Avnet's Blackfin BF609
Embedded Vision Starter Kit

Video Pass-Through Tutorial

Version 1.0

Revision History

Version	Description	Date
1.0	Preliminary Version	May 15, 2013

Table of Contents

Revision History 1
Table of Contents 2
Table of Figures 2
Table of Tables 3
FinBoard BF609 Embedded Vision Starter Kit 3
About this Guide ..... - 1 -
Notation ..... - 2 -
Video Pass-Through Tutorial ..... - 3 -
Overview ..... - 3 -
Table 1 – Supported Output Video Resolutions ..... - 4 -
Start from an existing BF609 EZ-KIT example project ..... - 4 -
Modify the BF609 EZ-KIT project for the FinBoard hardware ..... - 5 -
Execute project on the FinBoard hardware ..... - 7 -
Increase the video resolution to 720P ..... - 10 -
Solution Archive ..... - 13 -
References ..... - 14 -
FinBoard ..... - 14 -
Analog Devices ..... - 14 -
Aptina ..... - 14 -
Known Issues and Limitations ..... - 15 -
VideoLoopbackYUV – Include path not found (\Blackfin\include). ..... - 15 -
VideoLoopbackYUV – sensor.c semantic errors ..... - 16 -
VideoLoopbackYUV – example project is not copied to user workspace ..... - 17 -
Troubleshooting ..... - 18 -
VideoLoopbackYUV – could not open source file “adi_mt9m114.h” ..... - 18 -

Table of Figures

Figure 1 – FinBoard Video Pass-Through – Block Diagram ..... - 3 -

Table of Tables

Table 1 – Supported Output Video Resolutions ..... - 4 -

FinBoard BF609 Embedded Vision Starter Kit

LICENSE AGREEMENT

THE AVNET DESIGN KIT ("DESIGN KIT" OR "PRODUCT") AND ANY SUPPORTING DOCUMENTATION ("DOCUMENTATION" OR "PRODUCT DOCUMENTATION") IS SUBJECT TO THIS LICENSE AGREEMENT ("LICENSE"). USE OF THE PRODUCT OR DOCUMENTATION SIGNIFIES ACCEPTANCE OF THE TERMS AND CONDITIONS OF THIS LICENSE. THE TERMS OF THIS LICENSE AGREEMENT ARE IN ADDITION TO THE AVNET CUSTOMER TERMS AND CONDITIONS, WHICH CAN BE VIEWED AT www.em.avnet.com. THE TERMS OF THIS LICENSE AGREEMENT WILL CONTROL IN THE EVENT OF A CONFLICT.

