Advanced SOC Design
Lab catapult

Jiin Lai

Catapult

High-level synthesis (HLS) creates RTL implementations from abstract specifications described with C / C++.

Tool setup

- You need to setup Catapult and Questasim in IC lab account.
  - Run “run_catapult” to check the tool is correctly installed.
- You also need to setup the directory of some environment variables to run up our sample code.
  - Run “lab1_fir” to check the environment variables is setup correctly.
  - You may need to create “bin” folder to save execution file.
- TA will give you another tool setup guide for tool setup.

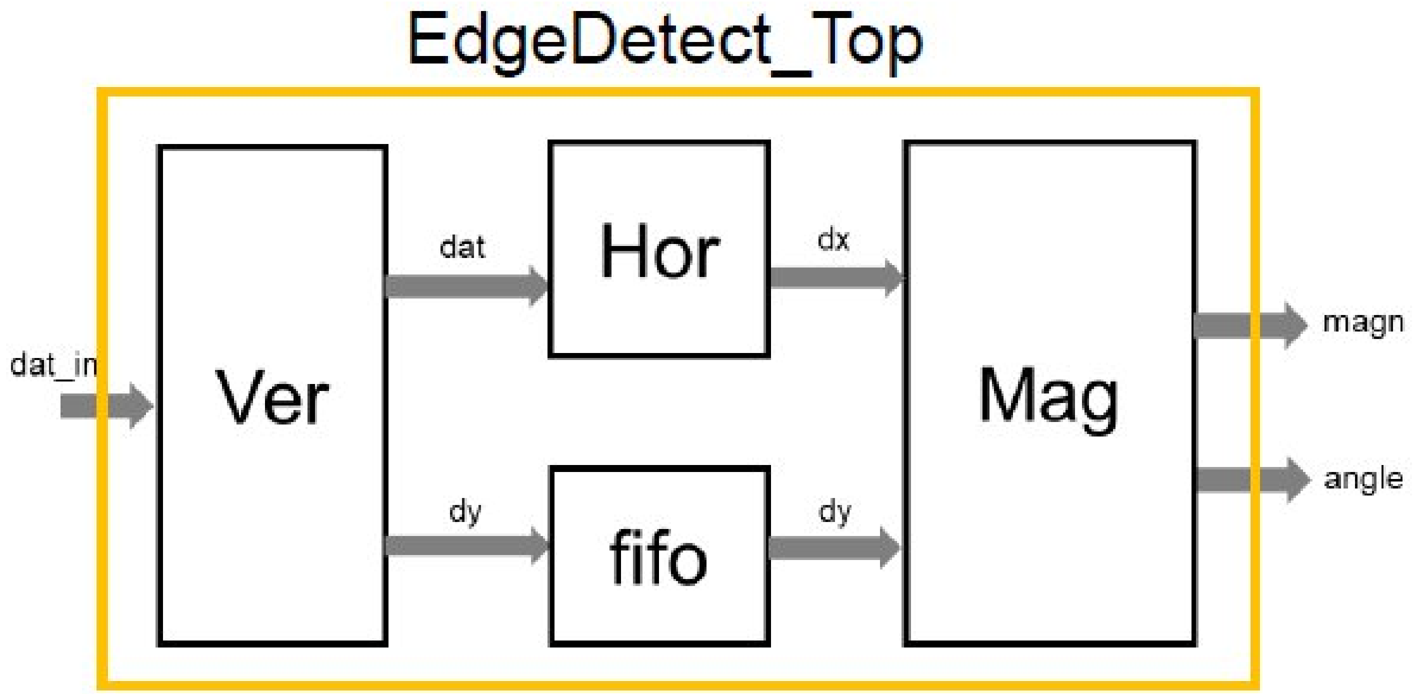
Content

- In this lab, we will provide the example code of lab1_fir and part of Lab2_edgedetect_fsic .
- Lab1_fir is for you to understand how to use catapult.
- Lab2_edgedetect_fsic needs rewrite the code to meet the spec.
  - Note: we will provide you a sample code in 01_edgedetect.

