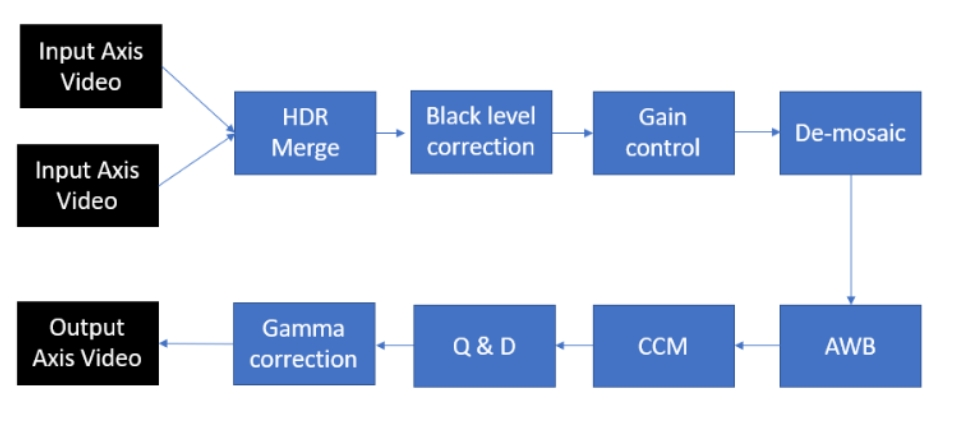
**Application Acceleration with High-Level Synthesis**

GitHub：

Lab\_C Vitis\_Library Vision - Isppipeline\_HDR 陳揚哲 周子翔

**實驗介紹：**

本次的實驗所執行的library為Vitis library中的Vision主題，其中我們選擇的題目為Isppipeline\_HDR(Image Sensor Processing pipeline)，Image Sensor Processing pipeline是Vision中的一大分類，當Sensor從外界蒐集raw image進入程式後，經過處理(Image Sensor Processing)可以根據輸入影像的特徵調整最終影像成像的品質，例如：當Sensor蒐集raw image進入程式後，整個影像的矩陣中可能包含著缺損的像素，需要透過BPC (Bad pixel correction)去移除這些缺損的像素。



資料來源：https://docs.xilinx.com/r/en-US/Vitis\_Libraries/vision/overview.html\_3\_11

HDR(High Dynamic Range)是一種高動態範圍成像方法，通常使用在影像包含強度落差極大的影像中，如果我們使用原始的成像方法，因為HDR影像的影像強度落差極大，導致影像會有強度過強而導致過度曝光的區塊(白)以及強度過弱而無法成像的區塊(黑)，使用HDR方法可使影像依然能保留亮部、暗部的細節。



資料來源：https://hojenjen.com/hdr-definition-and/

**Block的功能：**

1. HDR Merge

用多張不同曝光度的圖片組合HDR圖像，至少使用兩張圖片一張為高曝光度，目標是捕捉圖片中較暗的部分，另一張是低曝光度，目標是捕捉圖片中較亮的部分。



圖 1 左圖是低曝光度 右圖為高曝光度  
資料來源: https://zhuanlan.zhihu.com/p/38176640

1. Black Level Correction

相機的感光元件是由CMOS組成，也就是如同二極體一樣，在未有光線照射時依舊會有漏電流產生，導致沒有光照時也會有電壓輸出。

在現今的相機內部，通常會無光照的CMOS量測無光照的輸出電壓(optical black level)為多少，將原本的量測到的數值減去無光照的電壓，即可還原出原數值。但由於現今的相機廠商為了提高訊雜比，會加一個常數(pedestal)，因此得到的raw data如下公式。

在Black Level Correction的工作是將raw data減去pedestal，還原出真正相機偵測到的數值。

參考資料: <https://blog.csdn.net/weixin_38419133/article/details/115840015>

1. Gain control

Gain control可以改善輸入圖像的整體亮度。在這個block中，對輸入 Bayer 格式的圖像的紅色和藍色通道乘上一個增益（權重）。也就是說，這個block對原始圖像進行增益控制，將紅色和藍色通道的像素值分別乘以一個權重，以提高圖像的亮度。可以改善圖像的對比度和視覺效果。