1. Limited License. Avnet grants You, the Customer, ("You" "Your" or "Customer") a limited, non-exclusive, non-transferable, license to: (a) use the Product for Your own internal testing, evaluation and design efforts at a single Customer site; (b) create a single derivative work based on the Product using the same semiconductor supplier product or product family as used in the Product; and (c) make, use and sell the Product in a single production unit. No other rights are granted and Avnet and any other Product licensor reserves all rights not specifically granted in this License Agreement. Except as expressly permitted in this License, neither the Design Kit, Documentation, nor any portion may be reverse engineered, disassembled, decompiled, sold, donated, shared, leased, assigned, sublicensed or otherwise transferred by Customer. The term of this License is in effect until terminated. Customer may terminate this license at any time by destroying the Product and all copies of the Product Documentation.
2. Changes. Avnet may make changes to the Product or Product Documentation at any time without notice. Avnet makes no commitment to update or upgrade the Product or Product Documentation and Avnet reserves the right to discontinue the Product or Product Documentation at any time without notice.
3. Limited Warranty. ALL PRODUCTS AND DOCUMENTATION ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND. AVNET MAKES NO WARRANTIES, EITHER EXPRESS OR IMPLIED, WITH RESPECT TO THE PRODUCTS AND DOCUMENTATION PROVIDED HEREUNDER. AVNET SPECIFICALLY DISCLAIMS THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE AND ANY WARRANTY AGAINST INFRINGEMENT OF ANY INTELLECTUAL PROPERTY RIGHT OF ANY THIRD PARTY WITH REGARD TO THE PRODUCTS AND DOCUMENTATION.
4. **LIMITATIONS OF LIABILITY. CUSTOMER SHALL NOT BE ENTITLED TO AND AVNET WILL NOT LIABLE FOR ANY INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES OF ANY KIND OR NATURE, INCLUDING, WITHOUT LIMITATION, BUSINESS INTERRUPTION COSTS, LOSS OF PROFIT OR REVENUE, LOSS OF DATA, PROMOTIONAL OR MANUFACTURING EXPENSES, OVERHEAD, COSTS OR EXPENSES ASSOCIATED WITH WARRANTY OR INTELLECTUAL PROPERTY INFRINGEMENT CLAIMS, INJURY TO REPUTATION OR LOSS OF CUSTOMERS, EVEN IF AVNET HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. THE PRODUCTS AND DOCUMENTATION ARE NOT DESIGNED, AUTHORIZED OR WARRANTED TO BE SUITABLE FOR USE IN MEDICAL, MILITARY, AIR CRAFT, SPACE OR LIFE SUPPORT EQUIPMENT NOR IN APPLICATIONS WHERE FAILURE OR MALFUNCTION OF THE PRODUCTS CAN REASONABLY BE EXPECTED TO RESULT IN A PERSONAL INJURY, DEATH OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE. INCLUSION OR USE OF PRODUCTS IN SUCH EQUIPMENT OR APPLICATIONS, WITHOUT PRIOR AUTHORIZATION IN WRITING OF AVNET, IS NOT PERMITTED AND IS AT CUSTOMER'S OWN RISK. CUSTOMER AGREES TO FULLY INDEMNIFY AVNET FOR ANY DAMAGES RESULTING FROM SUCH INCLUSION OR USE.**
5. **LIMITATION OF DAMAGES. CUSTOMER'S RECOVERY FROM AVNET FOR ANY CLAIM SHALL NOT EXCEED CUSTOMER'S PURCHASE PRICE FOR THE PRODUCT GIVING RISE TO SUCH CLAIM IRRESPECTIVE OF THE NATURE OF THE CLAIM, WHETHER IN CONTRACT, TORT, WARRANTY, OR OTHERWISE.**
6. **INDEMNIFICATION. AVNET SHALL NOT BE LIABLE FOR AND CUSTOMER SHALL INDEMNIFY, DEFEND AND HOLD AVNET HARMLESS FROM ANY CLAIMS BASED ON AVNET'S COMPLIANCE WITH CUSTOMER'S DESIGNS, SPECIFICATIONS OR IN¬STRUCTIONS, OR MODIFICATION OF ANY PRODUCT BY PARTIES OTHER THAN AVNET, OR USE IN COMBINATION WITH OTHER PRODUCTS.**
7. U.S. Government Restricted Rights. The Product and Product Documentation are provided with "RESTRICTED RIGHTS." If the Product and Product Documentation and related technology or documentation are provided to or made available to the United States Government, any use, duplication, or disclosure by the United States Government is subject to restrictions applicable to proprietary commercial computer software as set forth in FAR

52.227-14 and DFAR 252.227-7013, et seq., its successor and other applicable laws and regulations. Use of the Product by the United States Government constitutes acknowledgment of the proprietary rights of Avnet and any third parties. No other governments are authorized to use the Product without written agreement of Avnet and applicable third parties.

8. Ownership. Licensee acknowledges and agrees that Avnet or Avnet's licensors are the sole and exclusive owner of all Intellectual Property Rights in the Licensed Materials, and Licensee shall acquire no right, title, or interest in the Licensed Materials, other than any rights expressly granted in this Agreement.

9. Intellectual Property. All trademarks, service marks, logos, slogans, domain names and trade names (collectively "Marks") are the properties of their respective owners. Avnet disclaims any proprietary interest in Marks other than its own. Avnet and AV design logos are registered trademarks and service marks of Avnet, Inc. Avnet's Marks may be used only with the prior written permission of Avnet, Inc.

10. General. The terms and conditions set forth in the License Agreement or at www.em.avnet.com will apply notwithstanding any conflicting, contrary or additional terms and conditions in any purchase order, sales acknowledgement confirmation or other document. If there is any conflict, the terms of this License Agreement will control. This License may not be assigned by Customer, by operation of law, merger or otherwise, without the prior written consent of Avnet and any attempted or purported assignment shall be void. Licensee understands that portions of the Licensed Materials may have been licensed to Avnet from third parties and that such third parties are intended beneficiaries of the provisions of this Agreement. In the event any of the provisions of this Agreement are for any reason determined to be void or unenforceable, the remaining provisions will remain in full effect.This constitutes the entire agreement between the parties with respect to the use of this Product, and supersedes all prior or contemporaneous understandings or agreements, written or oral, regarding such subject matter. No waiver or modification is effective unless agreed to in writing and signed by authorized representatives of both parties. The obligations, rights, terms and conditions shall be binding on the parties and their respective successors and assigns. The License Agreement is governed by and construed in accordance with the laws of the State of Arizona excluding any law or principle, which would apply the law of any other jurisdiction. The United Nations Convention for the International Sale of Goods shall not apply.

About this Guide

This manual describes how to quickly implement a simple video pass-through on FinBoard from Avnet Electronics Marketing.

