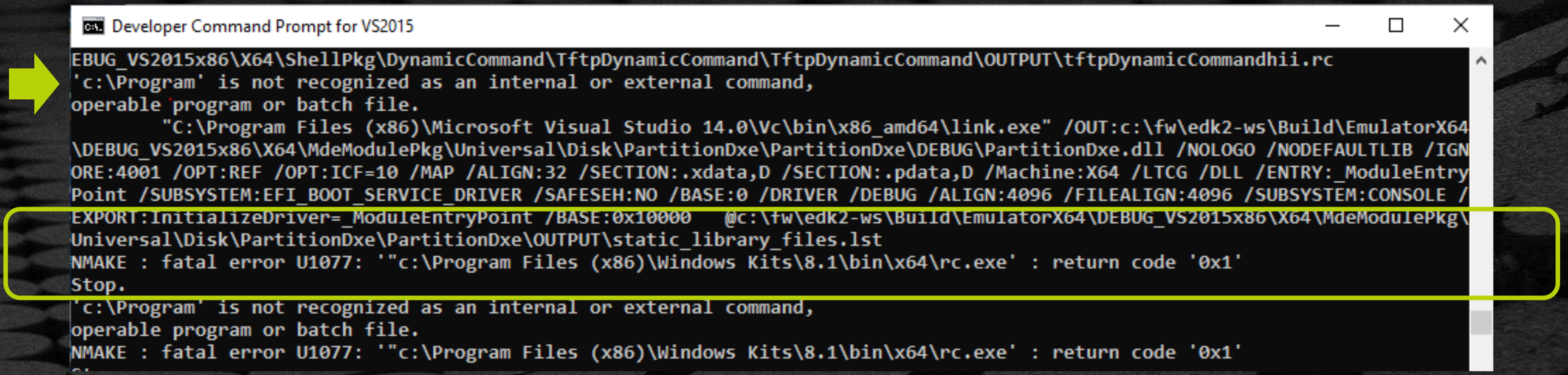
BACKUP

tianocore

BUILD ERRORS

Build Error- RC.exe

Error message:



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

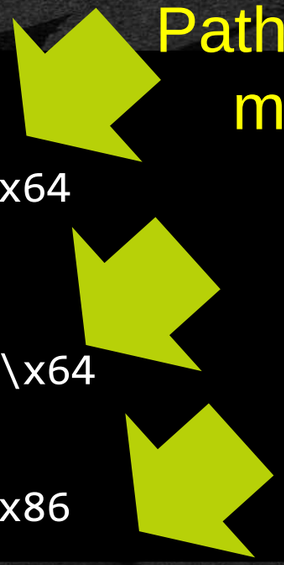
Search for your installation of Visual Studio (2013, 2015, 2017) “RC.EXE”
Probably in path C:\Program Files (x86)\Windows Kits\

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
```



tianocore

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