LAB 2

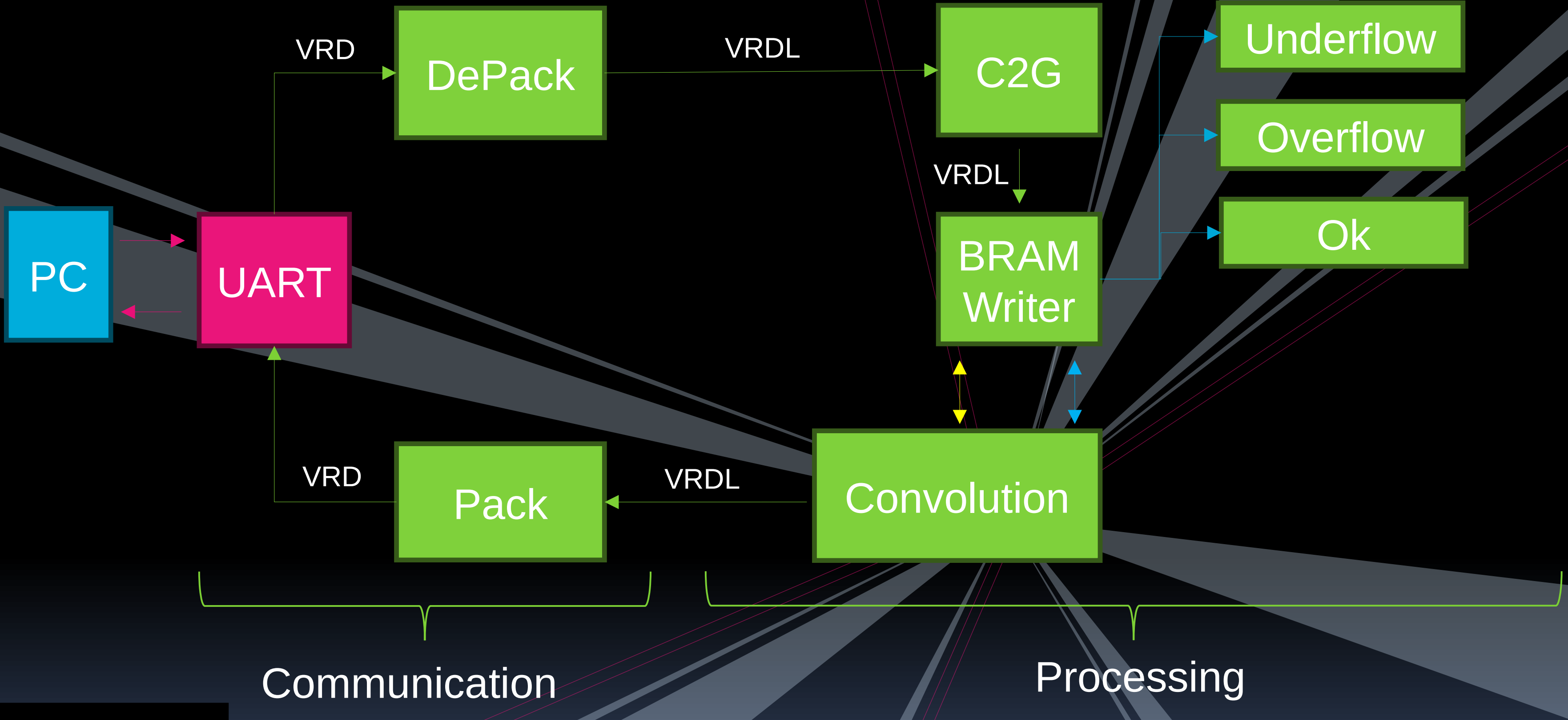
HOME ASSIGNMENT

Home assignment Goal

FPGA-based Image processing:

1. Send/receive images via UART
2. Color-to-Gray (C2G) conversion
3. Save Gray image in BRAM
4. Check correct image dimension
5. Edge detection via image convolution

