



Sensor Development Environment Setup Guide

Updated: 2021-08-17
Version: 0.2

Legal Notices and Disclaimers

Disclaimers

INTEL CORPORATION MAKES NO WARRANTY OF ANY KIND WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. INTEL CORPORATION ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS DOCUMENT. INTEL CORPORATION MAKES NO COMMITMENT TO UPDATE NOR TO KEEP CURRENT THE INFORMATION CONTAINED IN THIS DOCUMENT.

THIS SPECIFICATION IS COPYRIGHTED BY AND SHALL REMAIN THE PROPERTY OF INTEL CORPORATION. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED HEREIN.

INTEL DISCLAIMS ALL LIABILITY, INCLUDING LIABILITY FOR INFRINGEMENT OF ANY PROPRIETARY RIGHTS, RELATING TO IMPLEMENTATION OF INFORMATION IN THIS SPECIFICATION. INTEL DOES NOT WARRANT OR REPRESENT THAT SUCH IMPLEMENTATIONS WILL NOT INFRINGE SUCH RIGHTS.

NO PART OF THIS DOCUMENT MAY BE COPIED OR REPRODUCED IN ANY FORM OR BY ANY MEANS WITHOUT PRIOR WRITTEN CONSENT OF INTEL CORPORATION.

INTEL CORPORATION RETAINS THE RIGHT TO MAKE CHANGES TO THESE SPECIFICATIONS AT ANY TIME, WITHOUT NOTICE.

Legal Notices

Intel software products are copyrighted by and shall remain the property of Intel Corporation. Use, duplication or disclosure is subject to restrictions stated in Intel's Software License Agreement, or in the case of software delivered to the government, in accordance with the software license agreement as defined in FAR 52.227-7013.

The Intel logo is a registered trademark of Intel Corporation.

Other brands and names are the property of their respective owners.

Revision History

Revision	Date	Author	Reason for Changes
0.1	10/19/2017	Xiaochun Shi	Initial draft
0.2	8/17/2021	Yukuan Guo	Update and add chapter 5,7

Contents

Contents.....	4
1. Development Environment Setup	5
a) Install Microsoft Visual Studio 2019.....	5
b) Install Microsoft Windows Driver Kit (WDK) 10.....	5
2. Check driver development environment.....	5
3. DDK Package Instruction	5
4. Driver Visual Project Setup.....	6
a) New project setup	6
b) Import common source files for all platform	6
c) Import common source files for Sensor	6
d) Additional Include Directories.....	7
e) Add dependencies and library for sensor project.....	9
f) Enable WPP Trace configuration.....	10
5. Develop Camera Driver with CameraDDK.....	10
a) Implement APIs for sensor	11
b) Graph setting files and tuning files.....	13
c) Update sensor inf config	13
d) Update iacamera driver config.....	14
e) Add sensor trace GUID for WPP trace	15
6. Build Driver	15
7. Sign Driver	15

1. Development Environment Setup

a) Install Microsoft Visual Studio 2019

Download and install Visual Studio 2019 with the default or following:

- Select the custom installation.
- To install C++, go to Program Languages > Visual C++ > Common Tools for Visual C++ 2019
- Install the Microsoft .Net Framework 4.6
- Install latest Windows 10 SDK & tools under “Universal Windows App Development Tools” (This package includes the Windows SDK for Windows 10)
- Click Next to continue and follow the prompts to complete the installation.

Link: <https://developer.microsoft.com/zh-cn/windows/hardware/windows-driver-kit>

b) Install Microsoft Windows Driver Kit (WDK) 10

- Download and install Windows Driver Kit (WDK) 10 with the default

Link: <https://developer.microsoft.com/zh-cn/windows/hardware/windows-driver-kit>

2. Check driver development environment

- Run visual studio 2019
- Click [**File**]→[**New**]→[**Project**]
- Pop up the [**New Project**] dialog
- Select [**Installed**]→[**Templates**]→[**Visual C++**]→[**Windows Drivers**] of [**New Project**] dialog and click it
- If there are WDF in the [windows driver] than development environment install is successful

3. DDK Package Instruction

Unzip DDK zip file to PC. DDK should include following folders

- Documentation: setup & development guides.
- Include: Required header files to compile sensor driver code

- Lib: vcm, nvram, common static libraries
- Sensor: Common source files and example code (just a skeleton)
- Platform: Build script
- Test: Camera Test Cases

4. Driver Visual Project Setup

a) New project setup

Follow <https://docs.microsoft.com/en-us/windows-hardware/drivers/gettingstarted/writing-a-very-small-kmdf--driver> to create empty KMDF project for new sensor driver.

