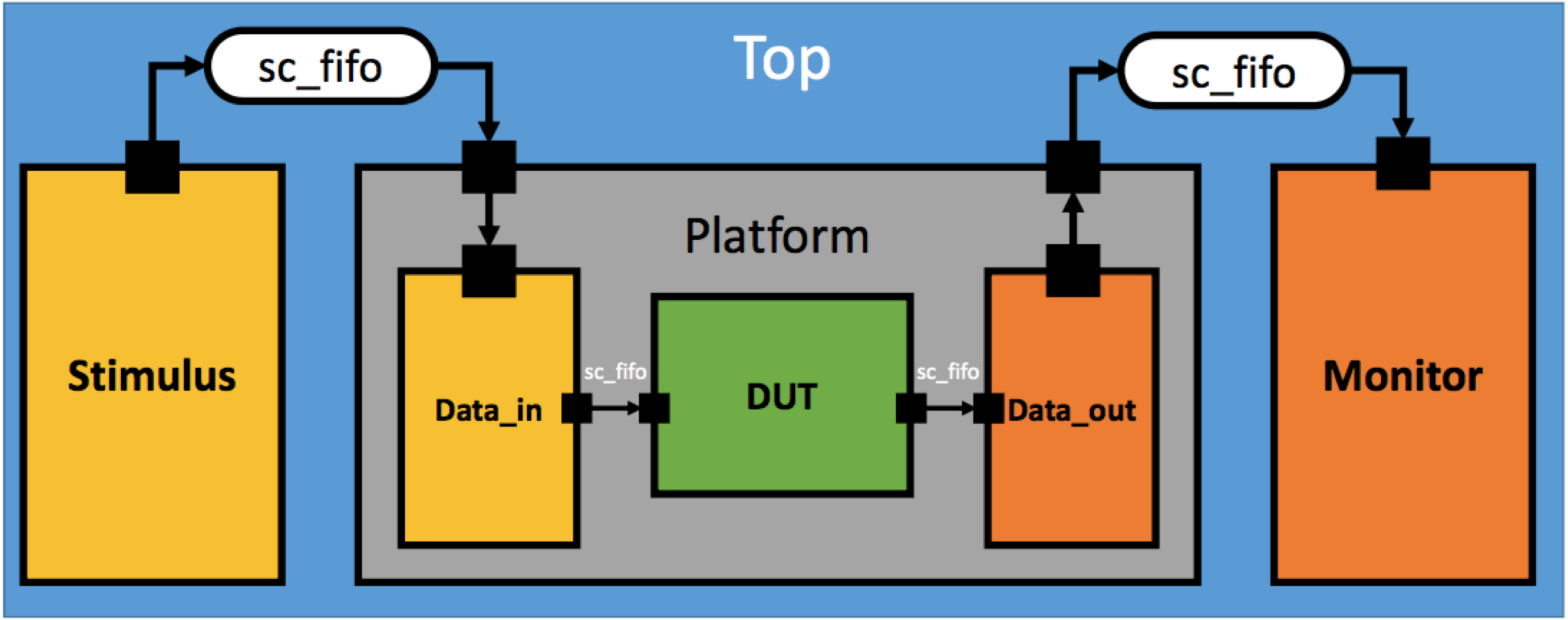
ECPS 203 Discussion

TA: Zhongqi Cheng

Agenda

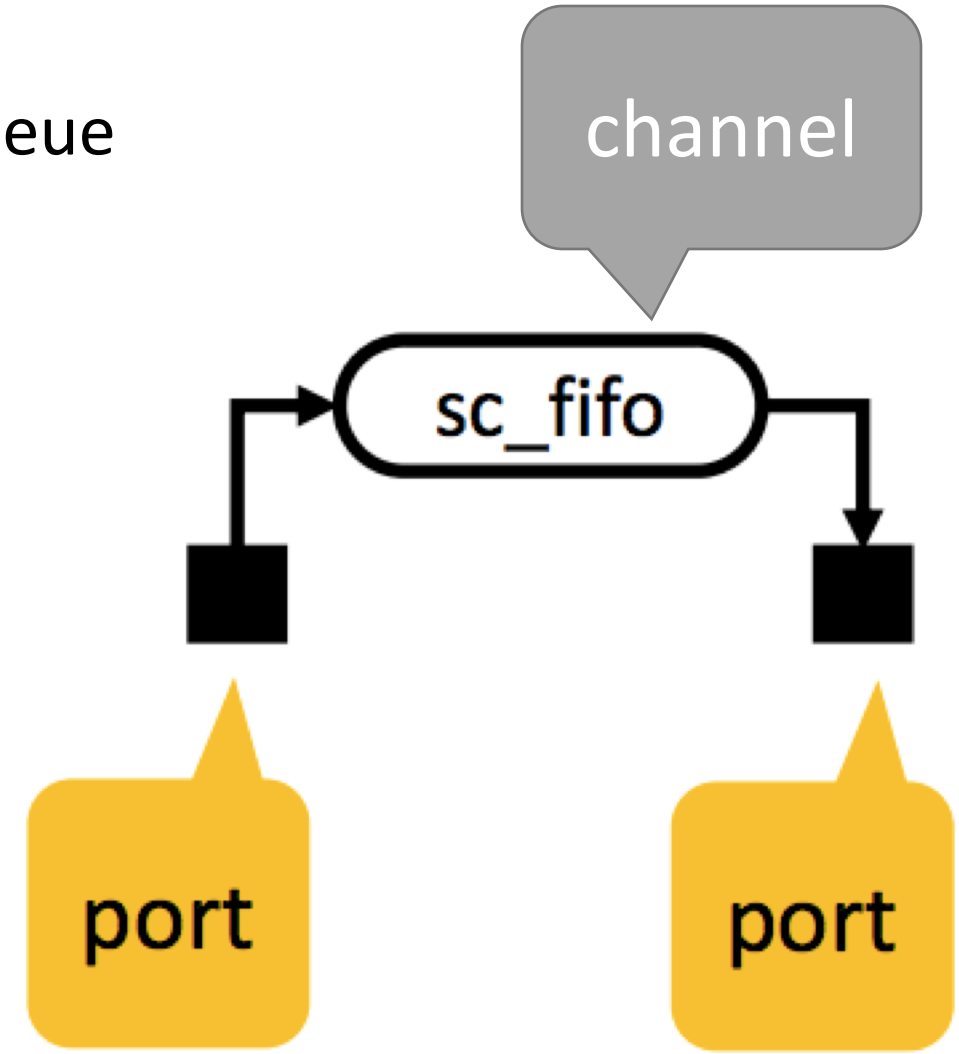
- Build Canny (HW5) in SystemC
  - sc_fifo channel
  - Stimulus
  - Monitor
  - Platform
    - Data_in
    - DUT (design under test)
    - Data_out

Hierarchy


Top
sc_fifo
sc_fifo
Stimulus
Platform
Data_in
sc_fifo
DUT
sc_fifo
Data_out
Monitor

