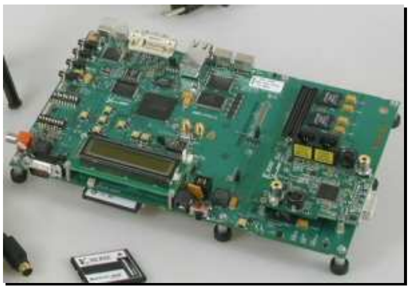
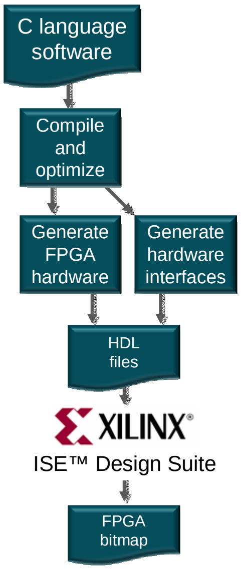
Impulse Embedded Processing Video Lab

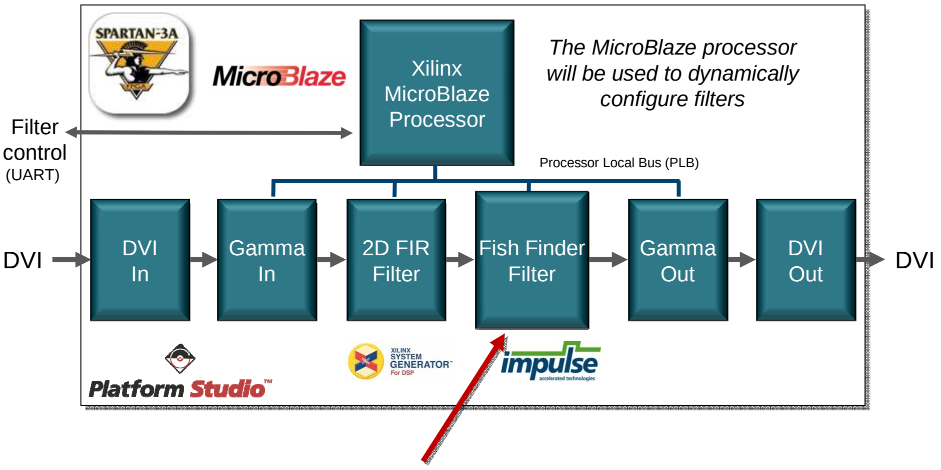




Workshop Agenda

- **Step-By-Step Creation of a Streaming Video Application:**
  - You will learn how to:
    - Use Xilinx reference hardware and software for fast development
    - Combine multiple streaming video filters in a single application
    - Use Xilinx Platform Studio (XPS) for system integration
    - Use an embedded MicroBlaze processor for video control
    - Combine multiple methods of design into a single project
  - Steps:
    - Start with a ready-to-use video reference design
    - Add a custom object detection and highlighting filter to the video stream
    - Control and configure the filter using an embedded MicroBlaze processor
    - Rebuild the project and test the enhanced video processing design

Video Design Overview



The MicroBlaze processor will be used to dynamically configure filters

Custom C-language filter

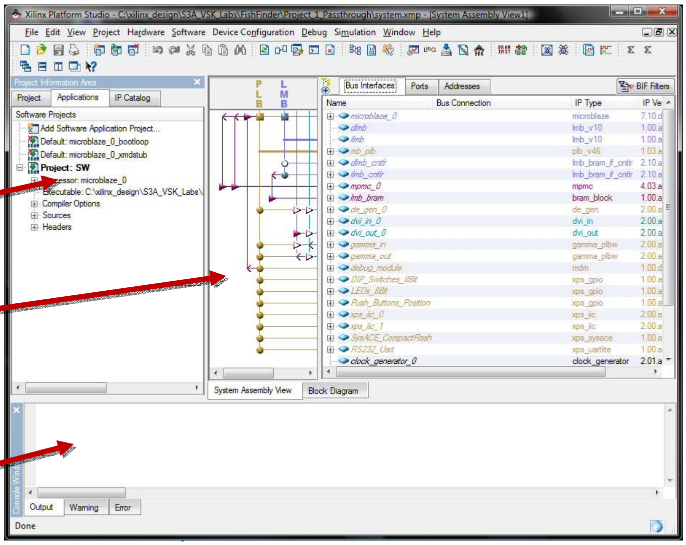
Video Starter Kit Hardware

Test the Pass-through Example Using Flash

- **Setup the Video Components**
  - 720p resolution video source
    - TViX player, laptop computer or other DVI/HDMI source
  - Xilinx Video Starter Kit
    - Spartan 3 Edition used for this workshop
  - Video monitor
    - Supporting 1280 x 720 resolution
- **Power Up and test the Video**
  - Start video source (“play”)
  - Xilinx reference designs will boot from Flash card
  - Press center push-button to load the DVI pass-through example
  - Verify video is displayed on monitor

Load Project_2_Passthrough_Completed

Xilinx Platform Studio™


Xilinx Platform Studio - C:\xilinx_design\S3A_VSK_Labs\FishFinder\Project_1_Passthrough\system.xmp - [System Assembly View1]
File Edit View Project Hardware Software Device Configuration Debug Simulation Window Help
Project Information Area
Project
Applications
IP Catalog
Software Projects
Add Software Application Project...
Default: microblaze_0_bootloop
Default: microblaze_0_xmdstub
Project: SW
Processor: microblaze_0
Executable: C:\xilinx_design\S3A_VSK_Labs\
Compiler Options
Sources
Headers
PLB
LMB
Bus Interfaces
Ports
Addresses
BIF Filters
System Assembly View
Block Diagram
Output
Warning
Error
Done

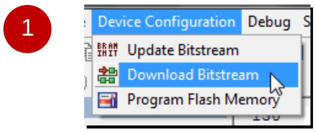
Name	Bus Connection	IP Type	IP Ve
microblaze_0		microblaze	7.10.d
dlmb		lmb_v10	1.00.a
ilmb		lmb_v10	1.00.a
mb_plb		plb_v46	1.03.a
dlmb_cntlr		lmb_bram_if_cntlr	2.10.a
ilmb_cntlr		lmb_bram_if_cntlr	2.10.a
mpmc_0		mpmc	4.03.a
lmb_bram		bram_block	1.00.a
de_gen_0		de_gen	2.00.a
dvi_in_0		dvi_in	2.00.a
dvi_out_0		dvi_out	2.00.a
gamma_in		gamma_plbw	2.00.a
gamma_out		gamma_plbw	2.00.a
debug_module		mdm	1.00.d
DIP_Switches_8Bit		xps_gpio	1.00.a
LEDs_8Bit		xps_gpio	1.00.a
Push_Buttons_Position		xps_gpio	1.00.a
xps_iic_0		xps_iic	2.00.a
xps_iic_1		xps_iic	2.00.a
SysACE_CompactFlash		xps_sysace	1.00.a
RS232_Uart		xps_uartlite	1.00.a
clock_generator_0		clock_generator	2.01.a

