**Lập trình song song trên GPU**

**Hoàng Minh Thanh (18424062)**



**TH2 : Reduction**

C:\Users\tdqua_000\Dropbox\SS-Slides\DeCuong-CDIO\Template CDIO v4.2\Templates\Hinh anh\LogoTruong.png

Bộ môn Công nghệ phần mềm

Khoa Công nghệ thông tin

Đại học Khoa học tự nhiên TP HCM

Contents

[I. Quá trình cài đặt 3](#_Toc45895078)

[1) Cài đặt hàm blurImg trên GPU 3](#_Toc45895079)

[2) Khởi tạo và thực thi trên GPU 4](#_Toc45895080)

[II. Kết quả thực thi 5](#_Toc45895081)

[1) Biên dịch 5](#_Toc45895082)

[2) 8x8 6](#_Toc45895083)

[3) 16x16 7](#_Toc45895084)

[4) 32x32 7](#_Toc45895085)

[5) 64x64 8](#_Toc45895086)

[Giải thích : 8](#_Toc45895087)

Vì máy tính cả nhân của em không có GPU nên bắt buộc em phải sử dụng Google Colab :

<https://colab.research.google.com/drive/1piW2XzDPXJOg3WBliJAvCGBbQqDXeT8N>

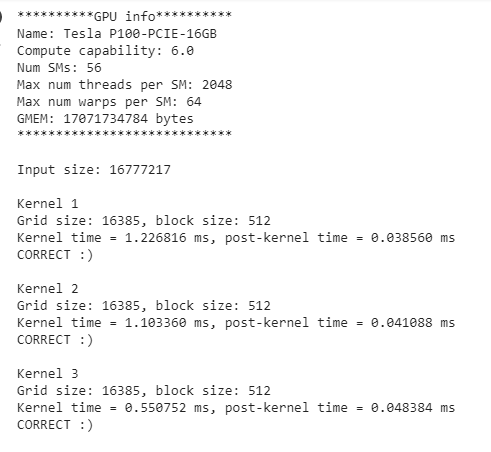
(Thầy có thể vào link Online để xem luồng chạy dễ hơn báo cáo )

# Quá trình cài đặt

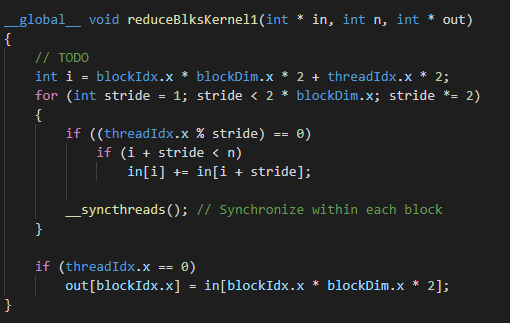
Chi tiết cài đặt trong file .cu và file Google Colab

Vì các file thực thi và câu lệnh em để trên github private repos nên phải cấu hình clone bằng ssh trên Google Colab

## Hoàn thành phần tính “gridSize” và hàm kernel 1 trong file “bt2.cu”



Kết quả cài đặt



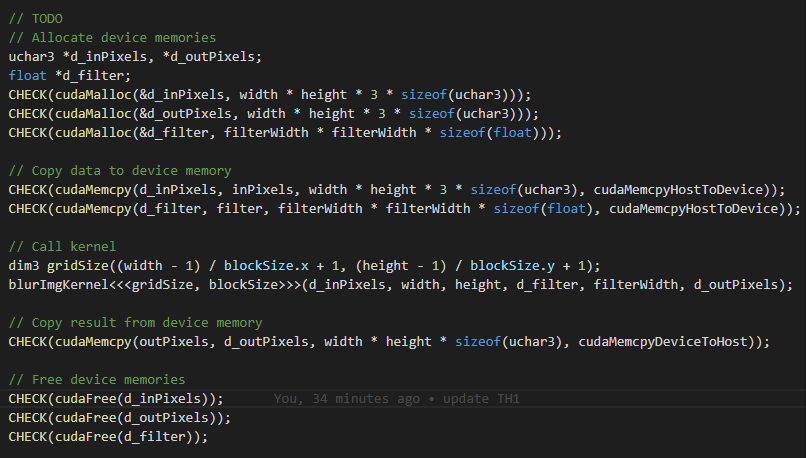
## Chạy chương trình ở câu 1 với các kích thước block khác nhau: 1024, 512, 256, 128.