- Run visual studio 2019
- Click [**File**]→[**New**]→[**Project**]
- Pop up the [**New Project**] dialog
- Select [**Installed**]→[**Templates**]→[**Visual C++**]→[**Windows Drivers**]→ [**WDF**]
- of [**New Project**] dialog and click it
- Select the “Kernel Mode Driver, Empty (KMDF)”
- Set the path of the project file to “Sensor” catalog of DDK Package
- Click [**OK**]

After creating a new sensor project, in Sensor folder, there are several files will be created automatically by Visual studio 2019. Take imx362 for example, files are as below: imx362.sln, imx362.vcxproj, imx362.vcxproj.filters, imx362.inf. Other files (imx362.rc, etc), copy from and refer to Sample.

b) Import common source files for all platform

- Right click Solution of Solution Explorer and pop up the dialog
- Select [**Add**]→[**Existing Item**] and click it
- Select common files of DDK Package and click [**Add**]

The common files for TGL Platform are following:

- Common\libiss\src\SsSensor.c
- Common\libiss\src\vcm\vcm_map.c
- Common\libiss\src\nvram\nvram_map.c

c) Import common source files for Sensor

Take imx362 for example.

- Copy files from Sample project Sample.c, Sample_setting.c, SensorTrace.h, device.h to your sensor folder, rename Sample.c as imx362.c, and rename Sample_setting.c as imx362_setting.c
- Manually modify the project file (imx362.vcxproj) to add below files:

```
<ItemGroup>
  <ClInclude Include="device.h" />
  <ClInclude Include="SensorTrace.h" />
</ItemGroup>
<ItemGroup>
  <ClCompile Include="..\Common\libiss\src\nvram\nvram_map.c" />
  <ClCompile Include="..\Common\libiss\src\SsSensor.c" />
  <ClCompile Include="..\Common\libiss\src\vcv\vcv_map.c" />
  <ClCompile Include="imx362.c" />
</ItemGroup>
```

