EECS 222: Embedded System Modeling Lecture 15

Rainer Dömer

doemer@uci.edu

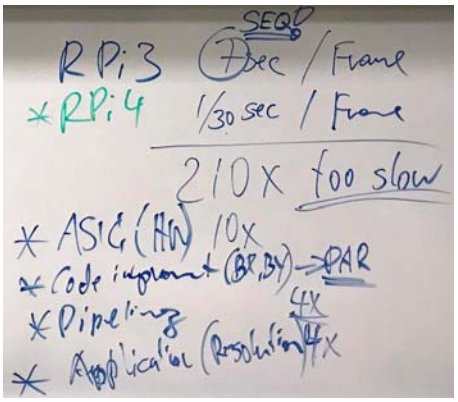
The Henry Samueli School of Engineering
Electrical Engineering and Computer Science
University of California, Irvine

Lecture 15: Overview

- Project Discussion
  - Status and next steps
- Project Assignment 8
  - Back-annotation of timing estimates
  - Pipelining and parallelization of the Canny Edge Detector
  - ➢ Model development on the whiteboard

Project Discussion

- Discussion Questions
  - Does the timing meet our real-time goals?
  - What can be done to improve the speed?
    - Pipelining
    - Parallelization
    - Hardware optimizations
    - Software optimizations
    - Application adjustments



Project Assignment 8

- Task: Pipelining and Parallelization of the Canny Model
  - Pipeline and parallelize the model to maximize throughput
- Steps
  1. Instrument model with logging of simulated time and frame delay
  2. Back-annotate estimated timing in DUT components
  3. Instrument model with logging of throughput (FPS)
  4. Pipeline the DUT into stages for each component
  5. Integrate Gaussian Smooth components into pipeline stages
  6. Slice the BlurX and BlurY blocks into parallel components
- Deliverables
  - **`canny.sc`** or **`canny.cpp`** (choose one!)
  - **`canny.txt`** (with observed timing and frame delays)
- Due: February 26, 2020, 6pm

Project Assignment 8

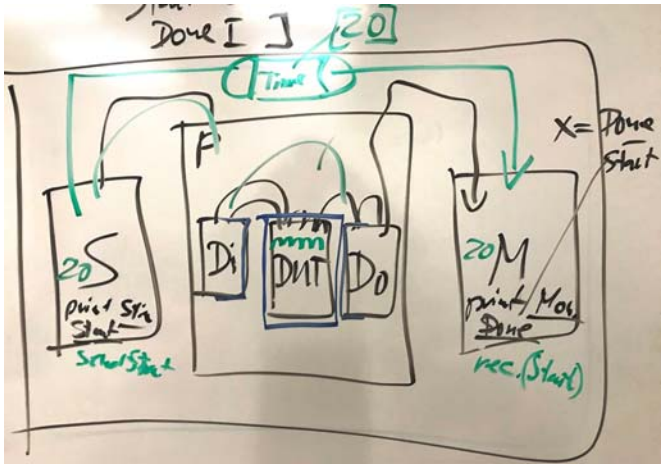
- Step 1: Logging of simulated time and frame delay
  - Expected execution log with timing instrumentation

```
0: Stimulus sent frame  1.
0: Stimulus sent frame  2.
0: Monitor received frame  1 with     0 ms delay.
0: Stimulus sent frame  3.
0: Monitor received frame  2 with     0 ms delay.
0: Stimulus sent frame  4.
0: Monitor received frame  3 with     0 ms delay.
[...]
0: Stimulus sent frame 20.
0: Monitor received frame 19 with     0 ms delay.
0: Monitor received frame 20 with     0 ms delay.
0: Monitor exits simulation.
```

EECS222: Embedded System Modeling, Lecture 15 (c) 2020 R. Doemer 5

Project Assignment 8

- Step 1: Logging of simulated time and frame delay
  - Extended test bench structure:



EECS222: Embedded System Modeling, Lecture 15 (c) 2020 R. Doemer 6

Project Assignment 8

- Step 2: Back-annotate timing in DUT components
  - Insert wait-for-time statements into your model
  - Assume Rasberry Pi 3 performance:

```
Receive_Image          0 ms per frame
Make_Kernel            0 ms per frame
BlurX               1880 ms per frame
BlurY               2010 ms per frame
Derivative_X_Y       530 ms per frame
Magnitude_X_Y        910 ms per frame
Non_Max_Supp         960 ms per frame
Apply_Hysteresis     740 ms per frame
```