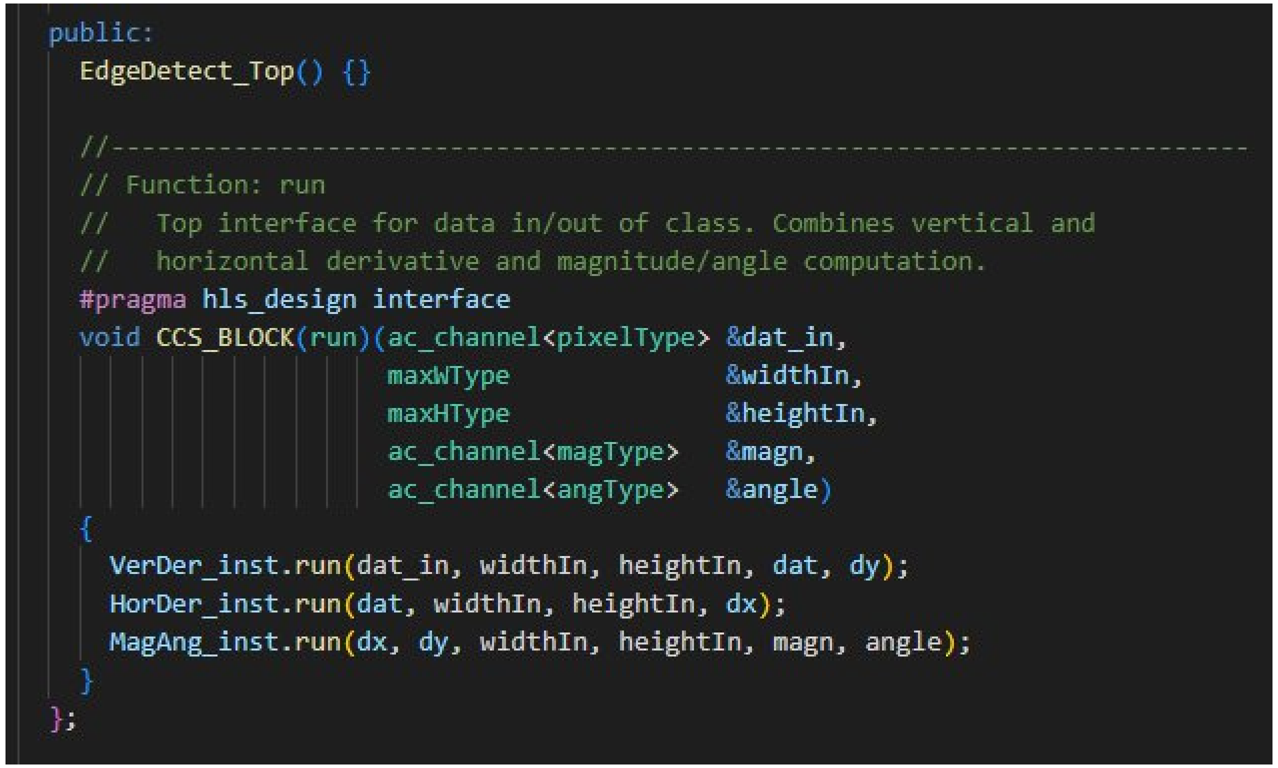
Lab1-fir work

- 3 subdirectory
  - Walkthrough
  - Memory
  - Multblk
- Work content
  - Run through the example – make
  - Study the HLS code to understand the catapult HLS