d) Additional Include Directories

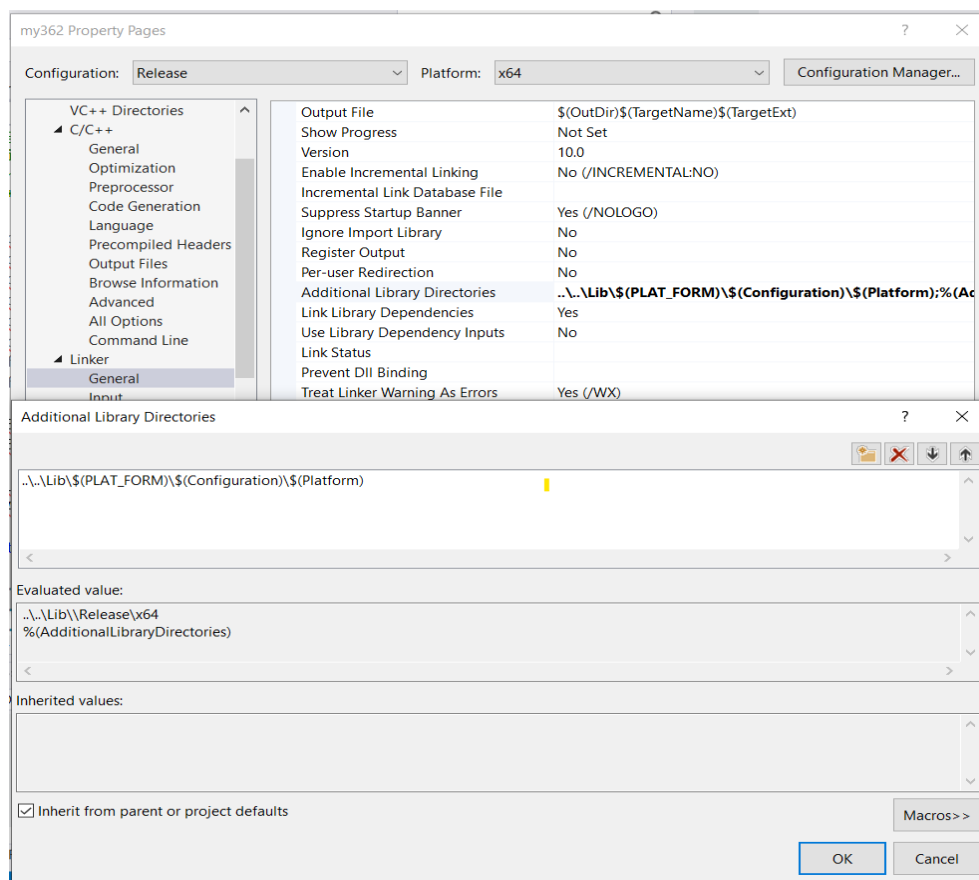
- Right click Solution of Solution Explorer and pop up the dialog
- Click [**Properties**] and pop up the dialog
- Select the [**Configuration Properties**]→[C/C++]→[**General**]
- Set “Additional Include Directories” to values of the following

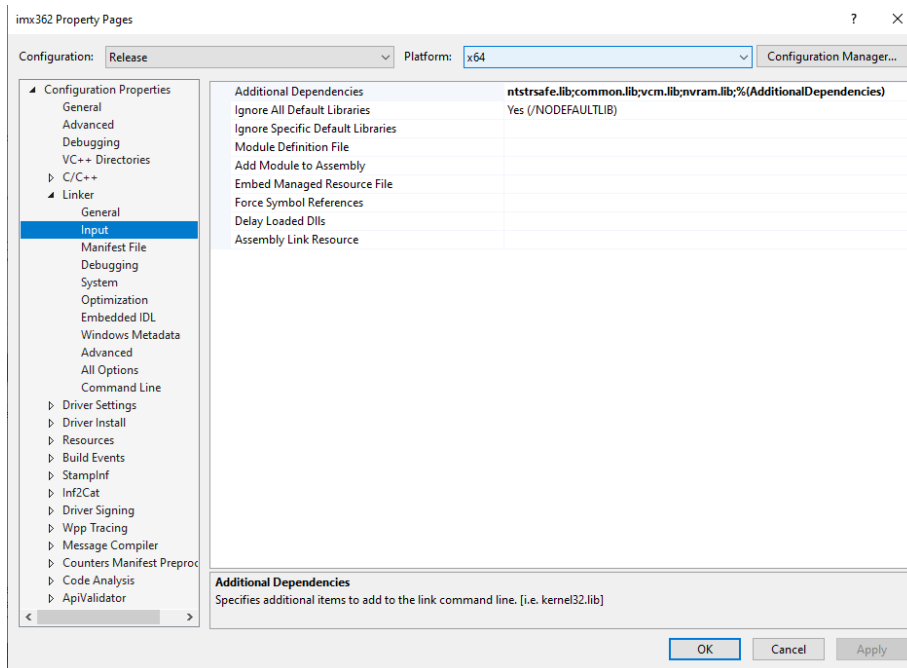
```
.\;
.\skc;
..\common\;
..\common\inc;
..\common\libiss\libinc;
..\common\libiss\src\nvram;
..\common\libiss\src\vcv;
..\..\include;..\..\include\Skycam;
```