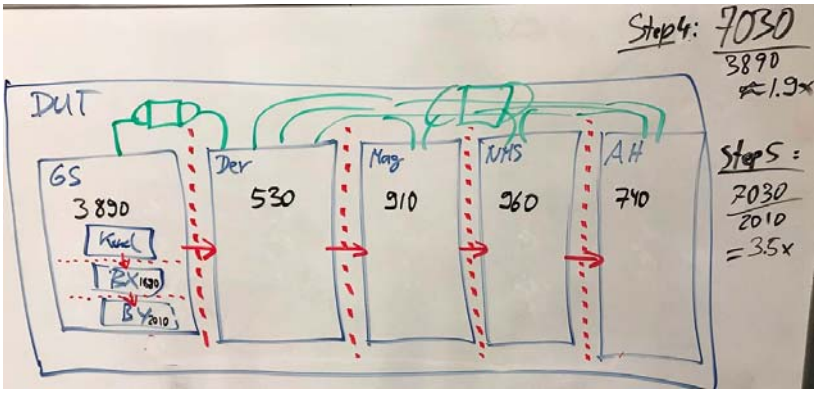

Project Assignment 8

- Step 3: Logging of frame throughput
  - Expected execution log with throughput instrumentation

```
[...]
133570: Monitor received frame 19 with    28120 ms delay.
133570:   7.030 seconds after previous frame,  0.142 FPS.
140600: Monitor received frame 20 with    28120 ms delay.
140600:   7.030 seconds after previous frame,  0.142 FPS.
140600: Monitor exits simulation.
```

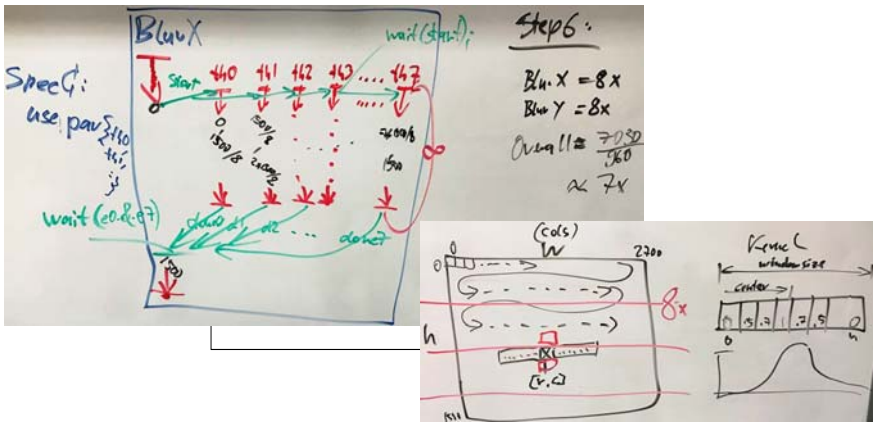
Project Assignment 8

- Step 4: Pipeline the DUT into stages
- Step 5: Integrate Gaussian Smooth into pipeline stages
  - Discussion on whiteboard: Chart of refined DUT structure



Project Assignment 8

- Step 6: Slice the BlurX and BlurY blocks into parallel components
  - Discussion on white board



Project Assignment 8

- Step 6: Slice the BlurX and BlurY blocks into parallel components

```
DUT canny
|------ Gaussian_Smooth gaussian_smooth
|       |------ Receive_Image receive
|       \------ Gaussian_Kernel gauss
|------ BlurX blurX
|       |------ BlurX_Slice sliceX1
|       |------ BlurX_Slice sliceX2
|       |       [...]
|       \------ BlurX_Slice sliceX8
|------ BlurY blurY
|       |------ BlurY_Slice sliceY1
|       |       [...]
|       \------ BlurY_Slice sliceY8
|------ Derivative_X_Y derivative_x_y
|------ Magnitude_X_Y magnitude_x_y
|------ Non_Max_Supp non_max_supp
\------ Apply_Hysteresis apply_hysteresis
```


Project Assignment 8

- Deliverable
  - Observed timing results after each refinement step:

Model	Frame Delay	Throughput	Total time
CannyA8_step1	... ms		... ms
CannyA8_step2	... ms		... ms
CannyA8_step3	... ms	... FPS	... ms
CannyA8_step4	... ms	... FPS	... ms
CannyA8_step5	... ms	... FPS	... ms
CannyA8_step6	... ms	... FPS	... ms