Guide to Add New VCM & EEPROM module Into DDK

Updated: 2017-08-27
Version: 0.1

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

Table of Contents

VCM module (take DW9807 for example)

In Vcm_type.h

Add new module name define:

```
#define DW9807_NAME          "DW9807"
```

`In enum VCM_TYPE:`

Replace VCM_ex1 to a real module name like VCM_DW9807;

In Vcm_map.c

1. `#include "vcm_dw9807.h"`
2. Add callback functions in “Vcm[VCM_NUM]”, replace below lines:
   // below are function callback holders
   { VCM_ex1, NONVCM_NAME, 0, NULL, NULL, NULL, NULL, NULL, NULL, NULL },   { VCM_ex2, NONVCM_NAME, 0, NULL, NULL, NULL, NULL, NULL, NULL, NULL },
   to

```
    { VCM_DW9807, DW9807_NAME, DW9807_DEFAULT_POS, DW9807_Init, DW9807_ResetPos,
DW9807_SetPos, DW9807_GetPos, DW9807_GetStatus, DW9807_GetHPStatus, DW9807_SetConfig },
    { VCM_ex2, NONVCM_NAME, 0, NULL, NULL, NULL, NULL, NULL, NULL, NULL },
```

```
 3.  In NTSTATUS ParseVcm(), add codes about dw9807:
    NTSTATUS ParseVcm(

    )
{
    ANSI_STRING ad5823;
    ......
    ANSI_STRING dw9807;

    ANSI_STRING name;

    if (!pSsVcm)
    {
        return STATUS_INVALID_PARAMETER;
    }

    RtlInitAnsiString(&ad5823, AD5823_NAME);
    ......
    RtlInitAnsiString(&dw9807, DW9807_NAME);

    RtlInitAnsiString(&name, pVcmName);

    if(RtlEqualString(&ad5823, &name, TRUE))
        pSsVcm->VcmType = VCM_AD5823;

    ......
    else if (RtlEqualString(&dw9807, &name, TRUE))
        pSsVcm->VcmType = VCM_DW9807;
    else
        pSsVcm->VcmType = VCM_NONE;

    DoTraceMessage(FLAG_LOG, "%s SensorCtx->VcmType:%d", DEVICE_NAME, pSsVcm->VcmType);

    return STATUS_SUCCESS;
     }
```

4. Add DW9807 source file and header file under CameraDDK\Sensor\Common\libiss\src\vcm

   Vcm_dw9807.c & vcm_dw9807.h provided as template to implement VCM control functions

5. There are some functions might help you to write a new module, which is included in vcm.h.

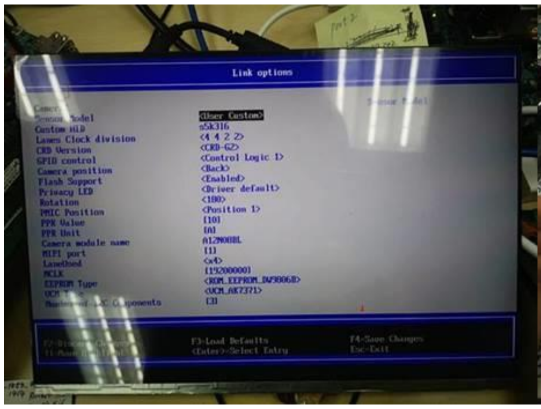
EEPROM module

Check EEPROM I2C salve address, then configure it in BIOS.

1. The salve I2C address is defined in the datasheet like below. Take 0xB0 for example, its binary representation is 10110000, right shift one bit, change to 01011000(0x58), configure 0x58 in BIOS.(take DW9808 for example)

   - ID : I2C Slave ID selection
     ☞ Low : 0x1C (VCM) / 0xA0 (EEPROM)
     ☞ High : 0x18 (VCM) / 0xB0 (EEPROM)