02_edgedetect block design



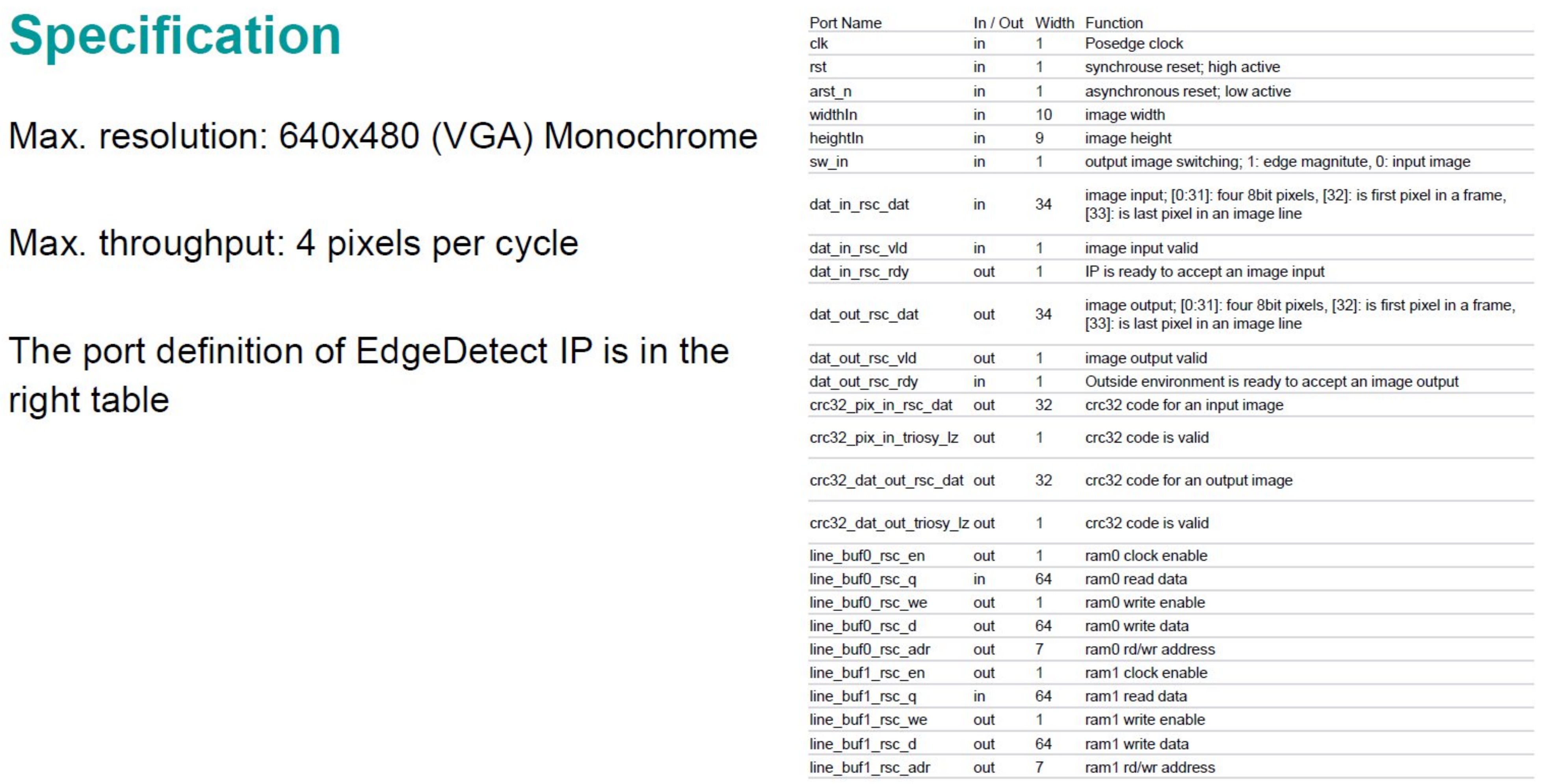
02_edgedetect



```
public:
  EdgeDetect_Top() {}

  //----------------------------------------------------------------------------
  // Function: run
  //   Top interface for data in/out of class. Combines vertical and
  //   horizontal derivative and magnitude/angle computation.
  #pragma hls_design interface
  void CCS_BLOCK(run)(ac_channel<pixelType> &dat_in,
                      maxWType               &widthIn,
                      maxHType               &heightIn,
                      ac_channel<magType>    &magn,
                      ac_channel<angType>    &angle)
  {
    VerDer_inst.run(dat_in, widthIn, heightIn, dat, dy);
    HorDer_inst.run(dat, widthIn, heightIn, dx);
    MagAng_inst.run(dx, dy, widthIn, heightIn, magn, angle);
  }
};
```

Lab work



Specification

Max. resolution: 640x480 (VGA) Monochrome

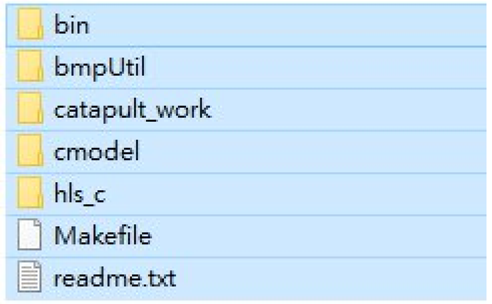
Max. throughput: 4 pixels per cycle

The port definition of EdgeDetect IP is in the right table

Port Name	In / Out	Width	Function
clk	in	1	Posedge clock
rst	in	1	synchrouse reset; high active
arst_n	in	1	asynchronous reset; low active
widthIn	in	10	image width
heightIn	in	9	image height
sw_in	in	1	output image switching; 1: edge magnitute, 0: input image
dat_in_rsc_dat	in	34	image input; [0:31]: four 8bit pixels, [32]: is first pixel in a frame, [33]: is last pixel in an image line
dat_in_rsc_vld	in	1	image input valid
dat_in_rsc_rdy	out	1	IP is ready to accept an image input
dat_out_rsc_dat	out	34	image output; [0:31]: four 8bit pixels, [32]: is first pixel in a frame, [33]: is last pixel in an image line
dat_out_rsc_vld	out	1	image output valid
dat_out_rsc_rdy	in	1	Outside environment is ready to accept an image output
crc32_pix_in_rsc_dat	out	32	crc32 code for an input image
crc32_pix_in_triosy_lz	out	1	crc32 code is valid
crc32_dat_out_rsc_dat	out	32	crc32 code for an output image
crc32_dat_out_triosy_lz	out	1	crc32 code is valid
line_buf0_rsc_en	out	1	ram0 clock enable
line_buf0_rsc_q	in	64	ram0 read data
line_buf0_rsc_we	out	1	ram0 write enable
line_buf0_rsc_d	out	64	ram0 write data
line_buf0_rsc_adr	out	7	ram0 rd/wr address
line_buf1_rsc_en	out	1	ram1 clock enable
line_buf1_rsc_q	in	64	ram1 read data
line_buf1_rsc_we	out	1	ram1 write enable
line_buf1_rsc_d	out	64	ram1 write data
line_buf1_rsc_adr	out	7	ram1 rd/wr address