Block Diagram



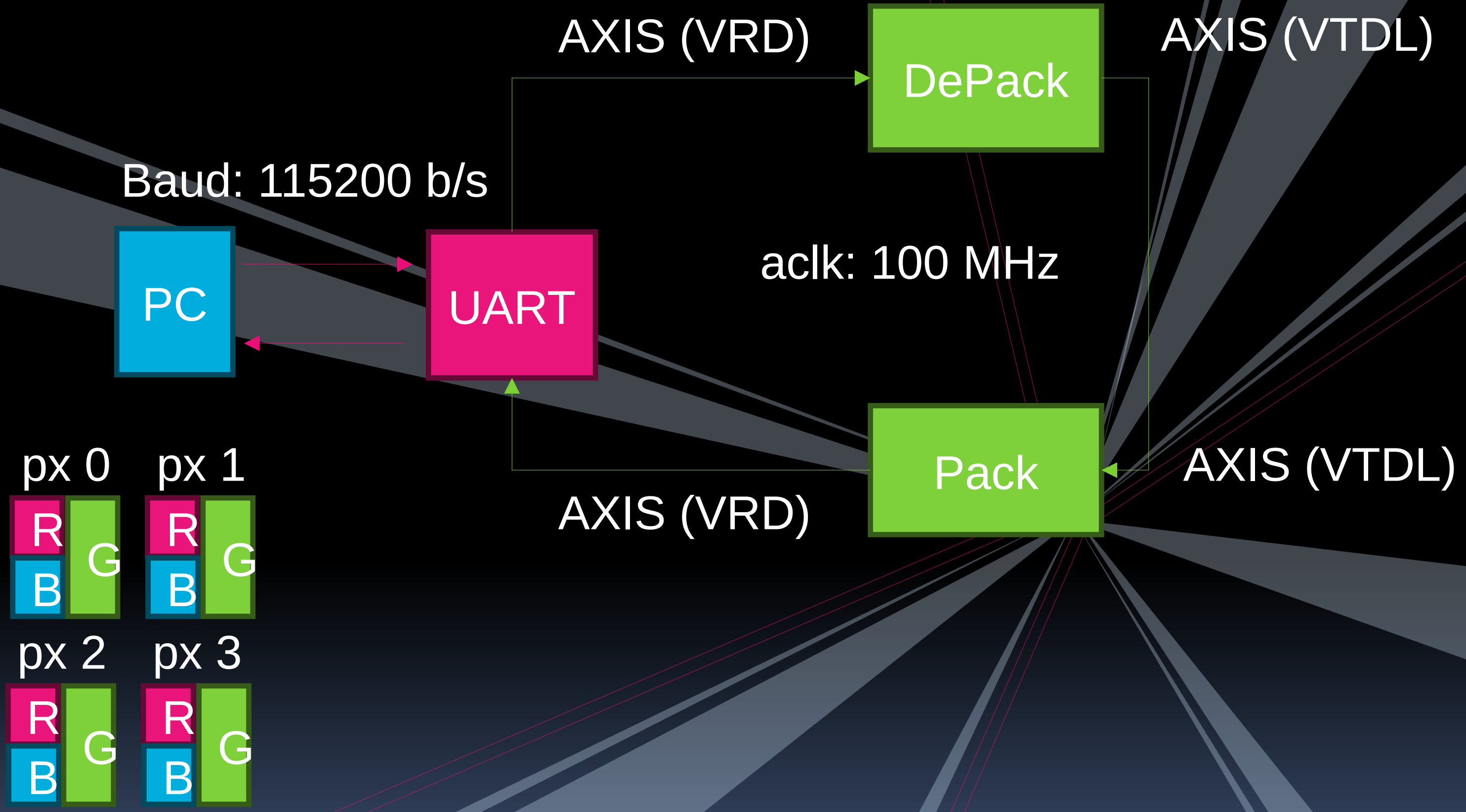
Interfaces:
UART
AXIS
BRAM
Signals

Legend
VRD: Valid Ready Data
VRDL: Valid Ready Data Last

IPs provided by teachers:
UART

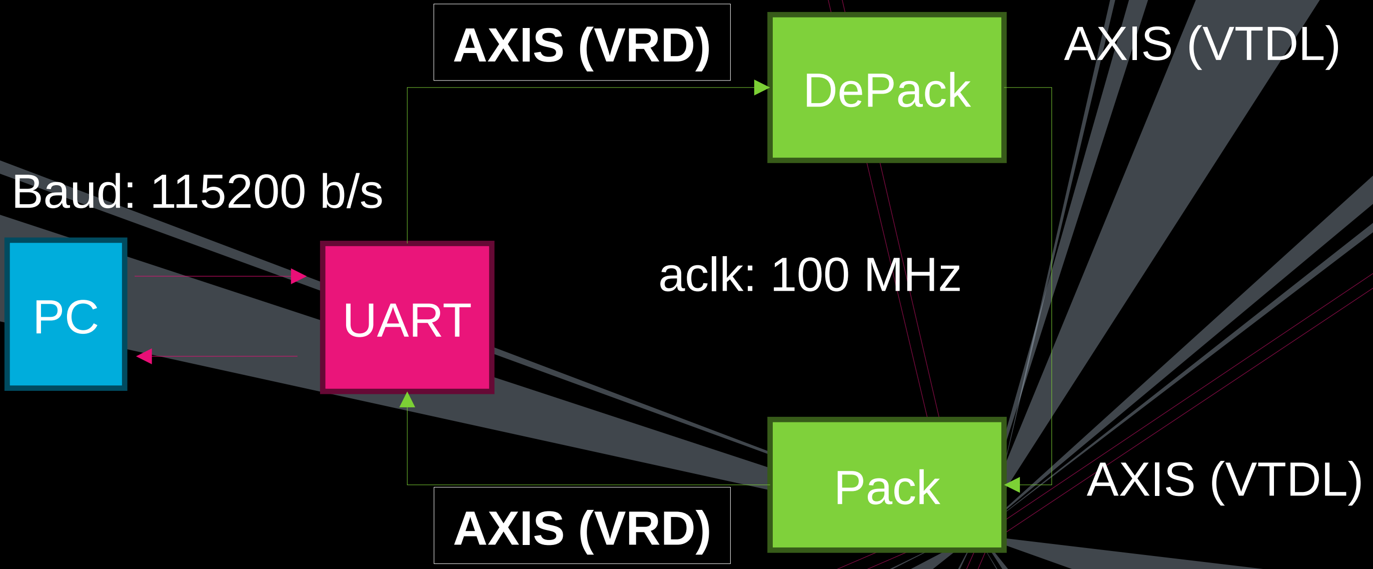
Communication loopback (I)