Please consult the FinBoard Getting Started Guide for more information on:

- Getting Started with the FinBoard
- Software requirements
- Hardware requirements

If not done so already, start with the FinBoard Getting Started Guide for information on tool installation and board setup.

Notation

Code excerpts in this tutorial are typeset using the familiar courier font, as shown below:

```
/* Configure the ... switches on BF609 EZ-Board */
ConfigSoftSwitches_BF609();
```

The portions of code that need to be modified by the user are identified in **BOLD**. As an example, the following code excerpt indicates that one line needs to be commented out, and three lines of code need to be added:

```
/* Configure the ... switches on BF609 EZ-Board */
//ConfigSoftSwitches_BF609();
FINBOARD_CLK_Synth_Config_OUT4_27_00_MHz();
FINBOARD_LED_Drivers_Init();
FINBOARD_LED_Drivers_Config(1);
```

Expected output from the serial console is also typeset with the courier font, with an additional border as shown below:

```
Auto-detecting devices on the JTAG chain...

TDO <-----+
          |
         [0] - [ADSP-BF609 rev 0.0 from Analog Devices]
          |
TDI >-----+
Loading application: "C:\...\Debug\VideoLoopbackYUV.dxe"
```

Video Pass-Through Tutorial

Overview

This tutorial will show you how to use the Analog Devices Cross-Core® Embedded Studio (CCES) tool suite to quickly create a video pass-through application for FinBoard.

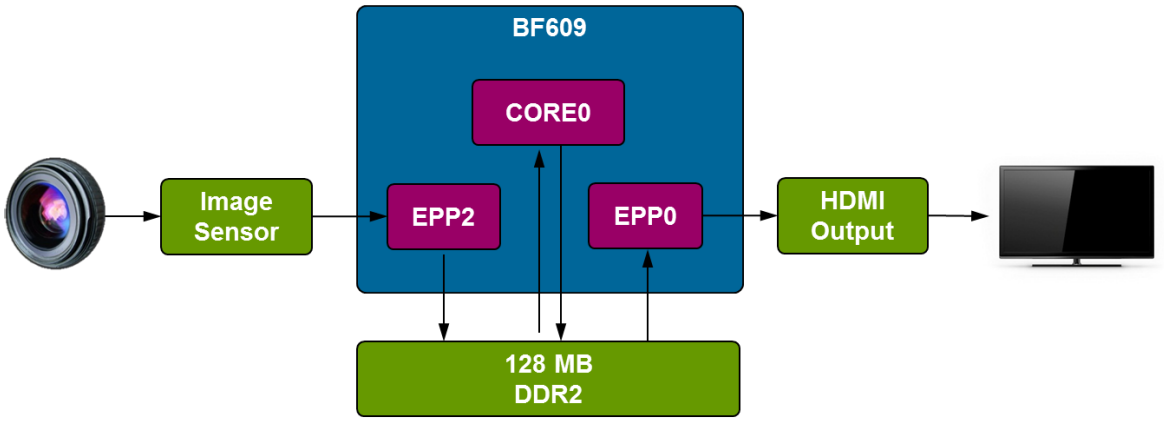


Figure 1 – FinBoard Video Pass-Through – Block Diagram

The application makes use of the ADSP-BF609's Enhanced Parallel Peripheral Interfaces (EPPI) to capture and transmit video content to/from external memory:

- EPPI2 : video input is received from the on-board Aptina MT9M114 image sensor
- EPPI0 : video output is generated on the ADV7511 HDMI transmitter

Only one of the BF609's BlackFin® cores is used in this application, and is responsible for:

- system initialization (image sensor, HDMI output, etc…)
- video buffer management
    - two buffers (ping/pong) are passed between EPPI2 and EPPI0, implementing a video pass-through

The image sensor captures video at 30 frames per second, and the output transmits video at 60 frames per second. Frame rate conversion is implemented by simply repeating frames at the output.

The video content captured from the image sensor is in 16 bit YCbCr 4:2:2 format, with the Cb and Cr channels already chroma sub-sampled

The following output video resolutions, generated at the HDMI output interface, are supported:

Resolution	Pixel Rate (MHz)	Frame Dimensions
480P60	27.00 MHz	720 x 480
720P60	74.25 MHz	1280 x 720

Table 1 – Supported Output Video Resolutions

Start from an existing BF609 EZ-KIT example project

CCES includes many example projects for the BF609 EZ-KIT that can be easily targeted to the FinBoard. This tutorial will use the VideoLoopbackYUV example to demonstrate the steps required.

Open CrossCore Embedded Studio (CCES), and specify a workspace.

If the Welcome window is open, close it.