又由於Bayer 格式的綠色通道是由紅色和藍色通道共同決定的，因此在Gain control的部分只需要針對紅色和藍色通道即可。

參考資料: https://xilinx.github.io/Vitis\_Libraries/vision/2020.1/api-reference.html

1. Demosaic

由於Bayer的資料格式每一格僅有R或G或B的顏色資料如下圖。

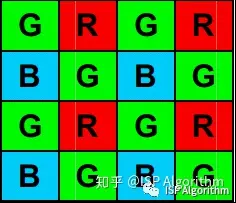


圖 2 Bayer資料格式  
參考資料: https://zhuanlan.zhihu.com/p/267166187

所以Demosaic目的是還原正常圖像的RGB資料，將每一格資料都有RGB資料，常見的方法為內插。

1. AWB(Auto White Balance)

在不同的顏色燈光下，Sensor所呈現的物體顏色可能會與原本顏色有所落差，如下圖所示。

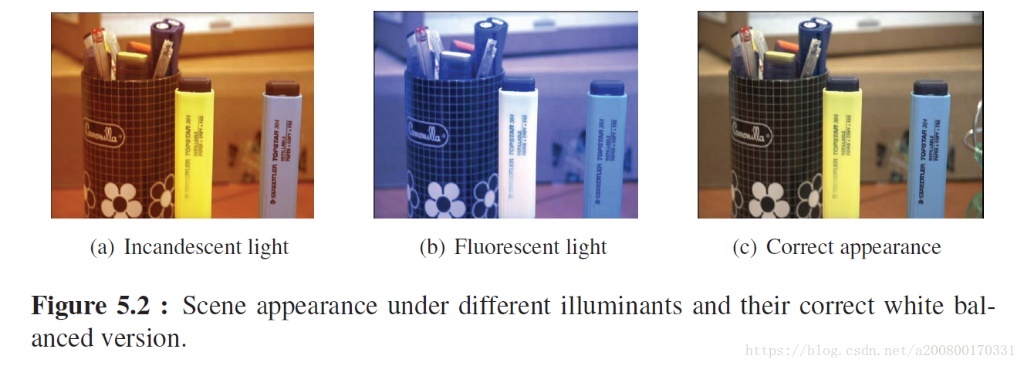


圖 3不同燈光下的物體顏色

AWB的目的是利用調整白平衡來確保場景中的白色物體看起來為白色，且不會呈現色偏。因此AWB會調整圖片色彩平衡和色溫，使得圖片看起來較為自然和真實。