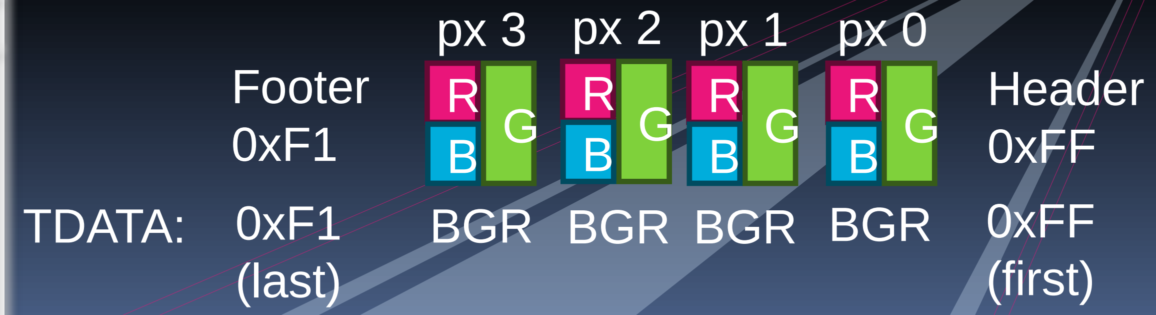
NB: use the smallest possible buffering!



Communication loopback (II)

NB: use the smallest possible buffering!

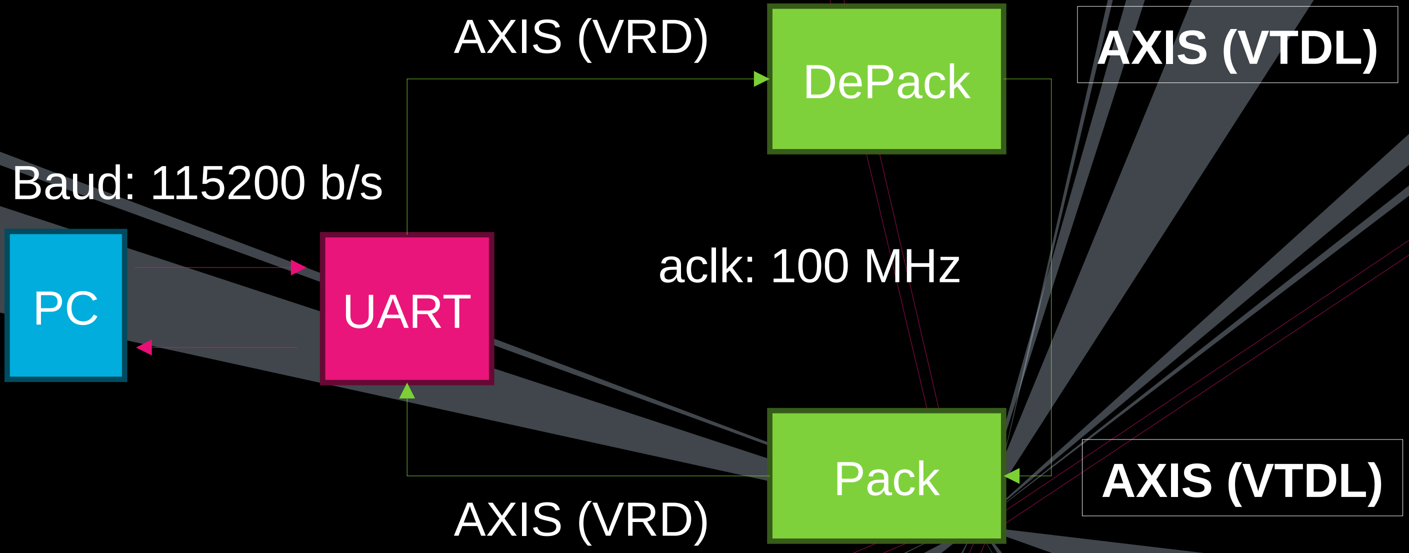


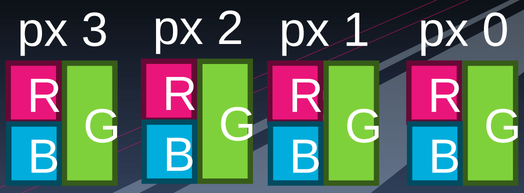


TDATA is 8 bit, **RGB are in range 0-to-127**

Communication loopback (III)

NB: use the smallest possible buffering!