Browse the existing BF609 EZ-KIT example projects

1. In the menu, select **Help → Browse Examples**
2. For the Processor drop-down list, select **ADSP-BF609**

Select the project that most closely resembles the application you want to create. In the case of this tutorial, we will select the **VideoLoopbackYUV** example project, which implements video pass-through between the MT9M114 image sensor and ADV7511 HDMI transmitter.

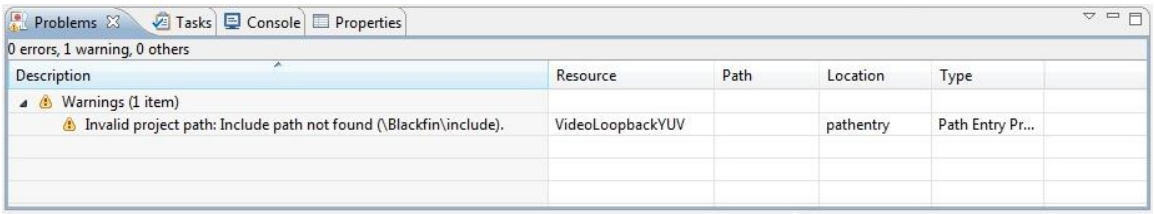
3. In the Search result window, select the **VideoLoopbackYUV** project.
4. Click on the **Open example** button.

Make sure this example project builds correctly.

5. In the Project Explorer window,
   Right-click on the VideoLoopbackYUV project and select **Clean Project**
6. Right-click on the VideoLoopbackYUV project and select **Build Project**.

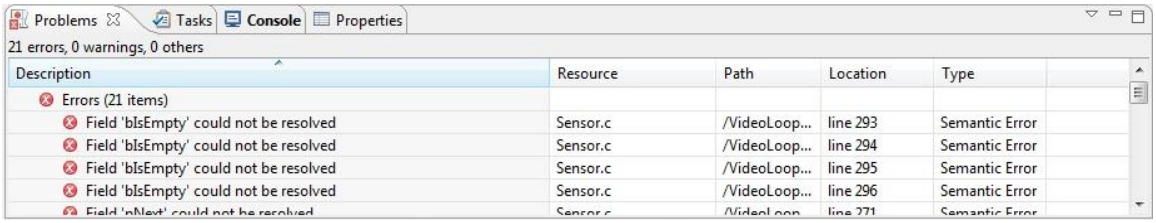
There are two known issues with the VideoLoopbackYUV project from the “Camera EI3 Extender Board v1.0.1” board support package. Neither issue prevent the project from building and executing successfully, however they are summarized below for your information.

The first issue will manifest itself as a warning, visible in the “Problems” tab:



This warning is not critical and can be ignored. If you want to fix it, refer to the **Known Issues and Limitations** section.

The second issue will occur when the **sensor.c** source file, and manifest itself as semantic errors, also visible in the “Problems” tab:



These errors will not prevent the project from building and executing on hardware. If you want to fix these, refer to the **Known Issues and Limitations** section.

If you run into other build errors, please refer to the **FinBoard Getting Started Guide** for more information.

Now that we have a clean project, we can modify it for use with FinBoard.

Modify the BF609 EZ-KIT project for the FinBoard hardware

For BF609 EZ-KIT example projects using the following peripherals, no changes should be required for FinBoard:

- Ethernet
- USB-OTG

For example projects using the following peripherals, the project must be modified:

- ADV7511, when used in 16 bit YCbCr 4:2:2 mode
- MT9M114

Modify the project to map the MT9M114 image sensor input to the EPPI2 port instead of the EPPI1 port.

1. In the Project Explorer window, expand the VideoLoopbackYUV project anddouble-click on the **system.svc** file
2. Click on the **Pin Multiplexing** tab at the bottom of the System Configuration Overview window.
3. In the Peripherals list, click on the **EPPI1 [EPPI Module]** check box to de-select all the EPPI1 pins
4. Expand the **EPPI2 [EPPI Module]**, and select the following EPPI2 pins

   D00 – D07
   CLK
   FS1
   FS2
5. Save and Close the system.svc file by closing the tab near the top menu.

6. In the Project Explorer window, Double-click on the **Sensor.h** file
7. Modify the **ADI_MT9M114_PPI_DEVNUM** definition to the value **2**

```
/* PPI Device number to receive MT91M114 sensor Data */
#define ADI_MT9M114_PPI_DEV_NUM         (2u)
```

8. Save and Close the Sensor.h file

Add the finboard_bsp.c / finboard_bsp.h files to the project. These files are included in this tutorial archive, they can be found in the location where the archive was extracted.