1. CCM(Color Correction Matrix)

Sensor對於光譜的響應和人眼對於光譜的響應有所偏差，因此通常需要乘上一個轉換矩陣，將sensor經由上述計算得到的RGB轉換成人眼響應的RGB數值如下。

一張含有 文字, 資料表 的圖片

自動產生的描述

1. Q & D(Quantization & Dithering)

Quantization: 將圖像的位元數從16bits降低至8bits。

Dithering : 將量化過的圖片(8bits)加入一些抖動的隨機數值，讓圖片看起來更為平滑，如下圖所示，其中左圖為量化後的圖片，右圖為抖動平滑後的結果。

圖 4 Dithering後的平滑結果  
參考資料: https://zhuanlan.zhihu.com/p/33637225



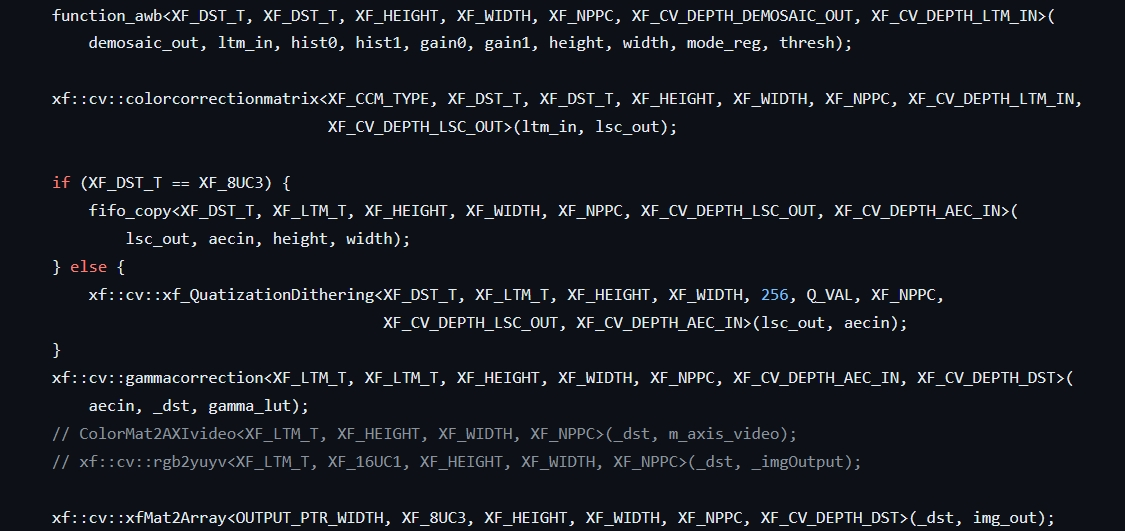
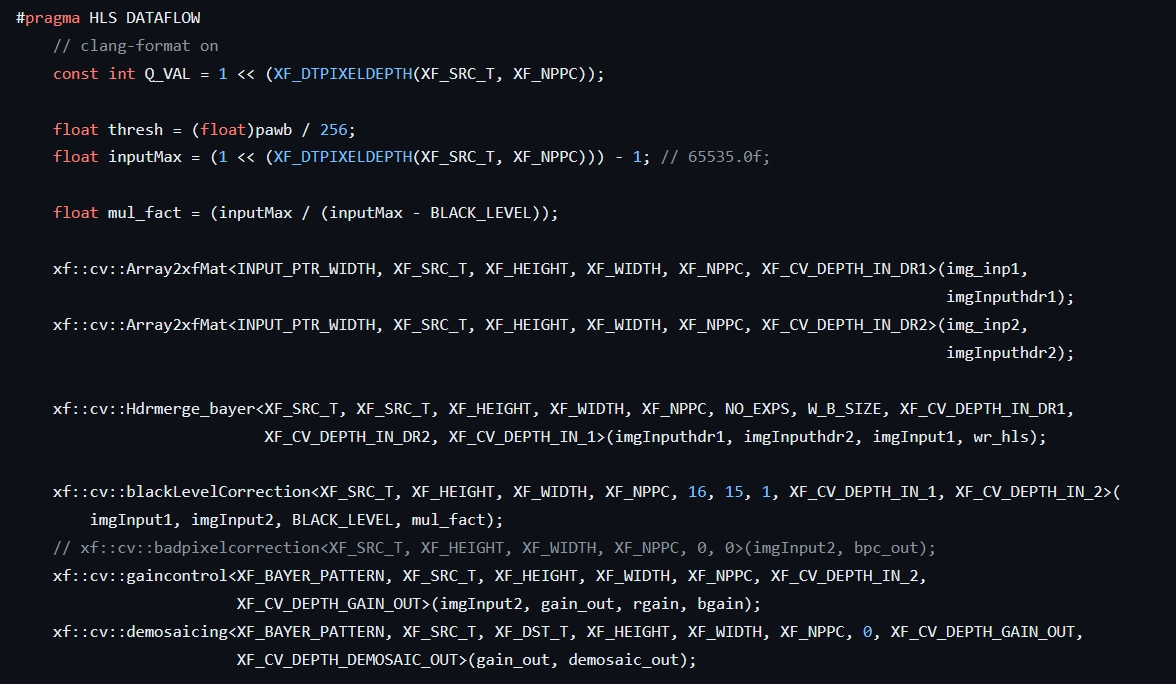
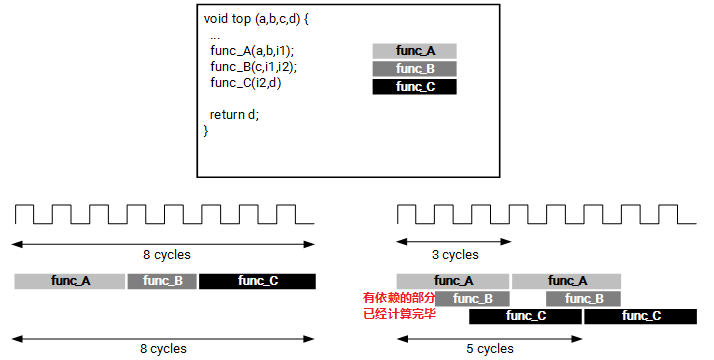
1. Gamma correction