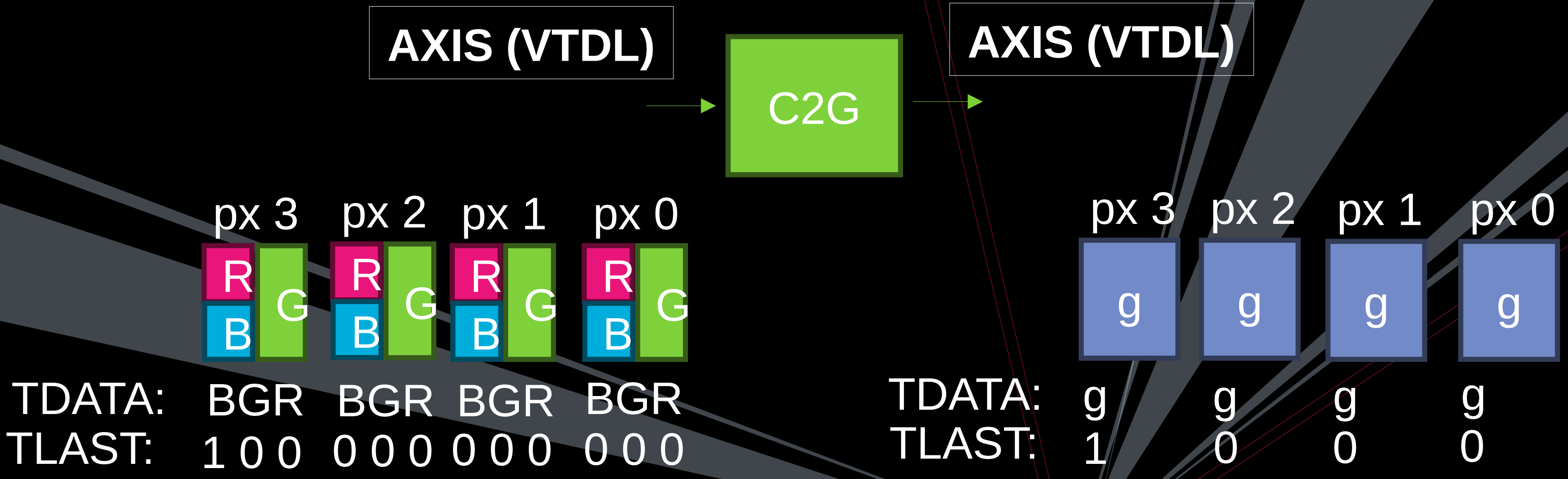


TDATA: BGR BGR BGR BGR

TLAST: 1 0 0 0 0 0 0 0 0 0 0 0

TDATA is 8 bit, **RGB are in range 0-to-127**

Color-to-Gray (I)



g = (R + G + B)/3

E.g.:

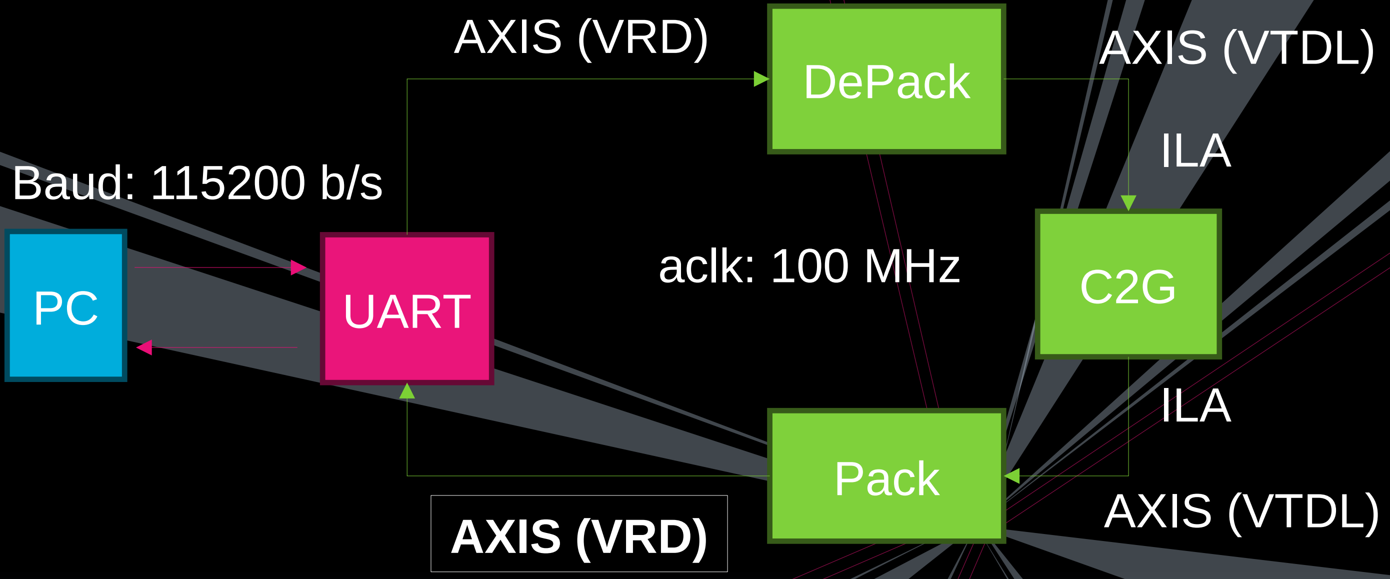
- 5 + 5 + 5 => 5
- 127 + 127 + 127 => 127
- 1 + 2 + 1 => 1 or 2 (*)