9. In the menu, select **File → Import**
10. Expand the **General** section and select **File System**
11. Click **Next**
12. Click the **Browse** button next to **From directory**
    a. Browse to the location of the finboard_bsp.c/.h files
    b. Click **OK**
13. Check the boxes for the following two files

    finboard_bsp.c
    finboard_bsp.h

14. Click the **Browse** button next to **Into folder**
    a. Select the VideoLoopbackYUV folder
    b. Click **OK**
15. Click **Finish**

Replace the BF609 EZ-KIT specific initialization with FinBoard specific initialization.

16. In the Project Explorer window,
    Double-click on the **VideoLoopbackYUV.c** file
17. After the “VideoLoopbackYUV.h” header include directive,
    Add the following include directive

```
#include "finboard_bsp.h"
```

18. **Comment out** or delete the function declaration for **ConfigSoftSwitches_BF609( )**

```
//extern void ConfigSoftSwitches_BF609(void);
```

19. **Comment out** or delete the call to **ConfigSoftSwitches_BF609( )**,
    which configures soft switches via I2C that are specific to the BF609 EZ-KIT.

```
/* Configure the ... switches on BF609 EZ-Board */
//ConfigSoftSwitches_BF609();
```

20. At the same location in the code, add the following call to configure port OUT4 of FinBoard's clock synthesizer to generate 27.00 MHz for the HDMI output interface.

```
FINBOARD_CLK_Synth_Config_OUT4_27_00_MHz();
```

21. At the same location in the code, add the following call to enable LED illumination (specify desired intensity : 0 is OFF, 1-7 is ON of varying intensity)

```
FINBOARD_LED_Drivers_Init();
FINBOARD_LED_Drivers_Config(1);
```

22. After the ConfigEncoder( ) call, add the following call to apply the FinBoard specific configuration for 16 bit YCbCr 4:2:2 video output mode

```
FINBOARD_ADV7511_16bit_Mode();
```

23. Just before the printf("All done \n"); call at the end of the main( ) function, add the following call to disable LED illumination, which will give a visual indication that the video pass-through application has completed.

```
FINBOARD_LED_Drivers_Config(0);
```

24. Save and Close the VideoLoopbackYUV.c file

Build the project.

25. In the Project Explorer window,
Right-click on the VideoLoopbackYUV project and select **Build Project**.

Execute project on the FinBoard hardware

The steps to execute a project on the FinBoard hardware are the same as for the BF609 EZ-KIT.

1. In the Project Explorer window,
Right-click on the VideoLoopbackYUV project
and select **Run As**, then **Run Configurations …**
2. Select **CrossCore Embedded Studio Application**
3. Select the **New** button to create a new configuration
4. In the **Select Processor** dialog:
    a. Select **Blackfin** for Processor family
    b. Select **ADSP-BF609** for Processor type
    c. Click **Next**
5. In the **Select Connection Type** dialog:
    a. Select **Emulator**
    b. Click **Next**
6. In the **Select Platform** dialog:

a. Select the **ADSP-BF609 via ICE-100B**

b. Click **Finish**

7. Ensure that the new **VideoLoopbackYUV Debug** configuration is selected in the "Program(s) to load:" window.
8. Click **Apply** to save the configuration
9. Click **Run** to execute the configuration
10. If you get a **Terminate Session** dialog (from a previous session), Click **Yes**.
11. If you get a **No Program Selected** dialog (for core 1), Click **Yes**.

You should see the following output in the Console:

```
Auto-detecting devices on the JTAG chain...

TDO <-----+
          |
         [0] - [ADSP-BF609 rev 0.0 from Analog Devices]
          |
TDI >-----+
Loading application: "C:\...\Debug\VideoLoopbackYUV.dxe"
Configuring Clock Synthesizer OUT4 for 480P60 resolution
LED Drivers Initialization
      LED Driver #1 Initialization
            STATUS1 = 0x00
      LED Driver #2 Initialization
            STATUS1 = 0x00
adi_twi_Write failed
adi_twi_Write failed
adi_twi_Write failed
adi_twi_Write failed
Configuring ADV7511 for 16bit YCbCr Mode
```

Your monitor should report 480P60 or "720x480 @ 60Hz" resolution and should display the live video captured by the image sensor. During the video pass-through, the on-board LEDs will be enabled.

After a few seconds, the LEDs will disable, and you will see the following output on the console, indicating that pass-through application has completed:

```
All done
```

To adjust how long the application runs, simply modify the EXAMPLE_TIMEOUT definition in the "VideoLoopbackYUV.h" header file. The code is shown below:

```
#define EXAMPLE_TIMEOUT                 500
```

The "`adi_twi_Write failed`" lines correspond to I2C transactions that are BF609 EZ-KIT specific, and are therefore failing on the FinBoard hardware.

It is left as an exercise to the user to find where these I2C transactions occur, and to comment them out.

HINT : the ConfigSoftSwitches_BF609( ) call would have generated more of these message if we had not commented it out. Look for a similar call in the encoder.c source file.