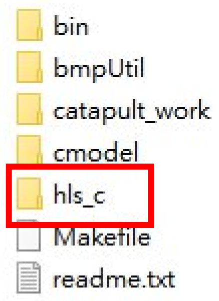
Folder Structure

- 01_edge_detect

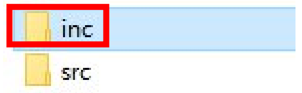


bin
bmpUtil
catapult_work
cmodel
hls_c
Makefile
readme.txt

- 02_edgedetect_fsic



bin
bmpUtil
catapult_work
cmodel
hls_c
Makefile
readme.txt

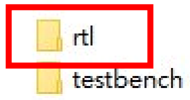


inc
src

Generate it on your own
(modify the code base on 01_edge_detect)

- 03_fsic_prj
  - Testbench is provided

dsn



rtl
testbench

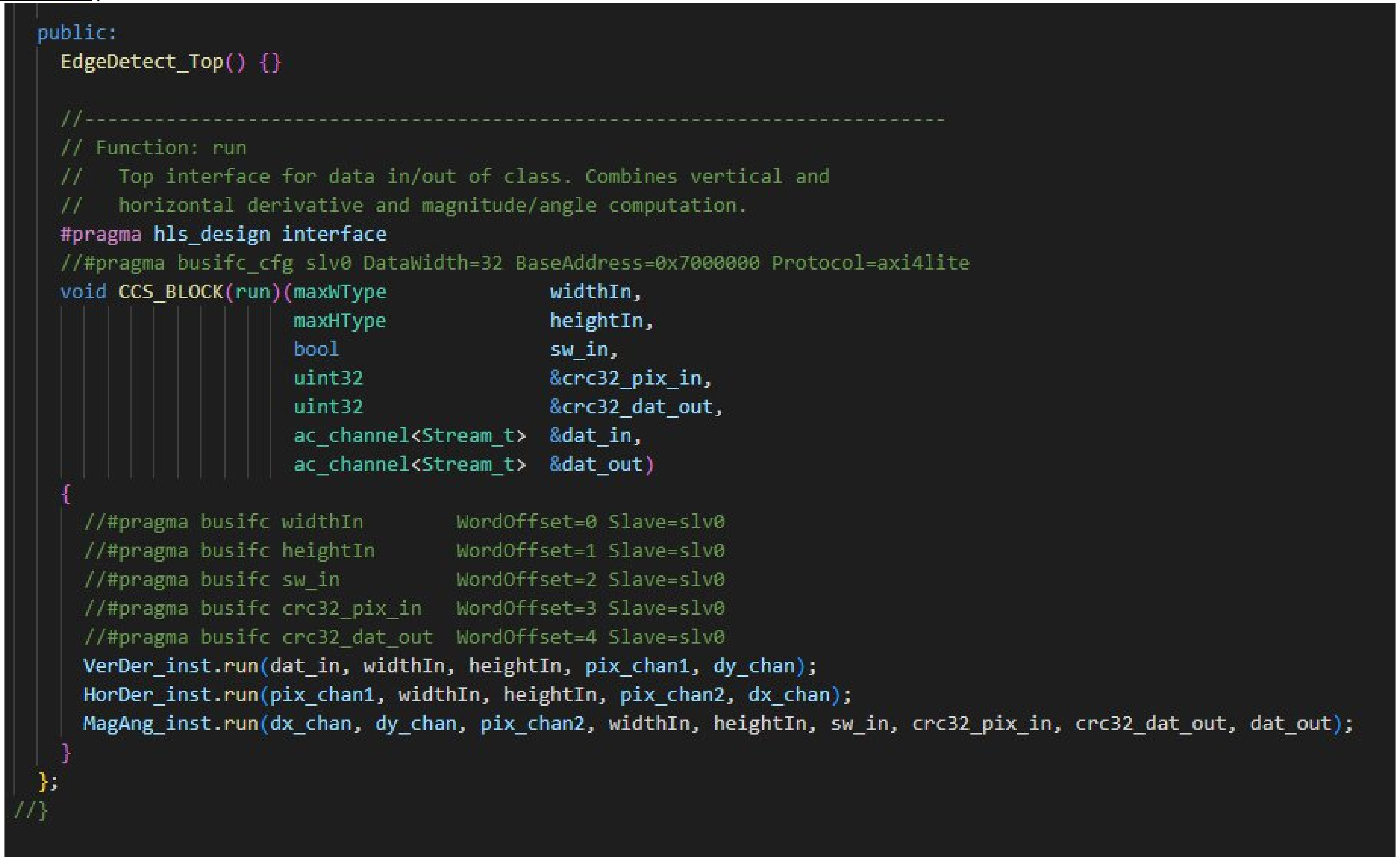
integrate to fsic

Lab work - Few Modifications of Original EdgeDetect

1. Folder:Lab1_edgedetect - Run through the example
2. Folder:Lab2_edgedetect : Based on 1-lab1_edgedect code base, modify and implement the following (under hls_c)
    1. Process four pixels per clock cycle.
       Change input data type to compress four 8-bit pixels data read in one read cycle.
    2. Use sum of absolute difference (SAD) for edge magnitude calculation.
       Originally, we use square root in sample code.
    3. Add two crc32 calculation on image input / output.
       We will provide you the crc32 code, you need to use it in your design.
    4. Select the output source from input image or the calculated magnitude.
       Add mux on the top of design.
    5. Remove the angle calculation.
3. Use path/concat_rtl.v to integrate with FSIC and run verification (refer to lab1-fsic-sim)

About four pixels per clock cycle, you can check the file “hls_bluebook.pdf” page 102.