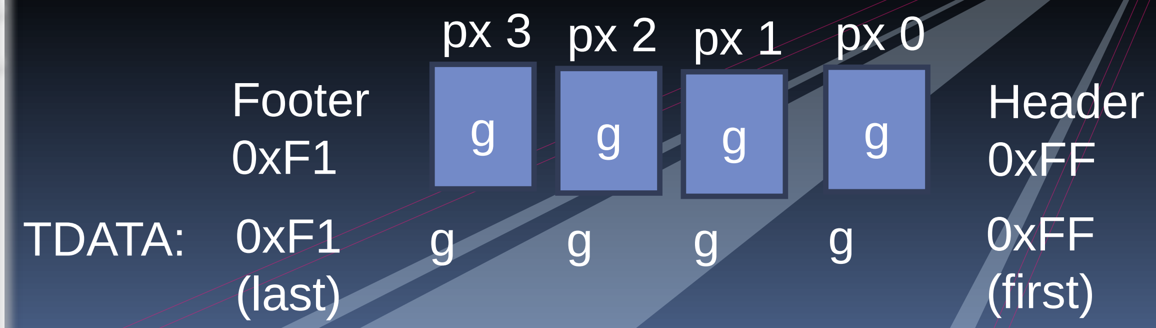
(*): implement the "/3" as a submodule of C2G and work with integer/unsigned and approximation; explain your decisions.

NB: use the smallest possible buffering!

TDATA is 8 bit, RGB and g are in range 0-to-127

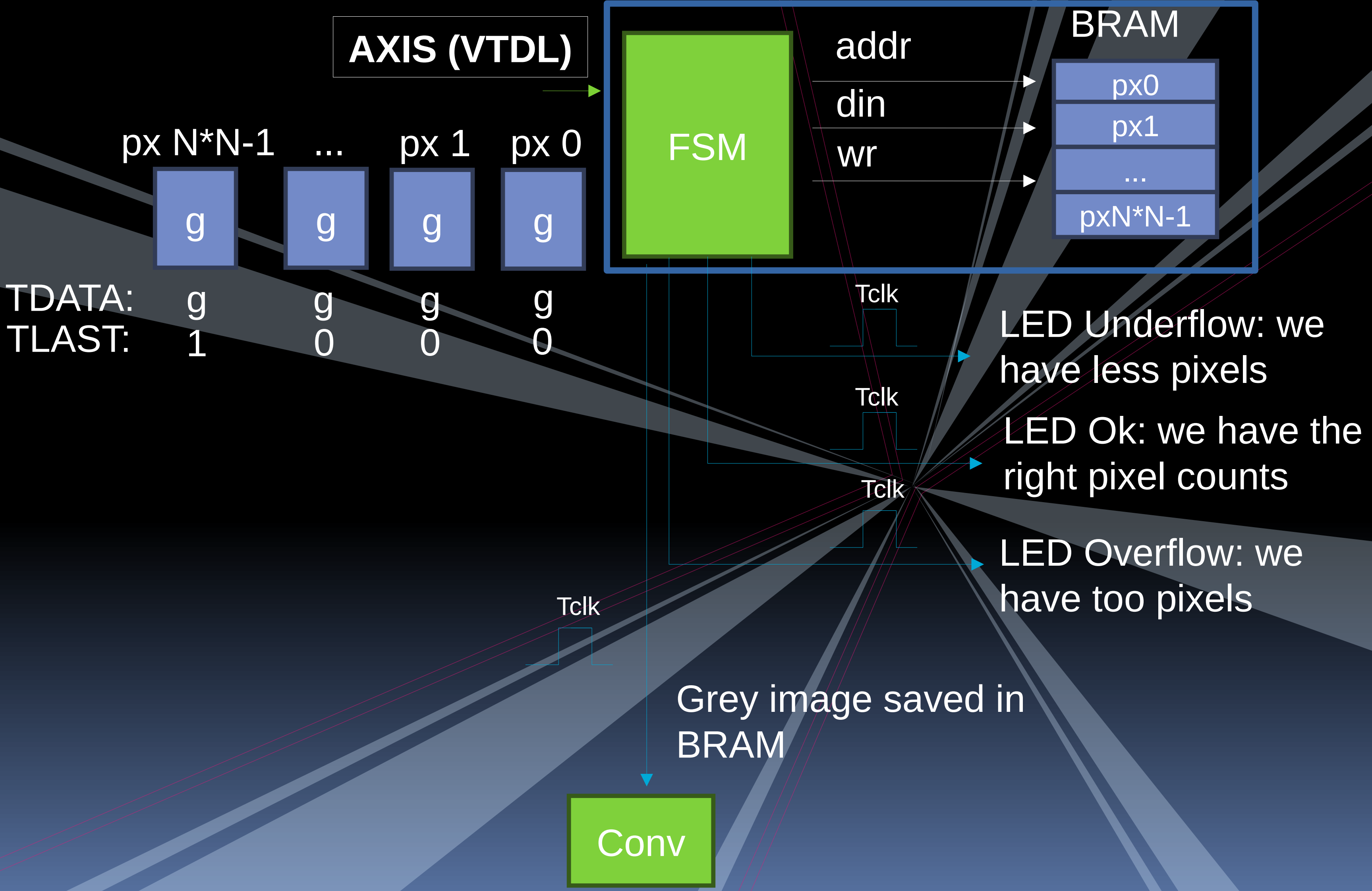
Color-to-Gray (II)