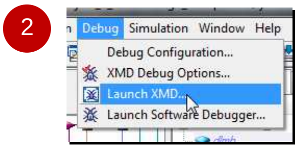
Project Information Area

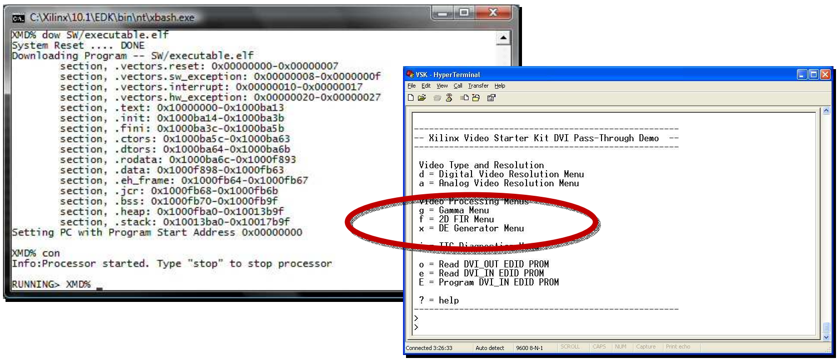
System Assembly View

Console Window

Test the Pass-Through Example







C:\Xilinx\10.1\EDK\bin\nt\xbash.exe

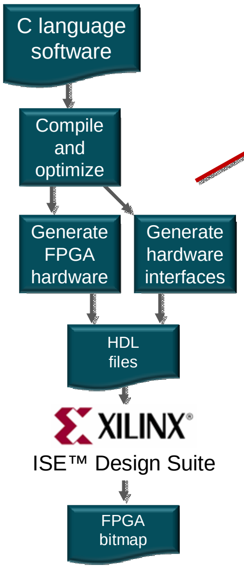
```
XMD% dow SW/executable.elf
System Reset .... DONE
Downloading Program -- SW/executable.elf
        section, .vectors.reset: 0x00000000-0x00000007
        section, .vectors.sw_exception: 0x00000008-0x0000000f
        section, .vectors.interrupt: 0x00000010-0x00000017
        section, .vectors.hw_exception: 0x00000020-0x00000027
        section, .text: 0x10000000-0x1000ba13
        section, .init: 0x1000ba14-0x1000ba3b
        section, .fini: 0x1000ba3c-0x1000ba5b
        section, .ctors: 0x1000ba5c-0x1000ba63
        section, .dtors: 0x1000ba64-0x1000ba6b
        section, .rodata: 0x1000ba6c-0x1000f893
        section, .data: 0x1000f898-0x1000fb63
        section, .eh_frame: 0x1000fb64-0x1000fb67
        section, .jcr: 0x1000fb68-0x1000fb6b
        section, .bss: 0x1000fb70-0x1000fb9f
        section, .heap: 0x1000fba0-0x10013b9f
        section, .stack: 0x10013ba0-0x10017b9f
Setting PC with Program Start Address 0x00000000

XMD% con
Info:Processor started. Type "stop" to stop processor

RUNNING> XMD% _
```

VSK - HyperTerminal

```
----------------------------------------------------
-- Xilinx Video Starter Kit DVI Pass-Through Demo  --
----------------------------------------------------

 Video Type and Resolution
 d = Digital Video Resolution Menu
 a = Analog Video Resolution Menu

 Video Processing Menus
 g = Gamma Menu
 f = 2D FIR Menu
 x = DE Generator Menu

 i = IIC Diagnostics Menu

 o = Read DVI_OUT EDID PROM
 e = Read DVI_IN EDID PROM
 E = Program DVI_IN EDID PROM

 ? = help
----------------------------------------------------
>
>
```

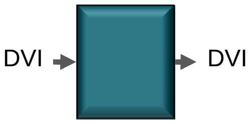
Connected 3:26:33 | Auto detect | 9600 8-N-1

Examining the Impulse C Code



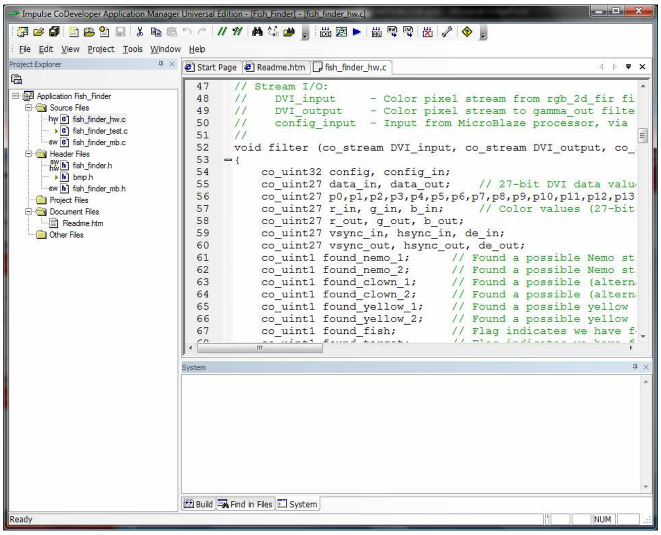
The Impulse C-to-FPGA compiler generates parallel FPGA hardware from sequential software… C-language in, HDL out!