Modified top design in 02_edgedetect_fsic (for your reference)



```
public:
  EdgeDetect_Top() {}

  //----------------------------------------------------------------------------
  // Function: run
  //   Top interface for data in/out of class. Combines vertical and
  //   horizontal derivative and magnitude/angle computation.
  #pragma hls_design interface
  //#pragma busifc_cfg slv0 DataWidth=32 BaseAddress=0x7000000 Protocol=axi4lite
  void CCS_BLOCK(run)(maxWType              widthIn,
                      maxHType              heightIn,
                      bool                  sw_in,
                      uint32                &crc32_pix_in,
                      uint32                &crc32_dat_out,
                      ac_channel<Stream_t>  &dat_in,
                      ac_channel<Stream_t>  &dat_out)
  {
    //#pragma busifc widthIn        WordOffset=0 Slave=slv0
    //#pragma busifc heightIn       WordOffset=1 Slave=slv0
    //#pragma busifc sw_in          WordOffset=2 Slave=slv0
    //#pragma busifc crc32_pix_in   WordOffset=3 Slave=slv0
    //#pragma busifc crc32_dat_out  WordOffset=4 Slave=slv0
    VerDer_inst.run(dat_in, widthIn, heightIn, pix_chan1, dy_chan);
    HorDer_inst.run(pix_chan1, widthIn, heightIn, pix_chan2, dx_chan);
    MagAng_inst.run(dx_chan, dy_chan, pix_chan2, widthIn, heightIn, sw_in, crc32_pix_in, crc32_dat_out, dat_out);
  }
};
//}
```