TDATA is 8 bit, g is in range 0-to-127

BRAM Writer (I)



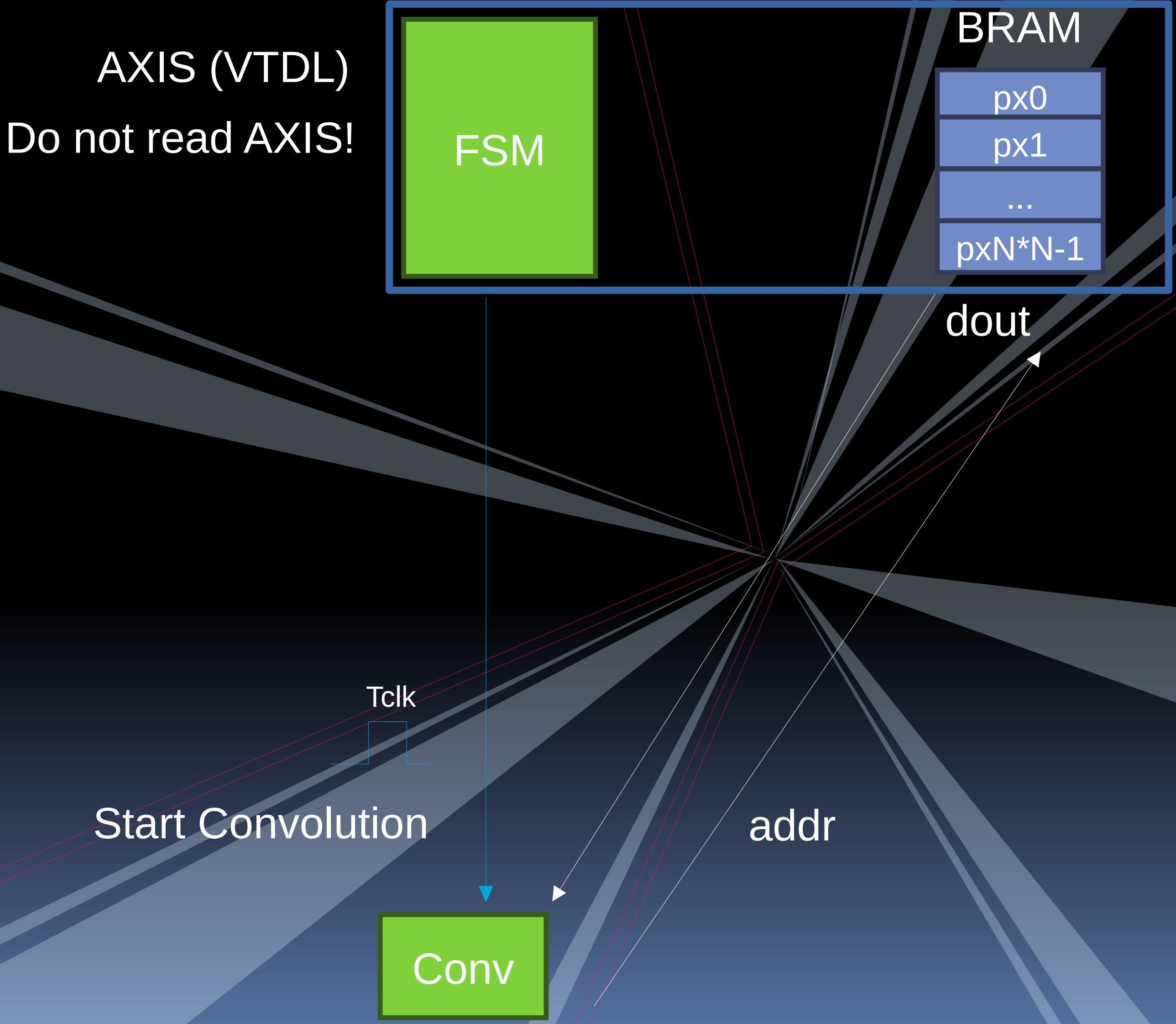
LED Underflow: we have less pixels

LED Ok: we have the right pixel counts

LED Overflow: we have too pixels

Grey image saved in BRAM