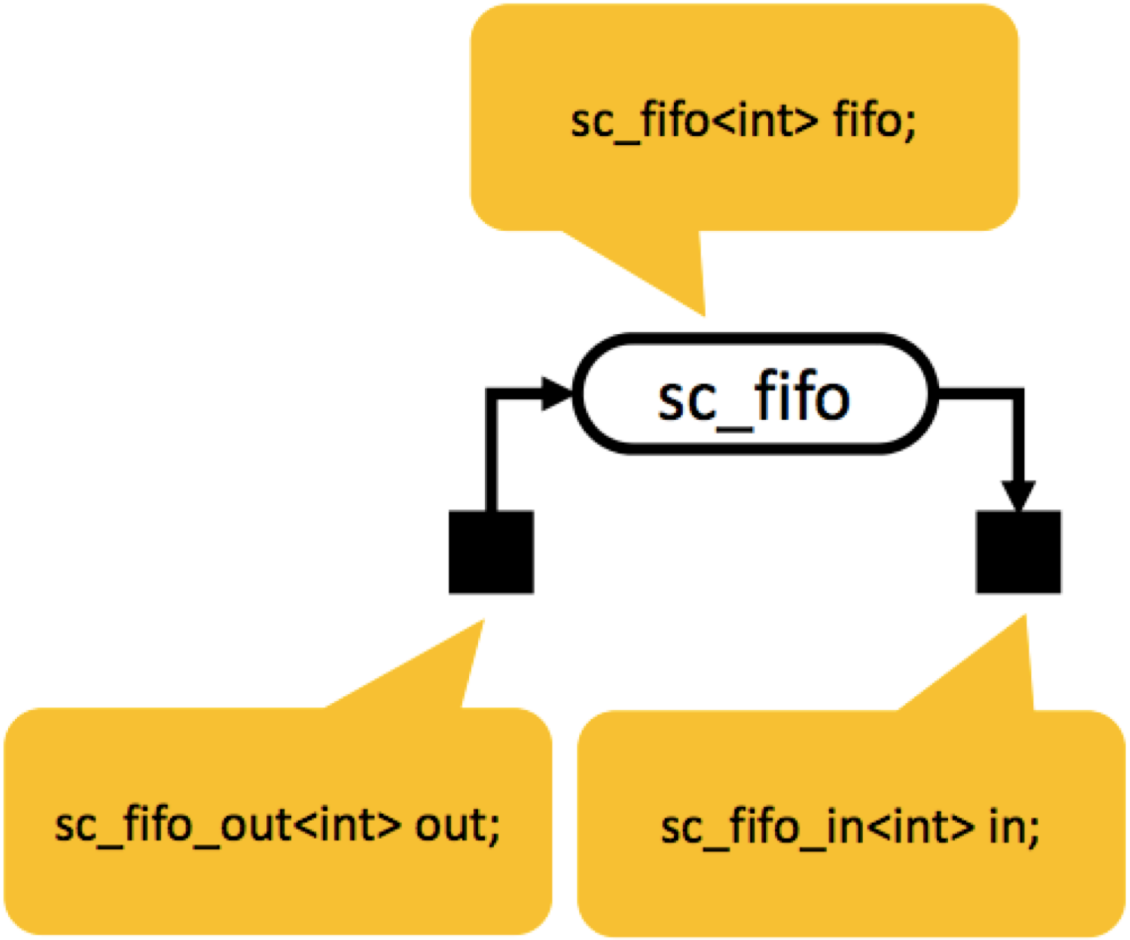
sc_fifo

- First in first out queue



sc_fifo

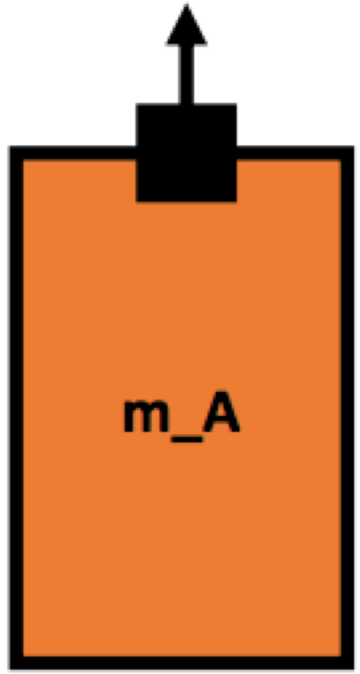
- output port:
  - sc_fifo_out: output from a module
  - sc_fifo_out ***out***;
- input port:
  - sc_fifo_in: input to a module
  - sc_fifo_in ***in***;
- channel:
  - sc_fifo: channel
  - sc_fifo ***fifo***;



sc_fifo

- output from a module: m_A

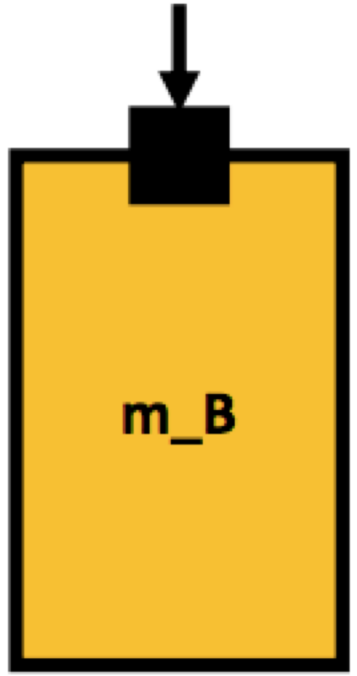
```
int a;
sc_fifo_out<int> out;
out.write(a);
```



sc_fifo

- input to a module: m_B

```
int a;
sc_fifo_in<int> in;
in.read(a);
```



sc_fifo

- binding in the higher level module

```
sc_fifo<int> fifo;
m_A.out.bind(fifo);
m_B.in.bind(fifo);
```

Struct Image_s

- use as the type parameter of fifo
- wraps an image array

- sc_fifo fifo;
- sc_in in;
- sc_out out;

```
typedef struct Image_s
{
        unsigned char img[SIZE];

        ...

} IMAGE;
```