After modifying the code, rebuilding, and executing on hardware, you should see the following output on the console.

```
Loading application: "C:\...\Debug\VideoLoopbackYUV.dxe"
Configuring Clock Synthesizer OUT4 for 480P60 resolution
LED Drivers Initialization
      LED Driver #1 Initialization
            STATUS1 = 0x00
      LED Driver #2 Initialization
            STATUS1 = 0x00
Configuring ADV7511 for 16bit YCbCr Mode
All done
```

Increase the video resolution to 720P

This section will illustrate the flexibility of the drivers provided by ADI.

First, add a definition to select the video resolution in the main header file.

1. In the Project Explorer window,
   Double-click on the **VideoLoopbackYUV.h** file
2. Add the following definition after the EXAMPLE_TIMEOUT definition

```
#define VIDEOLOOPBACKYUV_720P
```

3. Save and Close the VideoLoopbackYUV.h file

Using this preprocessor definition, we will modify the code to add support for 720P resolution. All of the modifications will be done using the following syntax:

```
#if defined(VIDEOLOOPBACKYUV_720P)
   // 720P specific code …
#else
   // 480P specific code …
#endif
```

By modifying the code in this manner, we can revert to the original 480P resolution by simply commenting out the VIDEOLOOPBACKYUV_720P definition.

Modify the sensor (image sensor input) specific code to support 720P resolution:

4. In the Project Explorer window,
   Double-click on the **Sensor.c** file
5. Make sure that the source file contains the following header directive

```
#include "VideoLoopbackYUV.h"
```

6. Search for the “480” keyword to find the 480P specific code, then modify it to support 720P
7. You should find one occurrence, which should be modified as follows:

```
#if defined(VIDEOLOOPBACKYUV_720P)
   if((eResult = adi_mt9m114_ConfigImage(hMTM114Dev,
                 ADI_MT9M114_RegConf_720p,
                 ADI_MT9M114_IMAGE_CFG_ARRAY_SIZE))
                              != ADI_MT9M114_SUCCESS)
#else
   if((eResult = adi_mt9m114_ConfigImage(hMTM114Dev,
                 ADI_MT9M114_RegConf_480p,
                 ADI_MT9M114_IMAGE_CFG_ARRAY_SIZE))
                              != ADI_MT9M114_SUCCESS)
#endif
```

8. Save and Close the Sensor.c file

Modify the encoder (HDMI output) specific code to support 720P resolution:

9. In the Project Explorer window,
   Double-click on the **encoder.h** file
10. Add the following header directive

```
#include "VideoLoopbackYUV.h"
```

11. Search for the “480” keyword to find the 480P specific code, then modify it to support 720P
12. You should find one occurrence, which should be modified as follows:

```
#if defined(VIDEOLOOPBACKYUV_720P)
   #define ENCODER_FRAME_WIDTH            1280u
   #define ENCODER_FRAME_HEIGHT     720u
#else
   #define ENCODER_FRAME_WIDTH            720u
   #define ENCODER_FRAME_HEIGHT           480u
#endif
```

13. Save and Close the encoder.h file
14. In the Project Explorer window,
    Double-click on the **encoder.c** file
15. Search for the “480” keyword to find the 480P specific code, then modify it to support 720P
16. You should find one occurrence, which should be modified as follows:

```
#if defined(VIDEOLOOPBACKYUV_720P)
   if ( adi_adv7511_ConfigDisplayMode( hVideoOutDriver,
       ADI_ADV7511_720P_YCBCR_60HZ) != ADI_ADV7511_SUCCESS)
#else
   if ( adi_adv7511_ConfigDisplayMode( hVideoOutDriver,
       ADI_ADV7511_480P_YCBCR_60HZ) != ADI_ADV7511_SUCCESS)
#endif
```

17. Save and Close the encoder.c file

Now modify the FinBoard initialization code to configure the clock synthesizer according to the desired video output resolution.

18. In the Project Explorer window,
    Double-click on the **VideoLoopbackYUV.c** file
19. Modify the FinBoard specific initialization code as follows:

```
#if defined(VIDEOLOOPBACKYUV_720P)
   FINBOARD_CLK_Synth_Config_OUT4_74_25_MHz();
#else
   FINBOARD_CLK_Synth_Config_OUT4_27_00_MHz();
#endif
```

20. Save and Close the VideoLoopbackYUV.c file

Build the project.

21. In the Project Explorer window,
    Right-click on the VideoLoopbackYUV project and select **Build Project**.

After rebuilding a binary, CCES will detect the new binary and offer to reload the program.