The compiler also generates hardware interfaces such as DVI for video and PLB for processor I/O



Compiler “export” scripts are used to generate auxiliary files as needed by Xilinx ISE Design Suite

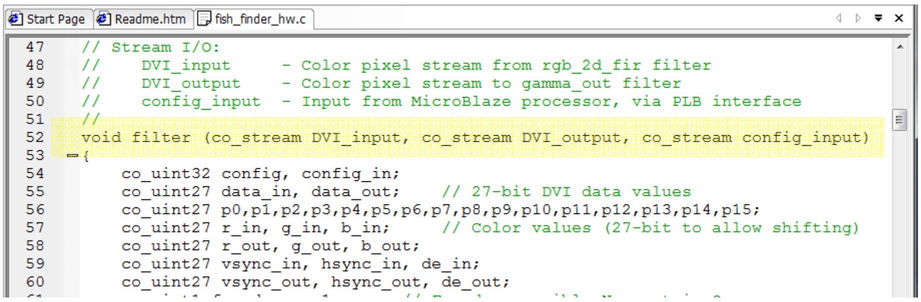
Open the Fish Finder Impulse C Project



```
// Stream I/O:
//     DVI_input     - Color pixel stream from rgb_2d_fir fi
//     DVI_output    - Color pixel stream to gamma_out filte
//     config_input  - Input from MicroBlaze processor, via
//
void filter (co_stream DVI_input, co_stream DVI_output, co_
{
        co_uint32 config, config_in;
        co_uint27 data_in, data_out;    // 27-bit DVI data valu
        co_uint27 p0,p1,p2,p3,p4,p5,p6,p7,p8,p9,p10,p11,p12,p13
        co_uint27 r_in, g_in, b_in;     // Color values (27-bit
        co_uint27 r_out, g_out, b_out;
        co_uint27 vsync_in, hsync_in, de_in;
        co_uint27 vsync_out, hsync_out, de_out;
        co_uint1 found_nemo_1;        // Found a possible Nemo st
        co_uint1 found_nemo_2;        // Found a possible Nemo st
        co_uint1 found_clown_1;       // Found a possible (altern
        co_uint1 found_clown_2;       // Found a possible (altern
        co_uint1 found_yellow_1;      // Found a possible yellow
        co_uint1 found_yellow_2;      // Found a possible yellow
        co_uint1 found_fish;          // Flag indicates we have f
```

Fish Finder C-Code Design Review

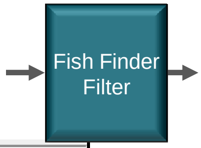
The Fish Finder algorithm is described using a C-language subroutine with streaming I/O interfaces…

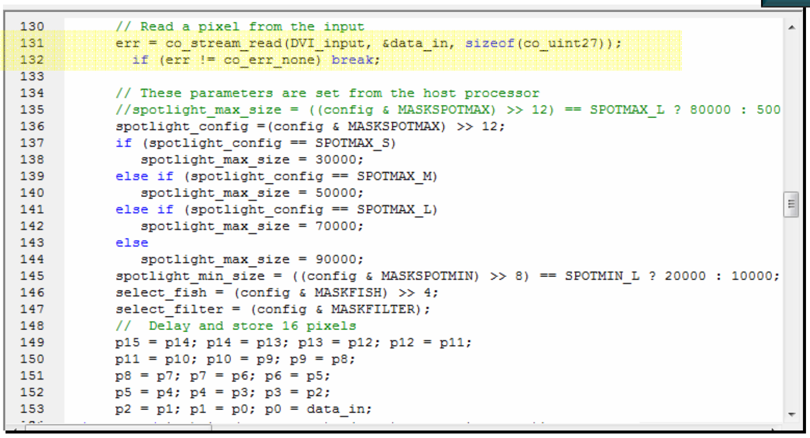


```
// Stream I/O:
//     DVI_input     - Color pixel stream from rgb_2d_fir filter
//     DVI_output    - Color pixel stream to gamma_out filter
//     config_input  - Input from MicroBlaze processor, via PLB interface
//
void filter (co_stream DVI_input, co_stream DVI_output, co_stream config_input)
{
        co_uint32 config, config_in;
        co_uint27 data_in, data_out;    // 27-bit DVI data values
        co_uint27 p0,p1,p2,p3,p4,p5,p6,p7,p8,p9,p10,p11,p12,p13,p14,p15;
        co_uint27 r_in, g_in, b_in;     // Color values (27-bit to allow shifting)
        co_uint27 r_out, g_out, b_out;
        co_uint27 vsync_in, hsync_in, de_in;
        co_uint27 vsync_out, hsync_out, de_out;
```

Fish Finder C-Code Design Review

Video I/O is described using
Impulse C types and functions:

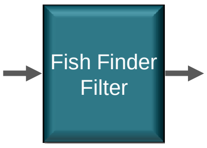




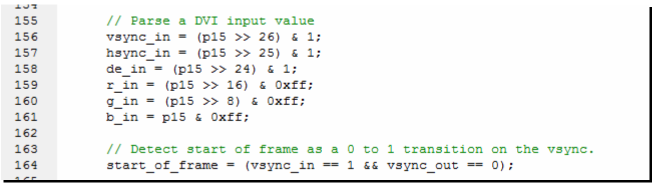
```
// Read a pixel from the input
err = co_stream_read(DVI_input, &data_in, sizeof(co_uint27));
  if (err != co_err_none) break;

// These parameters are set from the host processor
//spotlight_max_size = ((config & MASKSPOTMAX) >> 12) == SPOTMAX_L ? 80000 : 500
spotlight_config =(config & MASKSPOTMAX) >> 12;
if (spotlight_config == SPOTMAX_S)
   spotlight_max_size = 30000;
else if (spotlight_config == SPOTMAX_M)
   spotlight_max_size = 50000;
else if (spotlight_config == SPOTMAX_L)
   spotlight_max_size = 70000;
else
   spotlight_max_size = 90000;
spotlight_min_size = ((config & MASKSPOTMIN) >> 8) == SPOTMIN_L ? 20000 : 10000;
select_fish = (config & MASKFISH) >> 4;
select_filter = (config & MASKFILTER);
//  Delay and store 16 pixels
p15 = p14; p14 = p13; p13 = p12; p12 = p11;
p11 = p10; p10 = p9; p9 = p8;
p8 = p7; p7 = p6; p6 = p5;
p5 = p4; p4 = p3; p3 = p2;
p2 = p1; p1 = p0; p0 = data_in;
```

Fish Finder C-Code Design Review

DVI streaming video is represented
as 27-bit integer data (24-bits of color,
vsync, hsync and de):

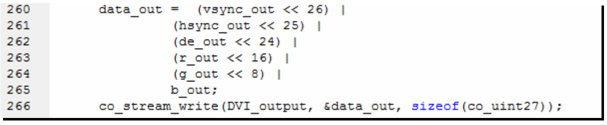


Unpacking a 27-bit video pixel and looking for start-of-frame…



```
// Parse a DVI input value
vsync_in = (p15 >> 26) & 1;
hsync_in = (p15 >> 25) & 1;
de_in = (p15 >> 24) & 1;
r_in = (p15 >> 16) & 0xff;
g_in = (p15 >> 8) & 0xff;
b_in = p15 & 0xff;

// Detect start of frame as a 0 to 1 transition on the vsync.
start_of_frame = (vsync_in == 1 && vsync_out == 0);
```