在影像處理中，Gamma correction是一種常見的技術，用於提高影像的對比及亮度，官方給的轉換數值如下。



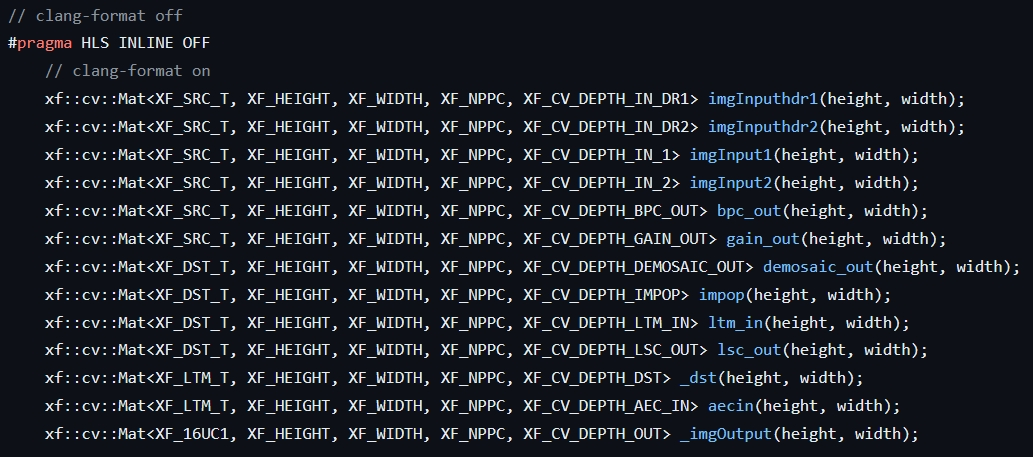
**L3：**

由L3的code可以知道整個演算法的流程依據實驗介紹中的流程圖拆成8個functions，根據之前學過的pipeline原理可知，若我們並未使用pipeline的話，對於各個function而言，需要等到第一次完全計算完畢後才會進行第二次運算。然而，若functions之間的數據依賴與運算流程並沒有feedback的情形發生，即可使用pipeline的方式進行加速，在此我們使用#pragma HLS DATAFLOW 使運算並不需依序執行，而是只要符合條件便可執行該function。



另外，由於運算過程每個function都會產生接續下一個function的中間參數，為了避免function被展開，維持function不被分層，因此對於這些中間參數使用#pragma HLS INLINE OFF以確保pipeline可以正常運作，意即避免

最後使系統可以以下方式執行



**L2：**

與L1相同，同為L3底下的function並進行指定的運算，L2函示庫是在L1級別的基礎上，因此L2的function通常執行比L1更為複雜的演算法，像是Cholesky 分解、LU 分解等等。

但在我們的Function中L1和L2的並沒有太大複雜度的差別，由下兩個gain control程式為例。

一張含有 文字 的圖片

自動產生的描述

一張含有 文字 的圖片

自動產生的描述

圖 5 L1和L2的gain control

L1和L2都有gain control這function，程式內容僅有在IO的設定上有些許不同。因此L1和L2最大的差別在於L1的function進行CSIM、CSYNTH和COSIM，並不能實際運作於FPGA上，而L2的則可以進行SW\_EMU、HW\_EMU和HW可實際被FPGA所調用，且L2的function可以與DDR/HBM進行讀寫。

一張含有 文字 的圖片

自動產生的描述

**L1：**

L1函示庫則是定義了一些基本操作，像是矩陣乘法、向量加法、點積、轉置等等。這些操作都是一些基本的數學運算，也支持多種資料型別，包含floating point、double、complex等，可以拿來構建成更為複雜的演算法。不過L1 function不能直接對DDR/HBM進行存取。

一張含有 文字 的圖片

自動產生的描述