Execute the project on hardware. You should see the following output in the Console:

```
Loading application: "C:\...\Debug\VideoLoopbackYUV.dxe"
Configuring Clock Synthesizer OUT4 for 720P60 resolution
LED Drivers Initialization
      LED Driver #1 Initialization
            STATUS1 = 0x00
      LED Driver #2 Initialization
            STATUS1 = 0x00
Configuring ADV7511 for 16bit YCbCr Mode
All done
```

This time, your monitor should report 720P60 or “1280x720 @ 60Hz” resolution.

Solution Archive

The video pass-through tutorial includes a solution archive. This section describes how to open this solution archive.

Extract the **VideoLoopbackYUV_solution.zip** archive somewhere on your computer and take note of the location of the solution directory :

{path to solution directory}\VideoLoopbackYUV

In a new workspace, import the project from the solution directory

1. From the menu, select **File → Import**
2. Expand the **General** section, then select **Existing Projects into Workspace**
3. Click on the **Browse** button next to **Select root directory**
4. Select the following directory

   {path to solution directory}\VideoLoopbackYUV
5. Select **VideoLoopbackYUV**
6. Select **Copy projects into workspace**
7. Click **Finish**

References

FinBoard

All documentation and support for FinBoard are located at the product website www.FinBoard.org. There you will find the following items:

- Design tutorials
- BDTI Dice Dot Counting Demo and Reference Design Software User's Guide
- Hardware User's Guide
- Hardware schematics & bill of materials
- Technical support forum

Analog Devices

For more information on the Analog Devices parts, please visit the following resources:

- BF609 – Blackfin Dual-Core Processor – http://www.analog.com/bf609
- ADV7511 – HDMI Transmitter – http://www.analog.com/adv7511

Aptina

For more information on the Aptina image sensor, please visit the following resources:

- MT9M114 : 1.3MP/ 720pHD 1/6-Inch SOC Image Sensor
  http://www.aptina.com/products/soc/mt9m114/

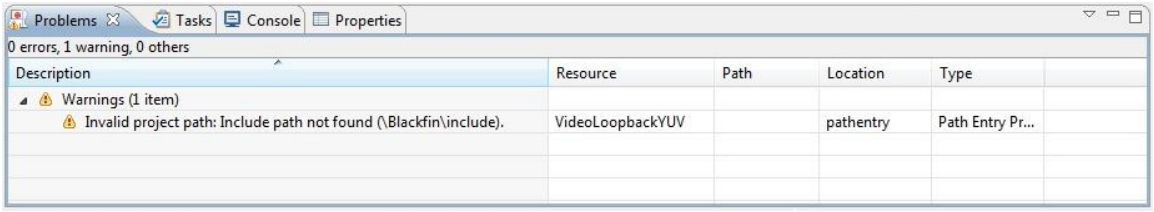
Known Issues and Limitations

The following issues are known to exist. When applicable, the workaround used is described.

VideoLoopbackYUV – Include path not found (\Blackfin\include).

When building the original VideoLoopbackYUV project, you may get the following warning, visible in the Problems tab:

Invalid project path: Include path not found (\Blackfin\include).



The (\Blackfin\include) is incomplete, which is the result of an incorrect environment variable, which can be fixed in the project settings:

1. In the CCES Project Explorer, select the **VideoLoopbackYUV** project
2. Right-click and select **Properties**.
3. Expand the **C/C++ Build** section, then select the **Settings** selection.
4. In the **CrossCore Blackfin C/C++ Compiler** section, select the **Preprocessor** selection.
5. Look at the Additional include directories (-I), which are correct

   "${workspace_loc:/${ProjName}/system}"

   "${COM_ANALOG_CROSSCORE_ADDINS_VIDEO_ENCODER_EI3_1_0_1_LOC}/Blackfin/include"

   "${COM_ANALOG_CROSSCORE_ADDINS_CAMERA_EI3_1_0_1_LOC}/Blackfin/include"
6. Now, in the **CrossCore Blackfin Assembler** section, select the **Preprocessor** selection.
7. Look at the Additional include directories (-I), which has one error:

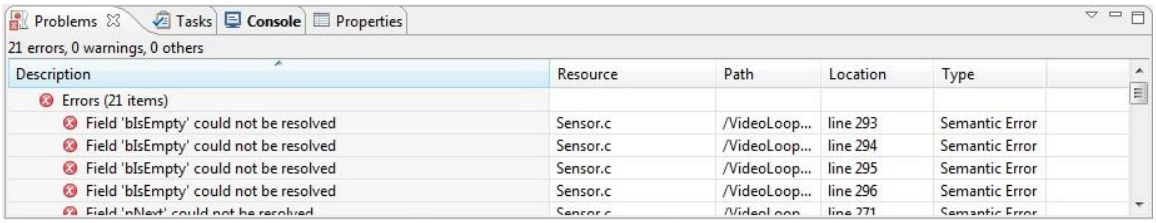
   "${workspace_loc:/${ProjName}/system}"