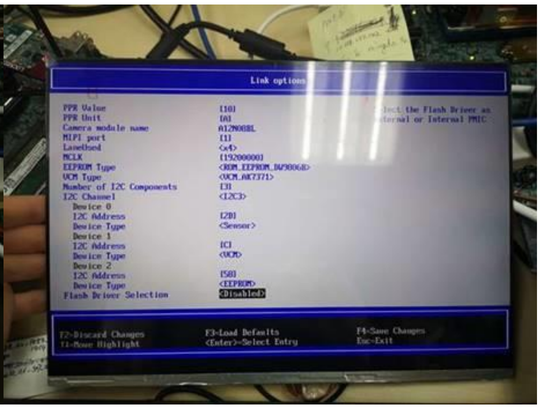
2. The final BIOS setting for Cam1 Link Option like below. In the Control Logic 1 Link Option, change the "EEPROM Type" to "ROM_EEPROM_xxx". (If BIOS don't support this new defined EEPROM Type, you can set an existing EEPROM Type. Then mocked it to new defined EEPROM Type in sensor driver). Change the "Number of I2C Components" to "3". In <Deivce 2>, Set I2C Address to "0x58", Set Device Type to "EEPROM".



```
                         Link options

Camera                                                  Sensor Model
Sensor Model             <User Custom>
Custom HID               s5k3l6
Lanes Clock division     <4 4 2 2>
CRD Version              <CRD-G2>
GPIO control             <Control Logic 1>
Camera position          <Back>
Flash Support            <Enabled>
Privacy LED              <Driver default>
Rotation                 <180>
PMIC Position            <Position 1>
PPR Value                [10]
PPR Unit                 [A]
Camera module name       A12N08BL
MIPI port                [1]
LaneUsed                 <x4>
MCLK                     [19200000]
EEPROM Type              <ROM_EEPROM_DW9806B>
VCM Type                 <VCM_AK7371>
Number of I2C Components [3]

F2-Discard Changes       F3-Load Defaults         F4-Save Changes
F1-Move Highlight        <Enter>-Select Entry     Esc-Exit
```



```
                         Link options

PPR Value                [10]                  Select the Flash Driver as
PPR Unit                 [A]                   External or Internal PMIC
Camera module name       A12N08BL
MIPI port                [1]
LaneUsed                 <x4>
MCLK                     [19200000]
EEPROM Type              <ROM_EEPROM_DW9806B>
VCM Type                 <VCM_AK7371>
Number of I2C Components [3]
I2C Channel              <I2C3>
  Device 0
  I2C Address            [2D]
  Device Type            <Sensor>
  Device 1
  I2C Address            [C]
  Device Type            <VCM>
  Device 2
  I2C Address            [58]
  Device Type            <EEPROM>
Flash Driver Selection   <Disabled>

F2-Discard Changes       F3-Load Defaults         F4-Save Changes
F1-Move Highlight        <Enter>-Select Entry     Esc-Exit
```

In nvram_type.h

Add new defined EEPROM type in enum type `NVM_TYPE`.

```
// place holder, change to real name when needed
NVM_M24C64,
```

Add new EEPROM files

1. Add new M24C64 source and header file (nvm_M24C64S.c & nvm_M24C64S.h) under folder Camera\Sensor\commonddk\libiss\src\nvram. There are some functions defined in nvram.h and nvm_M24C64S.h to read and write the EEPROM.

2. `Cmd_M24c64Write and Cmd_M24c64Read defined in` nvm_M24C64S.h, which should be implemented in nvm_M24C64S.c for specific EEPROM module. The implement details should refer to the EEPROM datasheet.

   `Check the Page size and Memory size in the EEPROM datasheet. There should be some description in the datasheet like:`

   - Memory organization: 64Kb (8,192 x 8)
   - Page Size: 32 bytes

Describe that information with Macro M24C64_PAGE_SIZE and M24C64_PAGE_NUM in nvm_M24C64S.h, like below code snippet:

```
#define M24C64_PAGE_SIZE  32         //bytes, Column address 0x00 to 0x1F
#define M24C64_PAGE_NUM   256    //Pages, 0x00 to 0xFF
```

nvm_M24C64S.c also define some functions named like: M24c64RomWrite/M24c64RomRead or RomWriteX/RomReadX to read/write this EEPROM by pages. Developer should define the Data Memory Address based on the Page, Addr and the address bytes format in the datasheet.

```
    buf[0] = (UINT8)(Page >> 3 & 0x1F);
    buf[1] = (UINT8)(Page << 5 & 0xE0) | (UINT8)(Addr);
```

In nvram_map.c

1. Include header file of this EEPROM.

```
#include "nvm_M24C64S.h"
```

2. Add new defined EEPROM type to the NVM_TYPE type array.

```
NVM_TYPE NvmType[] =
{
    NVM_NONE,
    NVM_OTP,
    ……
    NVM_M24C64,
    NVM_DW9806B,
    NVM_CAT24C16,
    // New NVM added to last in case to change above reserved values being used
externally
    NVM_CAT24C64,
    NVM_24AA16,
    NVM_DW9808,
};
```

3. In “Nvm[NVM_NUM]” add one line like below:

```
NVM_FUNC Nvm[NVM_NUM] =
{
......
    // below are function callback holders, please add module above this line for
inside.
    { NVM_M24C64, 8192, 32, M24C64_ADDR_MIN, Cmd_M24c64Write, Cmd_M24c64Read,
GetNvmData },
 }
```

In nvram.c

Include the new EEPROM head file and source file at begin and end of nvram.c.

```
#include "nvm_M24C64S.h"
#include "nvm_M24C64S.c"
```