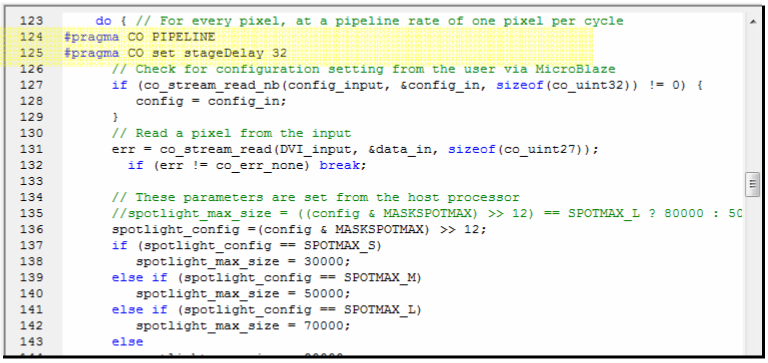
Packing and writing
a filtered pixel…



```
data_out =  (vsync_out << 26) |
         (hsync_out << 25) |
         (de_out << 24) |
         (r_out << 16) |
         (g_out << 8) |
        b_out;
co_stream_write(DVI_output, &data_out, sizeof(co_uint27));
```

Fish Finder C-Code Design Review

Loop pipelining and pipeline stage depth are easily specified using two pragmas in the C code…

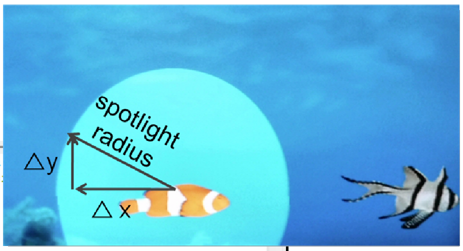


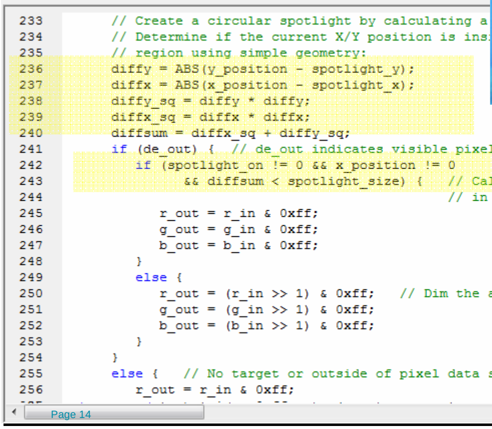
```
    do { // For every pixel, at a pipeline rate of one pixel per cycle
#pragma CO PIPELINE
#pragma CO set stageDelay 32
      // Check for configuration setting from the user via MicroBlaze
      if (co_stream_read_nb(config_input, &config_in, sizeof(co_uint32)) != 0) {
         config = config_in;
      }
      // Read a pixel from the input
      err = co_stream_read(DVI_input, &data_in, sizeof(co_uint27));
        if (err != co_err_none) break;

      // These parameters are set from the host processor
      //spotlight_max_size = ((config & MASKSPOTMAX) >> 12) == SPOTMAX_L ? 80000 : 50
      spotlight_config =(config & MASKSPOTMAX) >> 12;
      if (spotlight_config == SPOTMAX_S)
         spotlight_max_size = 30000;
      else if (spotlight_config == SPOTMAX_M)
         spotlight_max_size = 50000;
      else if (spotlight_config == SPOTMAX_L)
         spotlight_max_size = 70000;
      else
```

Automatic parallelizing of C statements enables complex, real-time processing of video signals.

Fish Finder C-Code Design Review

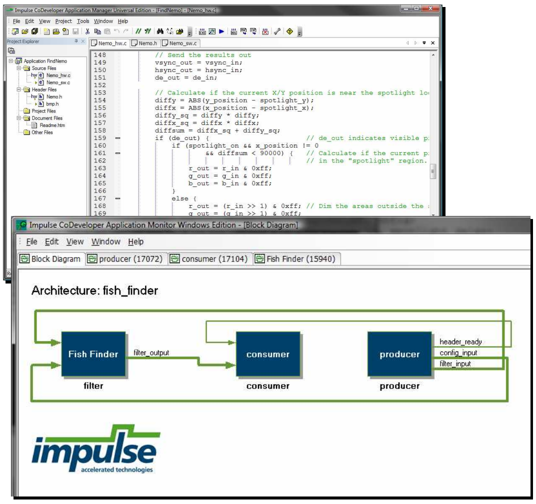
These C statements create a spotlight effect, using simple geometry to calculate the radius:





```
      // Create a circular spotlight by calculating a
      // Determine if the current X/Y position is ins
      // region using simple geometry:
      diffy = ABS(y_position - spotlight_y);
      diffx = ABS(x_position - spotlight_x);
      diffy_sq = diffy * diffy;
      diffx_sq = diffx * diffx;
      diffsum = diffx_sq + diffy_sq;
      if (de_out) {  // de_out indicates visible pixels
         if (spotlight_on != 0 && x_position != 0
               && diffsum < spotlight_size) {   // Calculate if the current pixel is
                                                // in the "spotlight" region.
            r_out = r_in & 0xff;
            g_out = g_in & 0xff;
            b_out = b_in & 0xff;
         }
         else {
            r_out = (r_in >> 1) & 0xff;   // Dim the areas outside the spotlight
            g_out = (g_in >> 1) & 0xff;
            b_out = (b_in >> 1) & 0xff;
         }
      }
      else {   // No target or outside of pixel data so just passthrough...
         r_out = r_in & 0xff;
```

Fish Finder Software Simulation

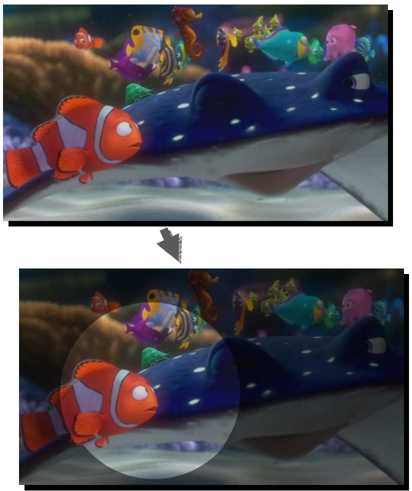


Impulse CoDeveloper Application Manager Universal Edition - [FindNemo] - [Nemo_hw.c]