e) Add dependencies and library for sensor project

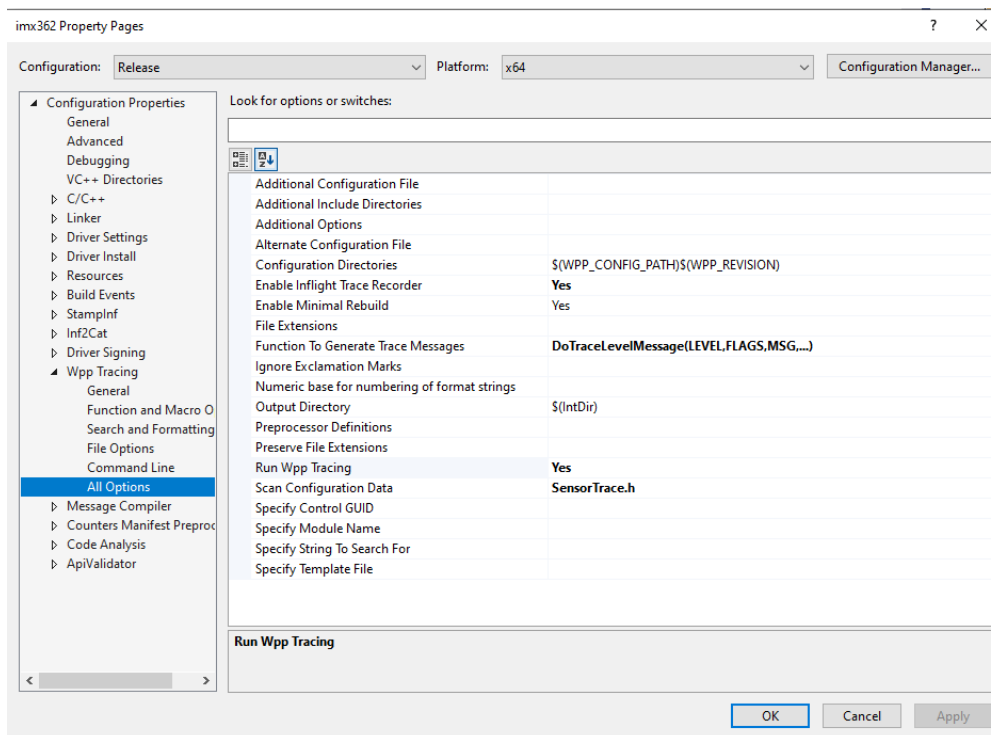
In “linker ->General->Additional Library Directories”:

Lib Files:	Lib Path:
ntstrsafe.lib; common.lib; vcm.lib; nvram.lib	..\..\Lib\\$(PLAT_FORM)\\$(Configuration)\\$(Platform);





f) Enable WPP Trace configuration



5. Develop Camera Driver with CameraDDK

Take imx362 for example.

a) Implement APIs for sensor

- Refine imx362_setting.c base on the sensor register setting
- Important content need implement in device.h. Such as, sensor register address definition, additional export interface definition for this specific sensor.
- Important APIs need to be implemented in imx362.c. imx362.c implements the function definition from device.h, which include the sensor.h and some export function definition specific for this sensor.

Configuration part:

- `const SNSR_MIPI_CONFIG SensorMipiConfig ;`
- `SNSR_CAPS SensorCaps;`
- `DefaultSensorMipiConfig` can be overwrite by BIOS setting.

```
// Sample MIPI config
const SNSR_MIPI_CONFIG DefaultSensorMipiConfig =
{
    MIPI_CSI2_PORT_0,
    MIPI_LANES_4,
    INPUT_FORMAT_RAW_10,
    TYPE_RAW,
};
```

- `DefaultSensorCaps` can be overwrite by developer registry definition.

```
SNSR_CAPS DefaultSensorCaps =
{
    MIPI,
    RAW,
    GET_RATIO(16, 9),           // Aspect ratio
    0, 0,                       // Aperture
    65000, 1,                   // Exposure
    800, 50,                    // ISO
    2000, 100, 2000,            // Focus min/max/default value - in millimeters
    TRUE,                       // Focus support
    TRUE,                       // Flash support
    FALSE,                     // Scale support
    29,                         // FOV FocalLengthX in mm
    26,                         // FOV FocalLengthY in mm
    0,                          // Camera Angle Offset pitch angle
    0,                          // Camera Angle Offset yaw angle
    1,                          // Embedded Line number 1
    1,
#ifdef AE_EXACT_CONFIG
    4,
    4,                          // ae store
#endif
};
```