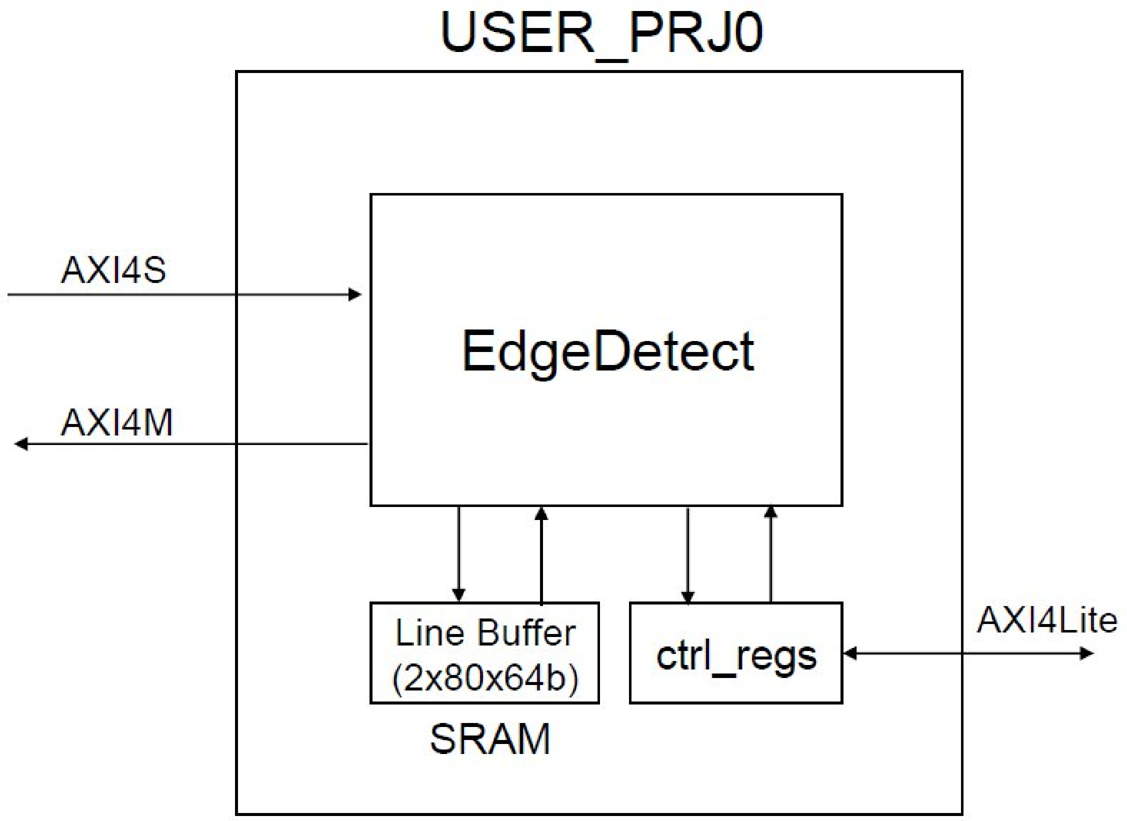
Integrate to FSIC

RTL Integration with FSIC

Copy concat_EdgeDetect_Top.v to be as concat_EdgeDetect_Top_fsic.v in 03_fsic_prj/dsn/rtl

Instantiate EdgeDetect in USER_PRJ0 in 03_fsic_prj/dsn/rtl/user_prj0.v

Also, add ctrl_regs rd/wr circuits and two SRAMs



Integrate to FSIC

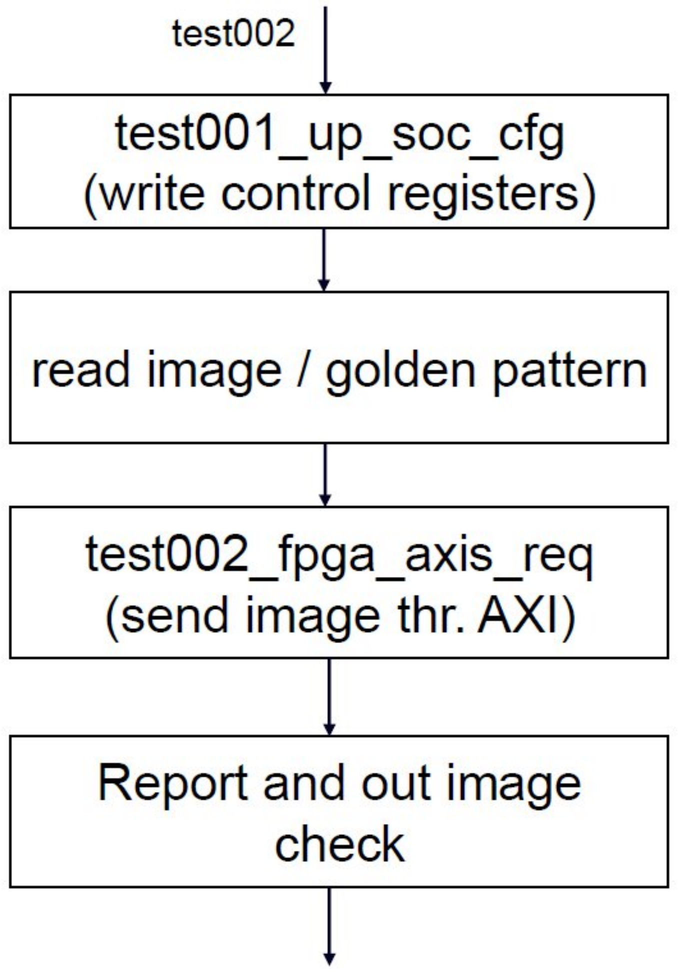
FSIC Testbench

Modify some functions in FSIC TB (tb_fsic.v) for verifying the EdgeDetect IP

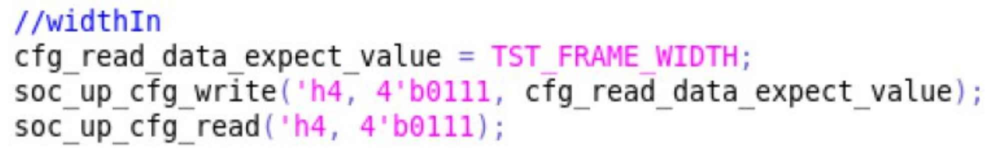
- test001_up_soc_cfg: control register read / write test
- test002, test002_fpga_axis_req, fpga_axis_req: send / receive input / output image through AXI