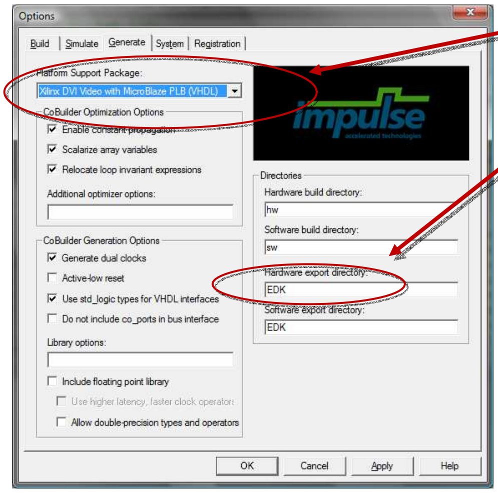
```
        // Send the results out
        vsync_out = vsync_in;
        hsync_out = hsync_in;
        de_out = de_in;

        // Calculate if the current X/Y position is near the spotlight lo
        diffy = ABS(y_position - spotlight_y);
        diffx = ABS(x_position - spotlight_x);
        diffy_sq = diffy * diffy;
        diffx_sq = diffx * diffx;
        diffsum = diffx_sq + diffy_sq;
        if (de_out) {                          // de_out indicates visible p
            if (spotlight_on && x_position != 0
                    && diffsum < 90000) {      // Calculate if the current p
                                               // in the "spotlight" region.
                r_out = r_in & 0xff;
                g_out = g_in & 0xff;
                b_out = b_in & 0xff;
            }
            else {
                r_out = (r_in >> 1) & 0xff; // Dim the areas outside the
                g_out = (g_in >> 1) & 0xff;
```

Impulse CoDeveloper Application Monitor Windows Edition - [Block Diagram]

Software simulation of a single frame…



Fish Finder Hardware Generation



Options

Platform Support Package: Xilinx DVI Video with MicroBlaze PLB (VHDL)

CoBuilder Optimization Options

☑ Enable constant propagation

☑ Scalarize array variables

☑ Relocate loop invariant expressions

Additional optimizer options:

CoBuilder Generation Options

☑ Generate dual clocks

□ Active-low reset

☑ Use std_logic types for VHDL interfaces

□ Do not include co_ports in bus interface

Library options:

□ Include floating point library

□ Use higher latency, faster clock operators

□ Allow double-precision types and operators

Directories

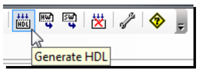
Hardware build directory: hw

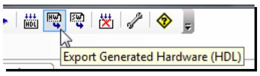
Software build directory: sw

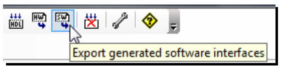
Hardware export directory: EDK

Software export directory: EDK

1. Choose a Platform Support Package (Xilinx DVI Video with PLB)
2. Specify an export directory
3. Generate hardware
4. Export hardware and software







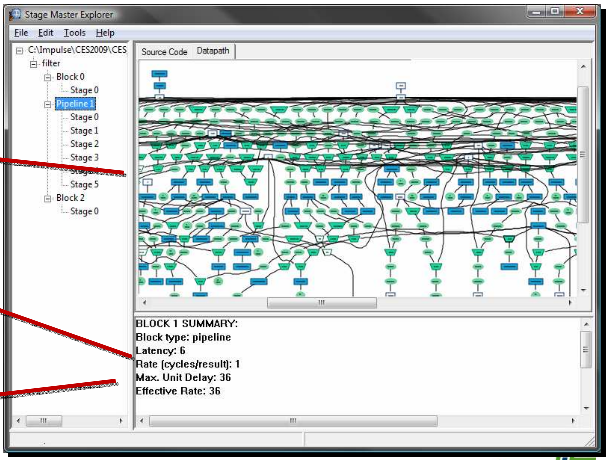
Fish Finder Hardware Optimization

Interactive pipeline optimization and analysis helps to quickly converge on the right solution for high-throughput video:

The dataflow graph shows how the C- language statements were automatically parallelized by the compiler.

The pipelining rate is critical for processing video signals at pixel-rate. A rate of 1 means a perfect video pipeline.

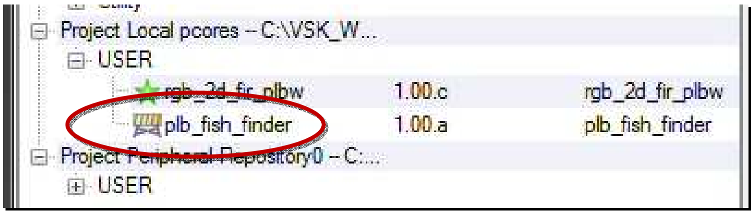
Max Unit Delay helps you to understand and control timing and clock rates.



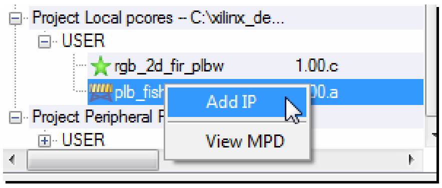
BLOCK 1 SUMMARY:
Block type: pipeline
Latency: 6
Rate (cycles/result): 1
Max. Unit Delay: 36
Effective Rate: 36

Add Fish Finder Filter to the EDK Project

After generating and exporting the hardware from the Impulse environment, we can important into Platform Studio as a pcore:

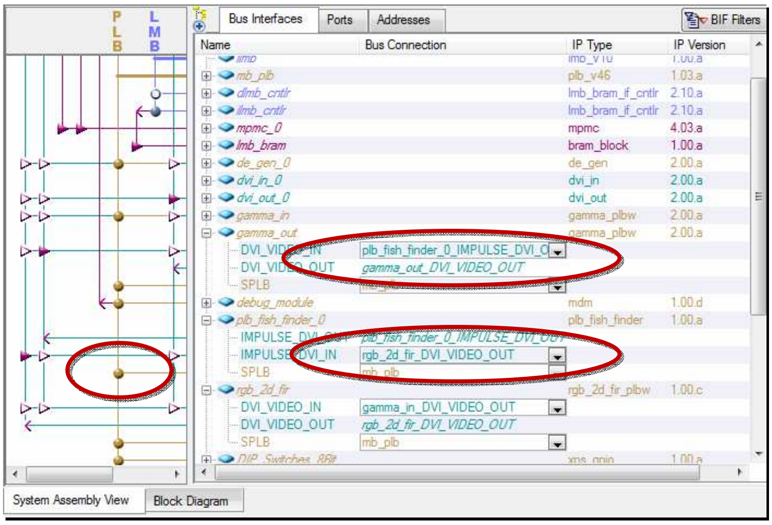


Project Local pcores – C:\VSK_W...
USER
rgb_2d_fir_plbw 1.00.c rgb_2d_fir_plbw
plb_fish_finder 1.00.a plb_fish_finder
Project Peripheral Repository0 – C:...
USER