Resolution assignment:

- Sensor resolution list, is the sensor support output resolution, those resolution list will expose to iacamera driver. Laterly, when iacamera need set resolution to

sensor driver based on use case requirement, it will firstly check this resolution list to make sure the sensor support this resolution output.

```
SENSOR_RESOLUTION sensor_res_preview[] =
{
    {1932, 1092, 30, BINNING1x1, 720000000, BGGR, BGGR, 0, 0, (REGS_SETTING
.....
    RESOLUTION_TABLE_END
};

SENSOR_RESOLUTION sensor_res_still[] =
{
    {1932, 1092, 30, BINNING1x1, 720000000, BGGR, BGGR, 0, 0, (REGS_SETTING
.....
    RESOLUTION_TABLE_END
};

SENSOR_RESOLUTION sensor_res_video[] =
{
    {1932, 1092, 30, BINNING1x1, 720000000, BGGR, BGGR, 0, 0, (REGS_SETTING
.....
    RESOLUTION_TABLE_END
};

SENSOR_RESOLUTION sensor_res_raw[] =
{
    {1932, 1092, 30, BINNING1x1, 720000000, BGGR, BGGR, 0, 0, (REGS_SETTING
.....
    RESOLUTION_TABLE_END
};
```

Initialization interface

- GetTimingFactor() to get timing information for each resolution
- InitContext() to initialize PDEVICE_CONTEXT-> SENSOR_INTERFACE
- Detect() to implement sensor ID read and check
- EnableEmbeddedLine() to enable embedded line function
- InitResolution() to set a wanted resolution setting into sensor
- Stream() to enable/disable output stream
- ModuleCheck() to check sensor power status
- ReleaseI2c() to release all I2C resource

Export interface:

- Cmd_SetExposure() to set exposure time and gain
- Cmd_GetExposure() to get exposure time
- Cmd_GetGain() to get gain
- Cmd_OtpRead() to read OTP data
- Cmd_OtpWrite() to write OTP

b) Graph setting files and tuning files

- Use ICG IQStudio PipeConfiguration tool to generate graph_setting files.
Graph_setting_(SensorName)_(Camera module name)_TGL.xml
- Tuning files, those are shared from Tuning team.
(SensorName)_(Camera module name)_TGL.aiqb
(SensorName)_(Camera module name)_TGL.cpf

c) Update sensor inf config

- Update imx362.inf, copy the .aiqb, .cpf, .xml files to this folder, then change inf to copy those file to target folder in DUT

```
[imx362_Device.NTamd64]
CopyFiles=imx362_Device.NT.Copy
CopyFiles=CopyCPFFiles
CopyFiles=CopyX64ExtraCPFFiles
```

```
[imx362_Device.NT.Copy]
imx362.sys
graph_settings_imx362_A12N08B_TGL.xml
```

```
[CopyCPFFiles]
IMX362_A12N08B_TGL.cpf
IMX362_A12N08B_TGL.aiqb
```

```
[CopyX64ExtraCPFFiles]
IMX362_A12N08B_TGL.cpf
IMX362_A12N08B_TGL.aiqb
```

```
[SourceDisksFiles]
imx362.sys = 1
IMX362_A12N08B_TGL.cpf = 1
IMX362_A12N08B_TGL.aiqb = 1
graph_settings_imx362_A12N08B_TGL.xml = 1
```

```
;----- Service installation
[imx362_Device.NTamd64.Services]
AddService = imx362,%SPSVCINST_ASSOCSERVICE%, imx362_Service_Inst
```

- Update imx362.vcxproj, copy the .aiqb, .cpf, .xml files to release folder.

```
<Target AfterTargets="Build" Name="CopyBinAndCPF">
  <Copy Condition="$(PLAT_FORM.Contains('TGL'))">
```

```

DestinationFolder="$ (BinplacePath)Product\$(DrvPkgDirName)"
SourceFiles="IMX362_A12N08B_TGL.cpf;
            IMX362_A12N08B_TGL.aiqb;
            graph_settings_imx362_A12N08B_TGL.xml" />
</Target>