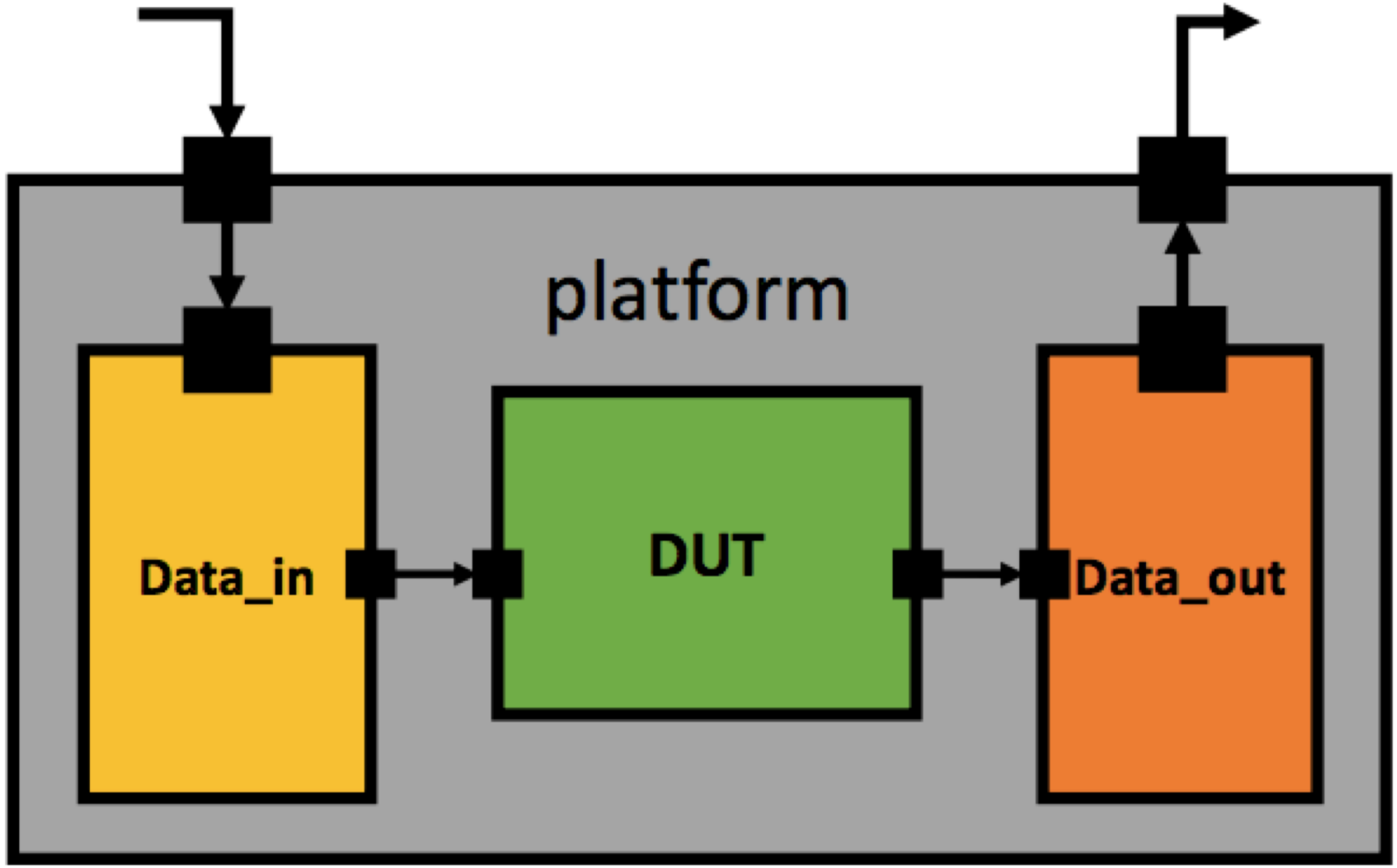
Struct Image_s

- use it as a regular array type

```
IMAGE imageout;
read_pgm_image(infilename, imageout, ROWS, COLS)
out.write(imageout)
```