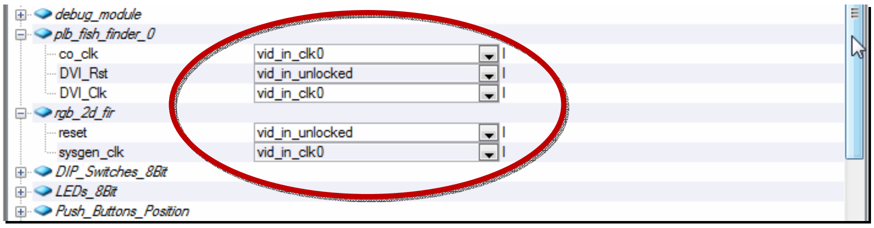
Project Local pcores – C:\xilinx_de...
USER
rgb_2d_fir_plbw 1.00.c
plb_fish 00.a
Project Peripheral F
USER

Connect Fish Finder Filter to PLB and DVI



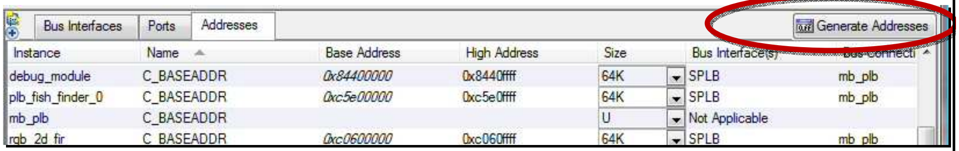
Name	Bus Connection	IP Type	IP Version
ilmb		lmb_v10	1.00.a
mb_plb		plb_v46	1.03.a
dlmb_cntlr		lmb_bram_if_cntlr	2.10.a
ilmb_cntlr		lmb_bram_if_cntlr	2.10.a
mpmc_0		mpmc	4.03.a
lmb_bram		bram_block	1.00.a
de_gen_0		de_gen	2.00.a
dvi_in_0		dvi_in	2.00.a
dvi_out_0		dvi_out	2.00.a
gamma_in		gamma_plbw	2.00.a
gamma_out		gamma_plbw	2.00.a
DVI_VIDEO_IN	plb_fish_finder_0_IMPULSE_DVI_O		
DVI_VIDEO_OUT	gamma_out_DVI_VIDEO_OUT		
SPLB	mb_plb		
debug_module		mdm	1.00.d
plb_fish_finder_0		plb_fish_finder	1.00.a
IMPULSE_DVI_OUT	plb_fish_finder_0_IMPULSE_DVI_OUT		
IMPULSE_DVI_IN	rgb_2d_fir_DVI_VIDEO_OUT		
SPLB	mb_plb		
rgb_2d_fir		rgb_2d_fir_plbw	1.00.c
DVI_VIDEO_IN	gamma_in_DVI_VIDEO_OUT		
DVI_VIDEO_OUT	rgb_2d_fir_DVI_VIDEO_OUT		
SPLB	mb_plb		
DIP_Switches_8Bit		xps_gpio	1.00.a

Connect Clocks and Reset



debug_module	
plb_fish_finder_0	
co_clk	vid_in_clk0
DVI_Rst	vid_in_unlocked
DVI_Clk	vid_in_clk0
rgb_2d_fir	
reset	vid_in_unlocked
sysgen_clk	vid_in_clk0
DIP_Switches_8Bit	
LEDs_8Bit	
Push_Buttons_Position	

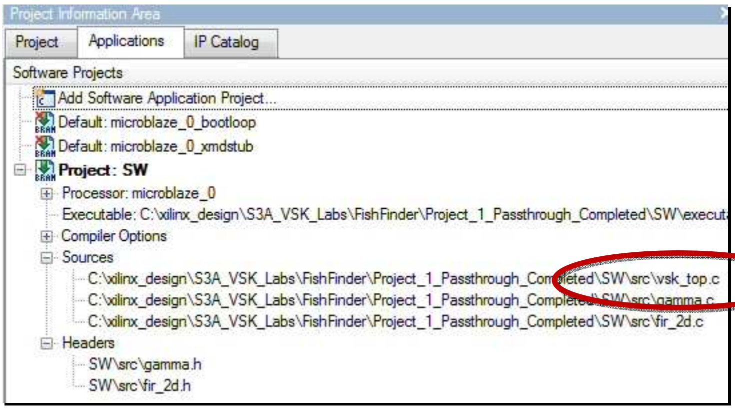
Generate Addresses



Instance	Name	Base Address	High Address	Size	Bus Interface(s)	Bus Connecti
debug_module	C_BASEADDR	0x84400000	0x8440ffff	64K	SPLB	mb_plb
plb_fish_finder_0	C_BASEADDR	0xc5e00000	0xc5e0ffff	64K	SPLB	mb_plb
mb_plb	C_BASEADDR			U	Not Applicable	
rgb_2d_fir	C_BASEADDR	0xc0600000	0xc060ffff	64K	SPLB	mb_plb

Note: addresses generated may be different than shown above

Modify the Software Application

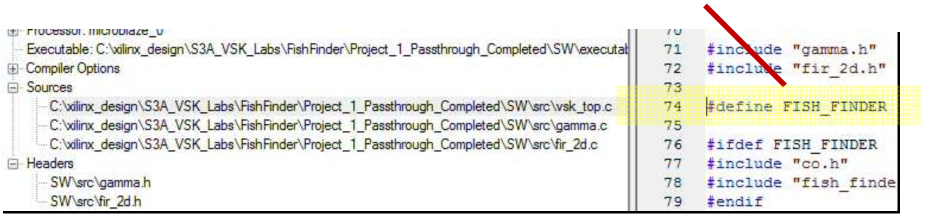


Software Projects
- Add Software Application Project...
- Default: microblaze_0_bootloop
- Default: microblaze_0_xmdstub
- **Project: SW**
  - Processor: microblaze_0
  - Executable: C:\xilinx_design\S3A_VSK_Labs\FishFinder\Project_1_Passthrough_Completed\SW\execut
  - Compiler Options
  - Sources
    - C:\xilinx_design\S3A_VSK_Labs\FishFinder\Project_1_Passthrough_Completed\SW\src\vsk_top.c
    - C:\xilinx_design\S3A_VSK_Labs\FishFinder\Project_1_Passthrough_Completed\SW\src\gamma.c
    - C:\xilinx_design\S3A_VSK_Labs\FishFinder\Project_1_Passthrough_Completed\SW\src\fir_2d.c
  - Headers
    - SW\src\gamma.h
    - SW\src\fir_2d.h

We will modify the software application to add in the new Fish Finder menu and add the related filter control code.