```

d) Update iacamera driver config

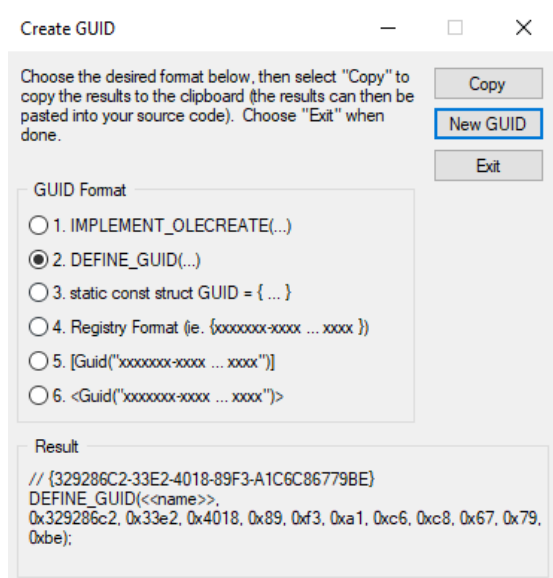
CameraDDK\Include -> include the related iacamera driver header file (sensor_intfclass.h & sensor_intf.h).

Sensor_intf.h define the interface between the iacamera driver and the sensor driver.

SsInitContextCommon will transfer the sensor, VCM and nvram interface to iacamera driver.

iacamera driver query the interface based on SymbolicLinkName.

- Generate new GUID for this new sensor with Visual Studio "Create GUID" tool, and add this definition in file
C:\proj\myproject\w\camerasw\Source\Camera\Include\sensor_intfclass.h



- Add the same GUID in your iacamera extension inf file (i.e. iacamera64_tgl_ext.inf) for this sensor and add filter and pin data range for camera APP to enumerate this sensor. Please refer to imx362 config in iacamera.inf which is released with driver build.
- Please note any changes in .inf need **resign** .inf and generate a new .cat file with DigitalSigning Tool.

Here is MSDN for INF files: <https://docs.microsoft.com/en-us/windows-hardware/drivers/install/overview-of-inf-files>

e) Add sensor trace GUID for WPP trace

- Add sensor GUID in the SensorTrace.h file.

```
#define WPP_CONTROL_GUIDS \  
    WPP_DEFINE_CONTROL_GUID(CtlGuid, (07A61642, E06E, 42EE, B030, A78D7C77CAF1), \  
        WPP_DEFINE_BIT(FLAG_LOG)          \  
        WPP_DEFINE_BIT(FLAG_WARN)         \  
        WPP_DEFINE_BIT(FLAG_ERROR)        \  
    )
```
- Create imx362.ctl file and add the same GUID in it.
07A61642,E06E,42EE,B030,A78D7C77CAF1 Trace
- Add sensor GUID in the trace.ctl in the DriverLog tool
07A61642-E06E-42EE-B030-A78D7C77CAF1 imx362

6. Build Driver

Execute batch file \Platform\BuildCameraSensor.bat to generate sample driver binary. Also you can modify this batch to add your own driver project.

The build results put in CameraDDK\Sensor\imx362\x64\Release\imx362.

Example:

Build SKL & CNL sensor driver: **BuildCameraSensor.bat**

Build SKL sensor driver: **BuildCameraSensor.bat SKL**

Build CNL sensor driver: **BuildCameraSensor.bat CNL**

Build TGL sensor driver: **BuildCameraSensor.bat TGL**

7. Sign Driver

Sign sensor driver(imx362.inf) and resign iacamera64_tgl_ext.inf if there is any change in it.

- Share the build sensor driver package to ICG, Use ICG DigitalSigning Tool to sign.
- Enable test sign during development and test stage
Refer to [Enable Loading of Test Signed Drivers](#)
Bcdedit.exe -set TESTSIGNING ON
Bcdedit.exe /set nointegritychecks ON