   "${COM_ANALOG_CROSSCORE_ADDINS_VIDEO_ENCODER_EI3_1_0_1_LOC}/Blackfin/include"

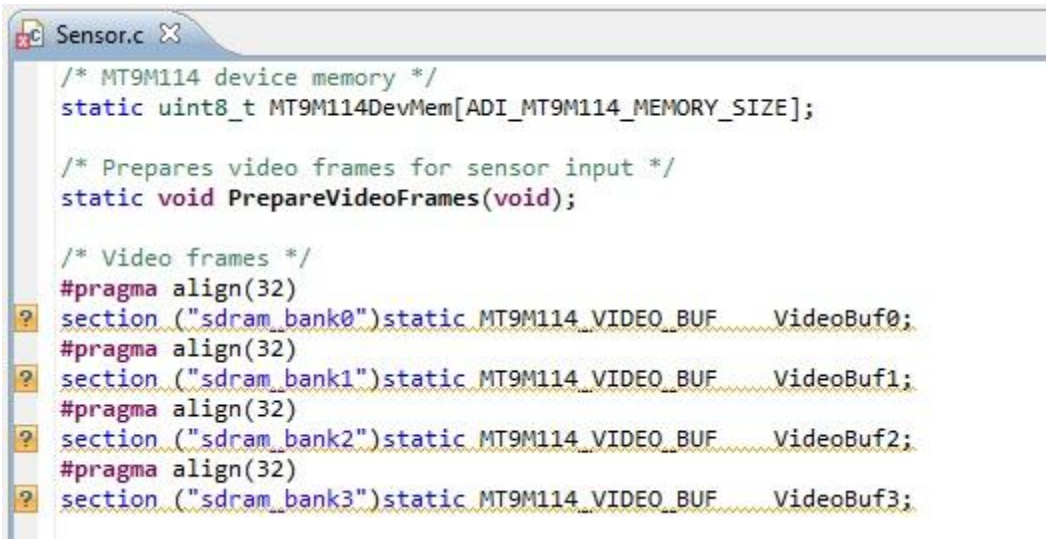
   "${**COM_ANALOG_CROSSCORE_ADDINS_CAMERA_EI3_1_0_0_LOC**}/Blackfin/include"
8. Click on the Edit button and fix the environment variable for the CAMERA_EI3 adding.
9. Click **OK** when done.
10. Click **OK**.
11. Right-click and select **Build Project**

VideoLoopbackYUV – sensor.c semantic errors

When the sensor.c source file is opened for editing, you will see semantic errors, visible in the Problems tab:



These semantic errors are related to the VideoBuf0-3 declarations.



To fix this issue, re-write the VideoBuf0 – VideoBuf3 declarations as follows:

```
/* Video frames */
#pragma align(32)
#pragma section ("sdram_bank0")
static MT9M114_VIDEO_BUF    VideoBuf0;
#pragma align(32)
#pragma section ("sdram_bank1")
static MT9M114_VIDEO_BUF    VideoBuf1;
#pragma align(32)
#pragma  section ("sdram_bank2")
static MT9M114_VIDEO_BUF    VideoBuf2;
#pragma align(32)
#pragma  section ("sdram_bank3")
static MT9M114_VIDEO_BUF    VideoBuf3;
```

VideoLoopbackYUV – example project is not copied to user workspace

Usually, the example projects get copied to the user workspace. This prevents the user from modifying the original example project.

For the VideoLoopbackYUV, this is not the case. Any modifications to the project will modify the original source.

One workaround is to import the project

1. From the menu, select **File => Import**
2. Expand the **General** section, then select **Existing Projects into Workspace**
3. Click on the **Browse** button next to **Select root directory**
4. Select the following directory

   C:\Analog Devices\Camera_EI3_Extender_Board-Rel1.0.1\Blackfin\Examples\ADSP-BF609\MT9M114\VideoLoopbackYUV
5. Select **VideoLoopbackYUV**
6. Select **Copy projects into workspace**
7. Click **Finish**

Troubleshooting

VideoLoopbackYUV – could not open source file “adi_mt9m114.h”

When the VideoLoopbackYUV is built the first time, the following error will sometimes occur:

```
… could not open source file "adi_mt9m114.h"
#include "adi_mt9m114.h"
                          ^
1 catastrophic error detected in the compilation of "…\Sensor.c"
```

The solution to this issue is to simply clean the project, then re-build as follows:

1. In the CCES Project Explorer, select the **VideoLoopbackYUV** project
2. Right-click and select **Clean Project**.
3. Right-click and select **Build Project**

.