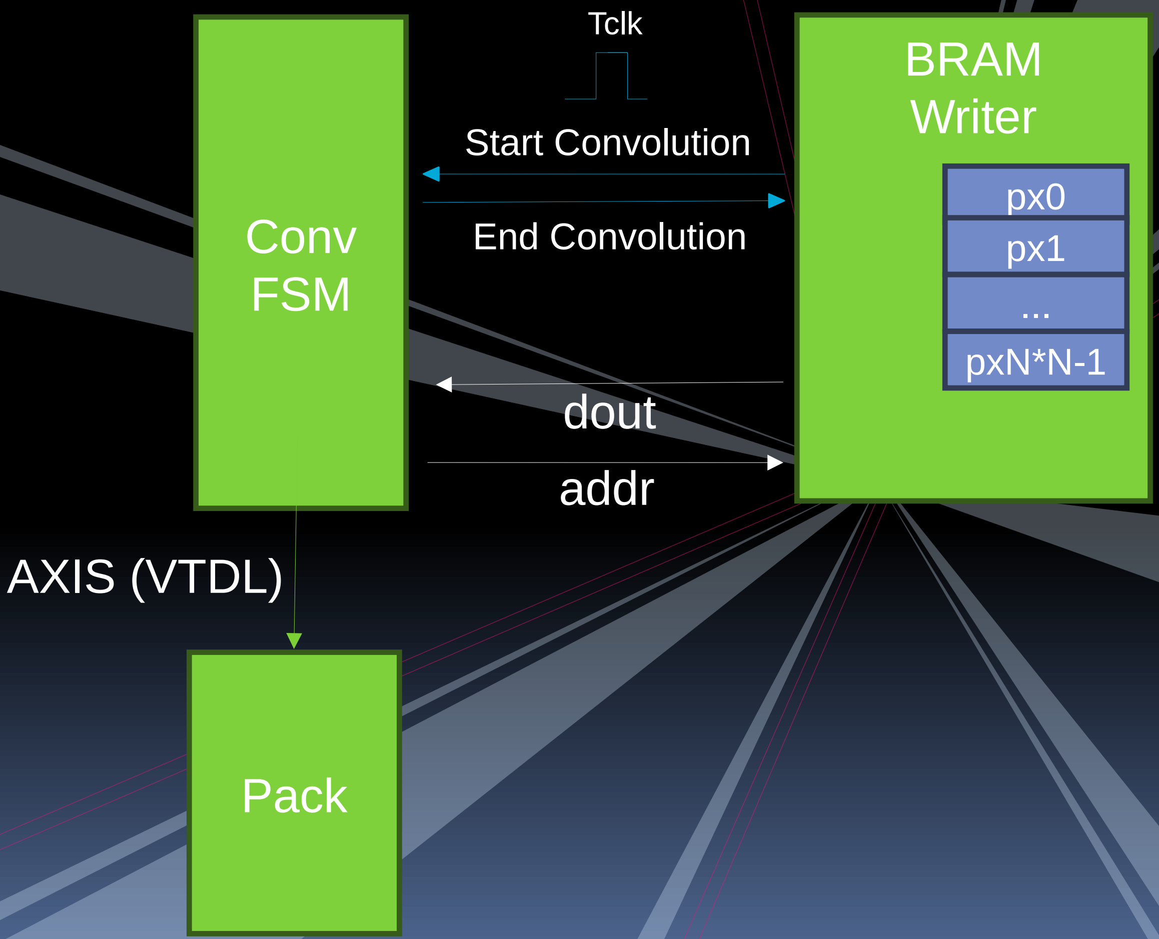
BRAM Writer (II)



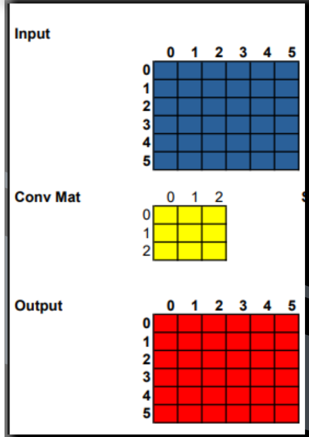
BRAM Writer (III)

Optionally write yourself a VHDL tesbench to simulate the BRAM writer and/or use an ILA.

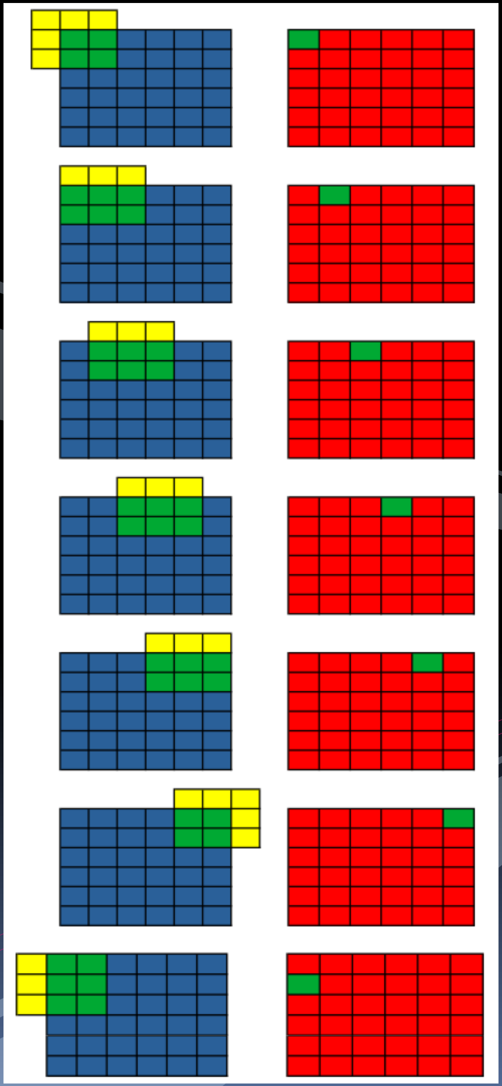
Convolution (I)