Modify the Software Application

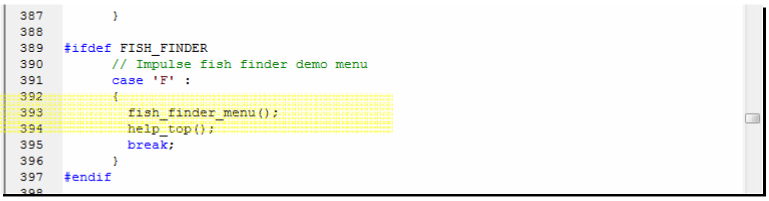
Remove the leading comment characters (//)



```
Processor: microblaze_0
Executable: C:\xilinx_design\S3A_VSK_Labs\FishFinder\Project_1_Passthrough_Completed\SW\executal
Compiler Options
Sources
    C:\xilinx_design\S3A_VSK_Labs\FishFinder\Project_1_Passthrough_Completed\SW\src\vsk_top.c
    C:\xilinx_design\S3A_VSK_Labs\FishFinder\Project_1_Passthrough_Completed\SW\src\gamma.c
    C:\xilinx_design\S3A_VSK_Labs\FishFinder\Project_1_Passthrough_Completed\SW\src\fir_2d.c
Headers
    SW\src\gamma.h
    SW\src\fir_2d.h
```

```c
#include "gamma.h"
#include "fir_2d.h"

#define FISH_FINDER

#ifdef FISH_FINDER
#include "co.h"
#include "fish_finde
#endif
```

This change will enable the new menu feature:

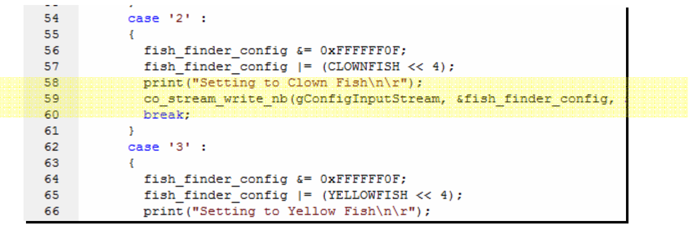


```c
        }

#ifdef FISH_FINDER
        // Impulse fish finder demo menu
        case 'F' :
        {
          fish_finder_menu();
          help_top();
          break;
        }
#endif
```

Fish Finder C-Code Design Review

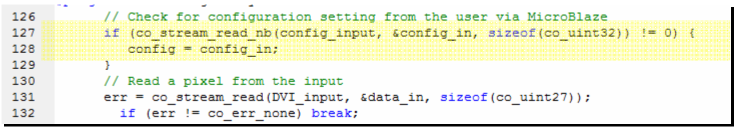
Communication between MicroBlaze and Fish Finder is described using Impulse C co_stream API functions:

Writing a configuration word from the MicroBlaze application…



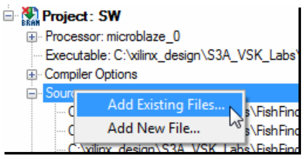
```c
        case '2' :
        {
          fish_finder_config &= 0xFFFFFF0F;
          fish_finder_config |= (CLOWNFISH << 4);
          print("Setting to Clown Fish\n\r");
          co_stream_write_nb(gConfigInputStream, &fish_finder_config,
          break;
        }
        case '3' :
        {
          fish_finder_config &= 0xFFFFFF0F;
          fish_finder_config |= (YELLOWFISH << 4);
          print("Setting to Yellow Fish\n\r");
```

Polling to read the configuration word in the hardware process…

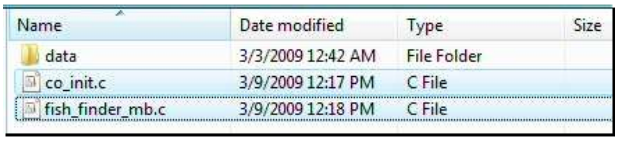


```c
      // Check for configuration setting from the user via MicroBlaze
      if (co_stream_read_nb(config_input, &config_in, sizeof(co_uint32)) != 0) {
         config = config_in;
      }
      // Read a pixel from the input
      err = co_stream_read(DVI_input, &data_in, sizeof(co_uint27));
        if (err != co_err_none) break;
```

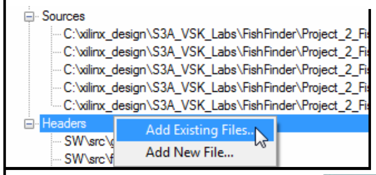
Add New Sources and Headers



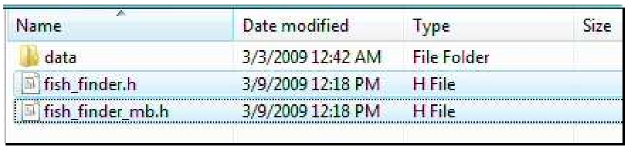
- Project: SW
  - Processor: microblaze_0
  - Executable: C:\xilinx_design\S3A_VSK_Labs\
  - Compiler Options
  - Sour
    - C \FishFind
    - C \FishFind
    - C:\xilinx_design\S3A_VSK_Labs\FishFind



Name	Date modified	Type	Size
data	3/3/2009 12:42 AM	File Folder	
co_init.c	3/9/2009 12:17 PM	C File	
fish_finder_mb.c	3/9/2009 12:18 PM	C File	

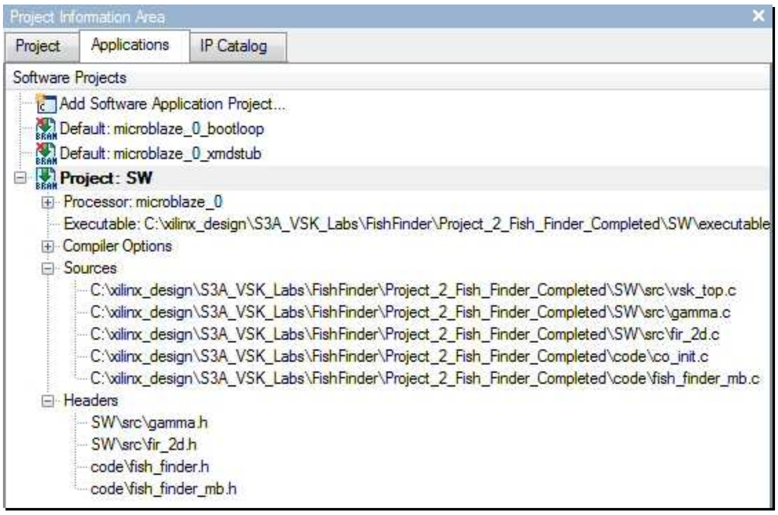


- Sources
  - C:\xilinx_design\S3A_VSK_Labs\FishFinder\Project_2_Fi
  - C:\xilinx_design\S3A_VSK_Labs\FishFinder\Project_2_Fi
  - C:\xilinx_design\S3A_VSK_Labs\FishFinder\Project_2_Fi
  - C:\xilinx_design\S3A_VSK_Labs\FishFinder\Project_2_Fi
  - C:\xilinx_design\S3A_VSK_Labs\FishFinder\Project_2_Fi
- Headers
  - SW\src\g
  - SW\src\f