Image and golden pattern are put in dsn/testbench/vsim/pattern which is generated in 02_edgedetect_fsic

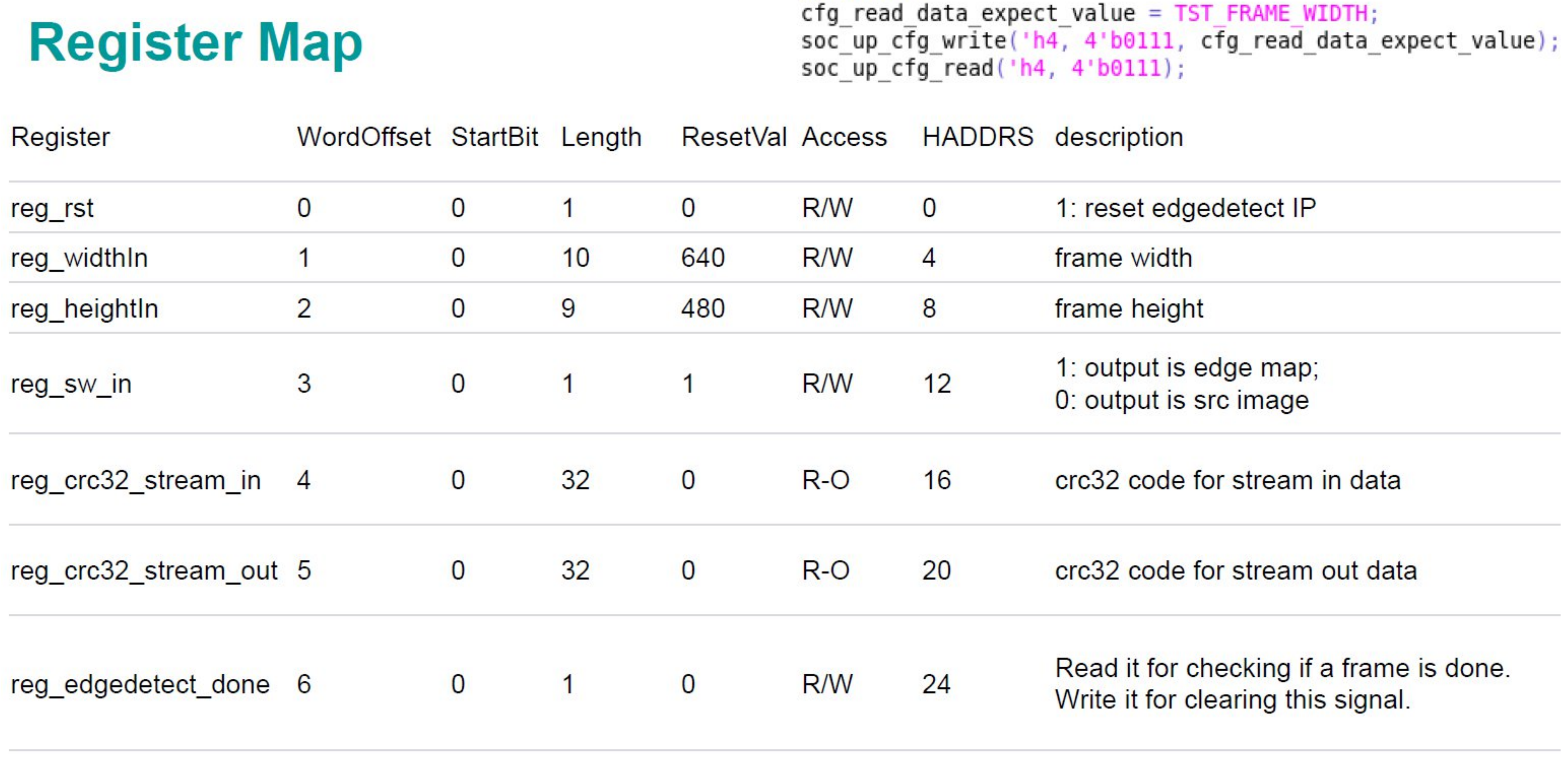
Search the keyword ‘USE_EDGEDETECT_IP’ for detail


test002
test001_up_soc_cfg (write control registers)
read image / golden pattern
test002_fpga_axis_req (send image thr. AXI)
Report and out image check

Integrate to FSIC



```
//widthIn
cfg_read_data_expect_value = TST_FRAME_WIDTH;
soc_up_cfg_write('h4, 4'b0111, cfg_read_data_expect_value);
soc_up_cfg_read('h4, 4'b0111);
```