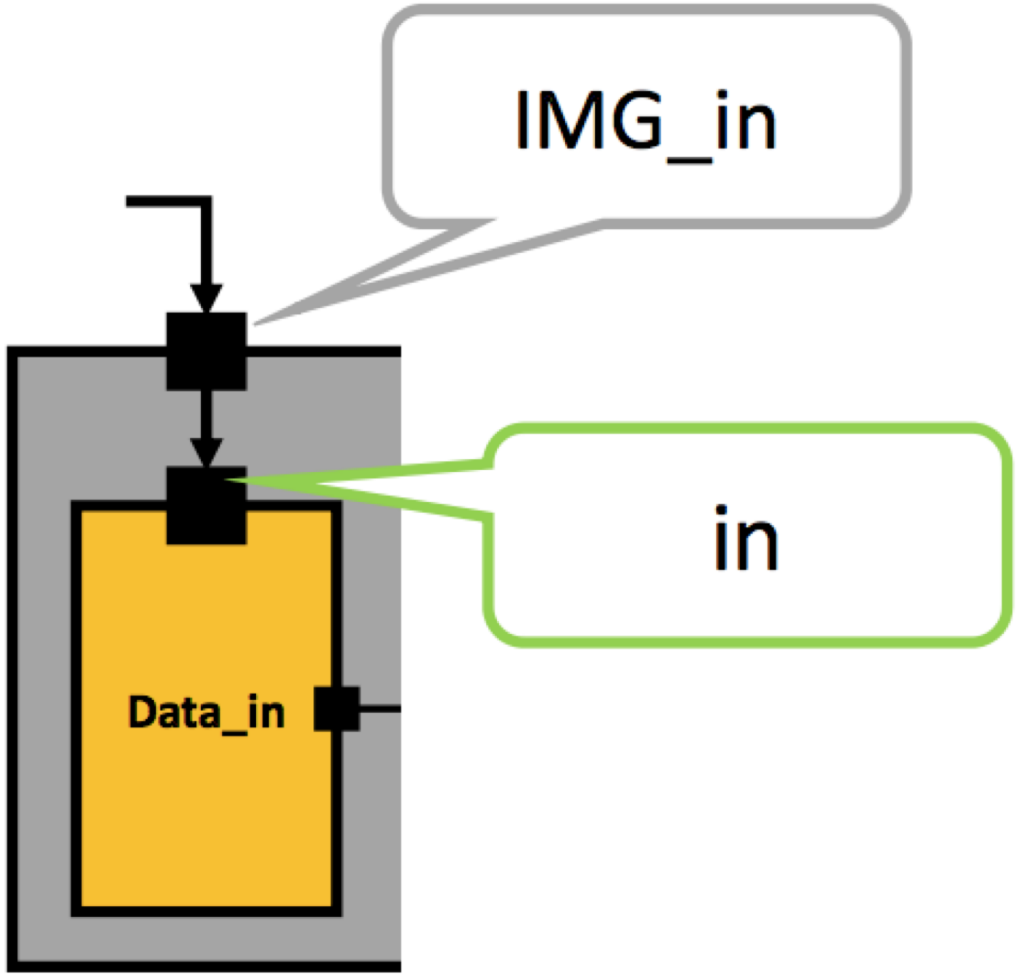
Platform

- Data_in
- DUT
- Data_out



Platform

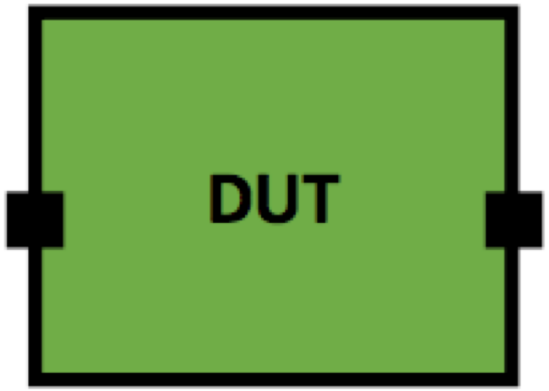
- Data_in.in.bind(IMG_in)



DUT

- performs canny algorithm

1. define a function canny()
2. run all the steps in canny()
3. set canny() as an SC_THREAD



stimulus and monitor

- stimulus:

1. read_pgm_image(…)
2. output the image from port

- monitor:

1. input the image from port
2. write_pgm_image(…)

set stack size

- set_stack_size(128*1024*1024);
- put this code in SC_CTOR
- set stack size in shell as well

Compile and run

- cp ~ecps203/public/MakefileA5 Makefile
- make
- make test

Submission

- Canny.cpp
- Canny.txt