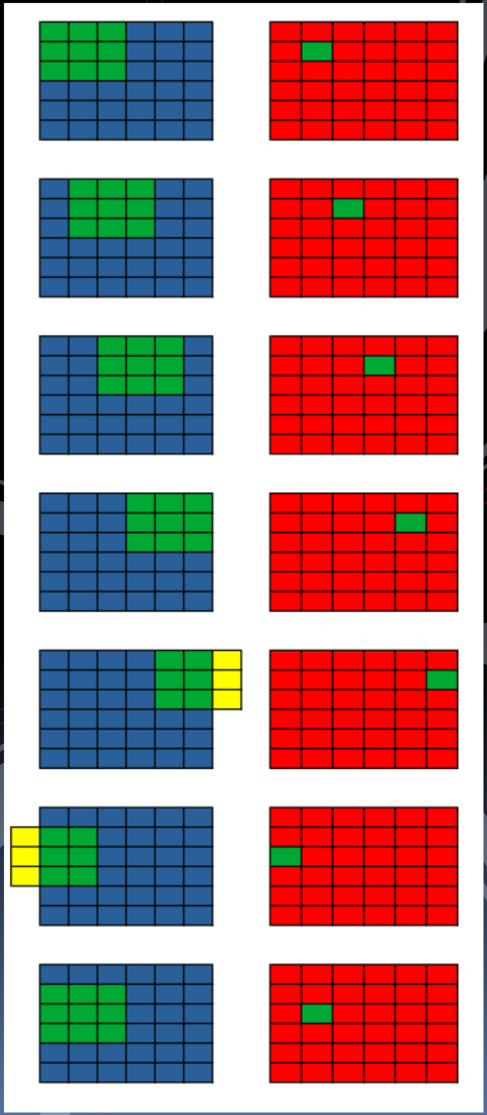


Convolution (II)

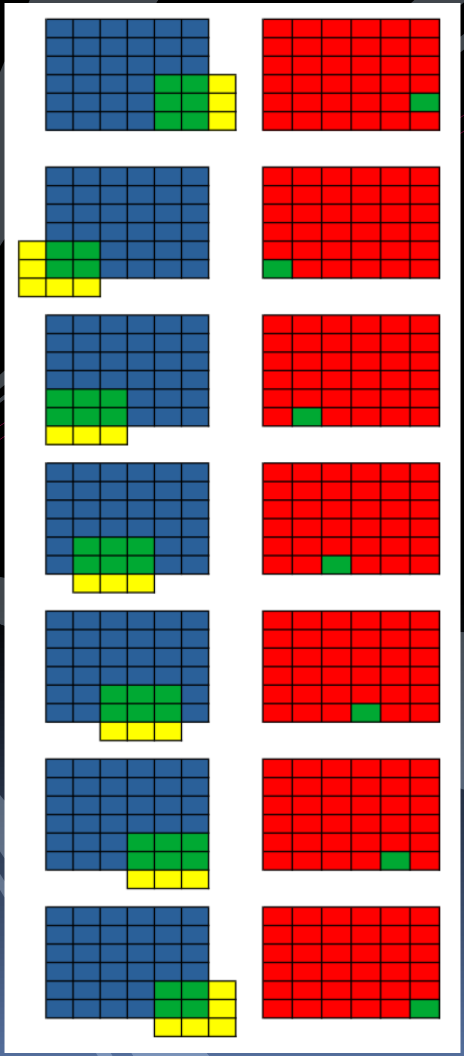


Sums of 9 Cell-by-Cell products

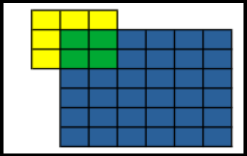




….



Convolution (III)



Use zero if the convolution matrix has cells without pixels

Cells of convolution matrix are *signed(7 downto 0)*
Pixels of grey image are *unsigned(7 downto 0)*

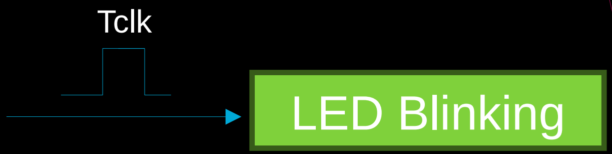
Attention:

- The 9 Cell-to-cell products are signed
- The sum of the 9 cell-to-cell product is signed
- Resize the result of the convolution as *unsigned(7 downto 0)* as follow:
  - If conv < 0 => is 0
  - If conv > 128 => is 127

Convolution (IV)

TestBench is provided

LED Blinking



4 blinking: 1 sec on and 1 sec off.

Delivery

1. Communication loopback (UART + Pack + Depack)
2. Replace the encypted modules with yours in the final project
3. If you write TestBench please provide it to us.
4. Default parameters are specified in the project/VHDL templates
5. Compile the final project with default parameters
6. Archive the full projects (from Vivado) for delivery

File List

1. DESD-LAB2-ENCRYPTED.xpr.zip, encrypted project
2. design_1_wrapper.bit/ltx bitstream and waveform ILA
3. DESD-LAB2-Template.srcs.zip, VHDL templates
4. test.py, send image via UART using python
5. LAB2-Test, test.py compiled for Linux OS
6. LAB2-Test.exe, test.py compiled for Windows OS
7. test1.png and test2.png images

WARNING: When execute the provided programs ensure that test1.png and test2.png images are in the same folder of the executable.

Correction Rules

Led Blinking,	1 pt
Depack,	2 pt
Pack,	2 pt
Color-to-Gray,	2 pt
BRAM writer,	2 pt
Convolution,	2 pt