Đầu tiên ta cần khởi tạo bộ nhớ của pixel trên GPU, và cũng như filer

Sau đó ta copy dữ liệu từ pixels input vào các biến trên GPU

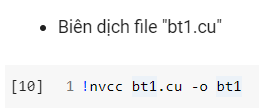
Và sau đó gọi lệnh thực thi trên block vừa tính được

Sau khi thực thì xong thì copy kết quả đã tính toán từ GPU ra và hủy bộ nhớ

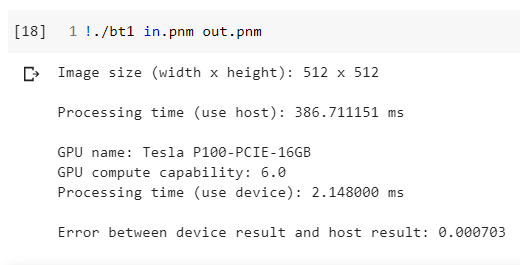


# Kết quả thực thi

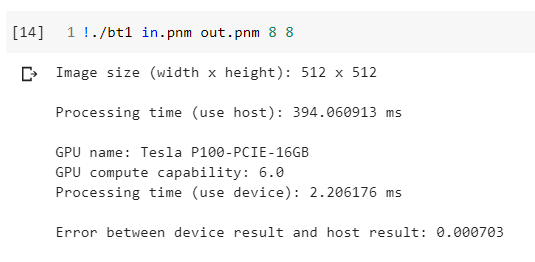
## Biên dịch



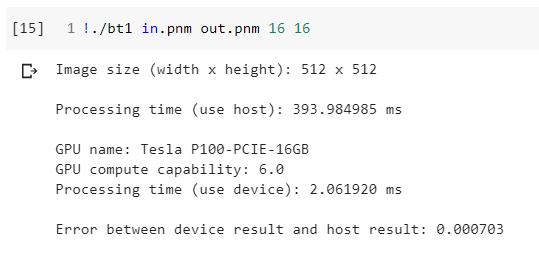
Ta thực thi với cấu hình mặc định



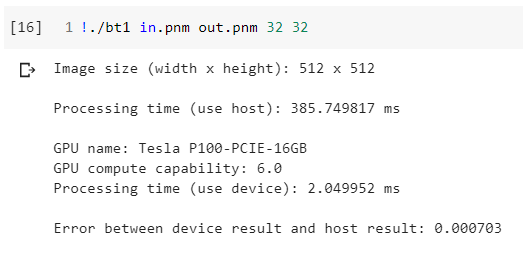
## 8x8



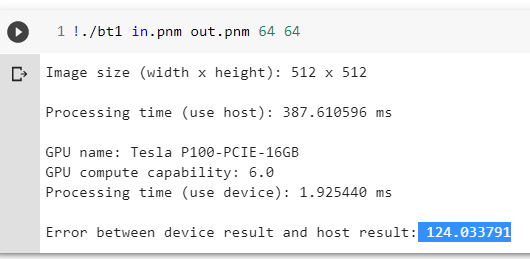
## 16x16



## 32x32



## 64x64



Có thể thấy độ lỗi lên đến **124.033791** khi thực thi với kích thước filter 64 x 64 đúng như với yêu cầu của thầy

### Giải thích :

Xảy ra lỗi như vậy là có thể là do các thread của mỗi block được phân vào các wrap (32 thread) khác nhau. Trong GPU thì mỗi thread trong một wrap thực thi bất đồng bộ sử dụng chung một thanh nhớ register (bộ nhớ tạm trên cùng một wrap) nên có thể với kích thước filter lớn hơn 32 thì quá trình tính toán song song sẽ dấn đến một thread đã xử lý xong và cập nhật kết quả mới vào bộ nhớ toàn cục nhưng thread khác đang xử lý sẽ lấy kết quả pixels từ bộ nhớ tạm trên thanh ghi.