Register Map

Register	WordOffset	StartBit	Length	ResetVal	Access	HADDRS	description
reg_rst	0	0	1	0	R/W	0	1: reset edgedetect IP
reg_widthIn	1	0	10	640	R/W	4	frame width
reg_heightIn	2	0	9	480	R/W	8	frame height
reg_sw_in	3	0	1	1	R/W	12	1: output is edge map; 0: output is src image
reg_crc32_stream_in	4	0	32	0	R-O	16	crc32 code for stream in data
reg_crc32_stream_out	5	0	32	0	R-O	20	crc32 code for stream out data
reg_edgedetect_done	6	0	1	0	R/W	24	Read it for checking if a frame is done. Write it for clearing this signal.

Integrate to FSIC

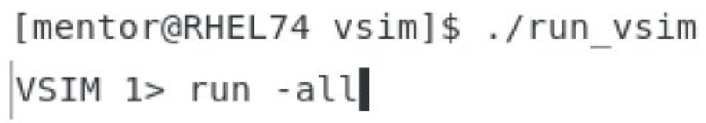
Wrap Up

Put the edgedetect rtl into dsn/rtl and *.hex test pattern into dsn/testbench/vsim/pattern

Modify the dsn/rtl/user_prj0.v and dsn/testbench/tb_fsic.v

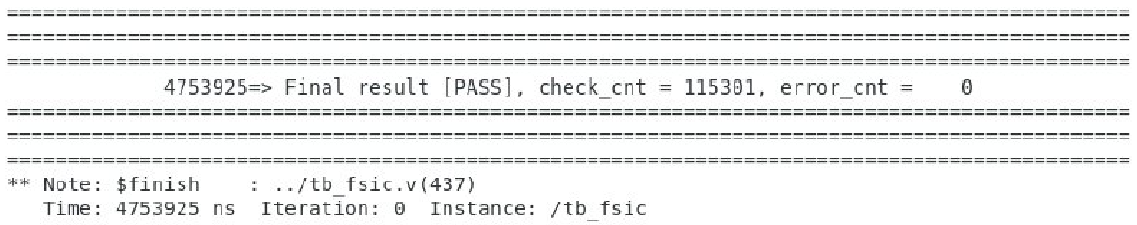
Add concat_EdgeDetect_Top_fsic.v in dsn/testbench/vsim/filelist

Run rtl simulation with the script run_vsim



```
[mentor@RHEL74 vsim]$ ./run_vsim
VSIM 1> run -all
```

Check if there is any errors during simulation



```
==========================================================================================
==========================================================================================
==========================================================================================
            4753925=> Final result [PASS], check_cnt = 115301, error_cnt =     0
==========================================================================================
==========================================================================================
==========================================================================================
** Note: $finish    : ../tb_fsic.v(437)
   Time: 4753925 ns  Iteration: 0  Instance: /tb_fsic
```

Provided Content

- All content of lab1_fir(walkthrough, mem_ifc, multi_blks)
- lab2_edgedetec_fsic
  - All content of 01_edgedetect
  - Only provide the testbench of 02_edgedetect_fsic, you need to modify the code of 01_edgedetect to satisfy specification.
  - crc32.cpp can be accessed by github, which should be used in your design

  https://github.com/bol-edu/caravel-soc_fpga-lab/tree/main/catapult_hls
  - 03_fsic_prj, after you generate the rtl design of 02_edgedetect_fsic, integrate it to FSIC and save all design into this folder.

- Note: This lab didn't need to do synthesis, only need simulation.

Lab2-1(FIR) Submission

- Please submit the following files to {NTHU eeclass / NTU COOL / NYCU E3}
  - You only need to submit the screenshot of Questasim result of Lab2-1 (FIR)
  - (This lab is only for getting familiar with catapult)

Note: Since the authorization of these lab content, don't upload the file which is not mentioned above to Github.

Lab2-2 Submission

- Please submit the following files to {NTHU eeclass / NTU COOL / NYCU E3}
  - hls code of 02_edgedetect_fsic
  - C simulation log
  - Synthesis report of hardware design
    - Screen shot of table in catapult (including general, run time, memory usage, timing, and area score)
    - The file of rtl report (rtl.rpt)
  - Questasim simulation result
  - tcl file generate by catapult (directives.tcl)
  - Integrate to FSIC (rtl design and testbench)
  - Simulation result of FSIC
  - report_StudentID.pdf (Describe the 5 modification made)
  - Github_link.txt

Note: Since the authorization of these lab content, don't upload the file which is not mentioned above to Github.

Report

- How you design your work (5 modifications)
- What’s the test result of catapult design(C design checker, testbench)
- How to integrate your design in FSIC
- What’s the simulation result of FSIC