Name	Date modified	Type	Size
data	3/3/2009 12:42 AM	File Folder	
fish_finder.h	3/9/2009 12:18 PM	H File	
fish_finder_mb.h	3/9/2009 12:18 PM	H File	

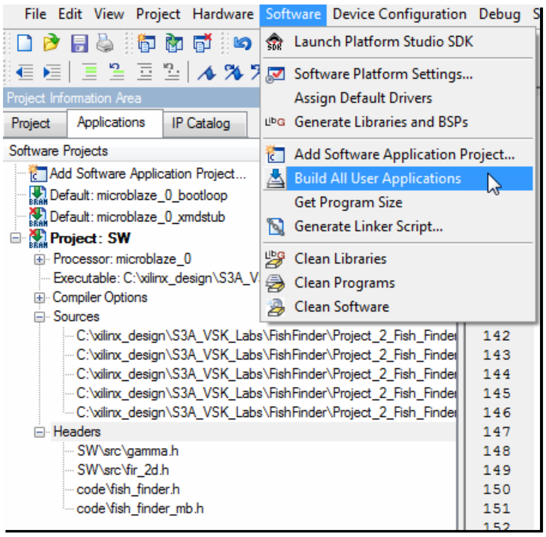
Add New Sources and Headers



Software Projects

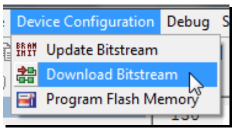
- Add Software Application Project...
- Default: microblaze_0_bootloop
- Default: microblaze_0_xmdstub
- Project: SW
  - Processor: microblaze_0
  - Executable: C:\xilinx_design\S3A_VSK_Labs\FishFinder\Project_2_Fish_Finder_Completed\SW\executable
  - Compiler Options
  - Sources
    - C:\xilinx_design\S3A_VSK_Labs\FishFinder\Project_2_Fish_Finder_Completed\SW\src\vsk_top.c
    - C:\xilinx_design\S3A_VSK_Labs\FishFinder\Project_2_Fish_Finder_Completed\SW\src\gamma.c
    - C:\xilinx_design\S3A_VSK_Labs\FishFinder\Project_2_Fish_Finder_Completed\SW\src\fir_2d.c
    - C:\xilinx_design\S3A_VSK_Labs\FishFinder\Project_2_Fish_Finder_Completed\code\co_init.c
    - C:\xilinx_design\S3A_VSK_Labs\FishFinder\Project_2_Fish_Finder_Completed\code\fish_finder_mb.c
  - Headers
    - SW\src\gamma.h
    - SW\src\fir_2d.h
    - code\fish_finder.h
    - code\fish_finder_mb.h

Build the Software Application

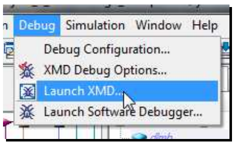

File Edit View Project Hardware Software Device Configuration Debug
Launch Platform Studio SDK
Software Platform Settings...
Assign Default Drivers
Generate Libraries and BSPs
Add Software Application Project...
Build All User Applications
Get Program Size
Generate Linker Script...
Clean Libraries
Clean Programs
Clean Software
Project Information Area
Project
Applications
IP Catalog
Software Projects
Add Software Application Project...
Default: microblaze_0_bootloop
Default: microblaze_0_xmdstub
Project: SW
Processor: microblaze_0
Executable: C:\xilinx_design\S3A_V
Compiler Options
Sources
C:\xilinx_design\S3A_VSK_Labs\FishFinder\Project_2_Fish_Finder
C:\xilinx_design\S3A_VSK_Labs\FishFinder\Project_2_Fish_Finder
C:\xilinx_design\S3A_VSK_Labs\FishFinder\Project_2_Fish_Finder
C:\xilinx_design\S3A_VSK_Labs\FishFinder\Project_2_Fish_Finder
C:\xilinx_design\S3A_VSK_Labs\FishFinder\Project_2_Fish_Finder
Headers
SW\src\gamma.h
SW\src\fir_2d.h
code\fish_finder.h
code\fish_finder_mb.h
142
143
144
145
146
147
148
149
150
151
152

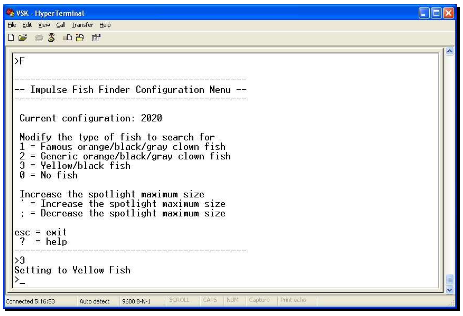
Test the Fish Finder

1

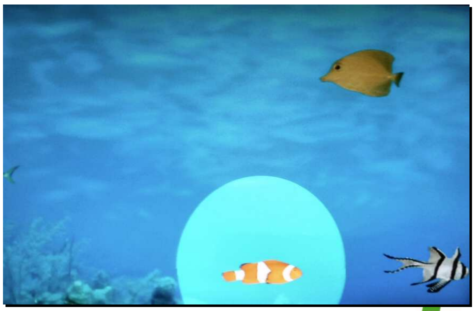


2





```
>F

---------------------------------------------
-- Impulse Fish Finder Configuration Menu --
---------------------------------------------

 Current configuration: 2020

 Modify the type of fish to search for
 1 = Famous orange/black/gray clown fish
 2 = Generic orange/black/gray clown fish
 3 = Yellow/black fish
 0 = No fish

 Increase the spotlight maximum size
 ' = Increase the spotlight maximum size
 ; = Decrease the spotlight maximum size

esc = exit
 ?  = help
---------------------------------------------
>3
Setting to Yellow Fish
>_
```



For Additional Information

www.xilinx.com/vsk_s3

www.ImpulseC.com/Tutorials/Xilinx/VSK_S3