University of Pennsylvania
Department of Electrical and System Engineering
System-on-a-Chip Architecture

ESE532, Fall 2017

Midterm

Monday, October 23

- Exam ends at 4:20PM; begin as instructed (target 3:00PM)
- Problems weighted as shown.
- Calculators allowed.
- Closed book = No text or notes allowed.
- Show work for partial credit consideration.
- Unless otherwise noted, answers to two significant figures are sufficient.
- Sign Code of Academic Integrity statement (see last page for code).

____

I certify that I have complied with the University of Pennsylvania's Code of Academic Integrity in completing this exam.

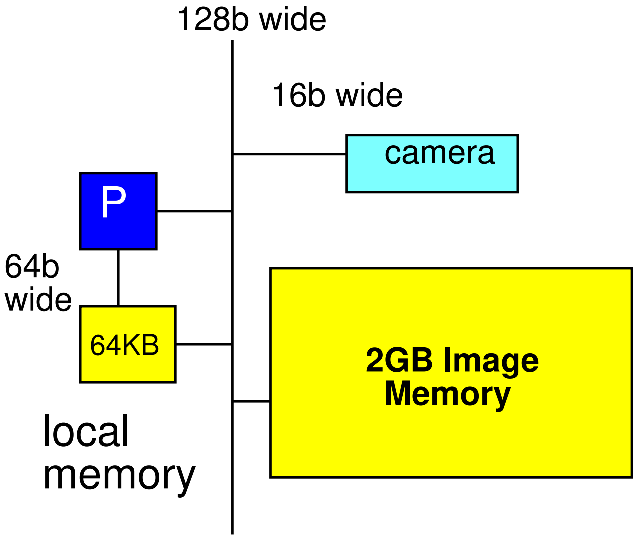
Name: Solution

1	2	3	4	5	6	7	8	9	10	Total
10	5	5	10	10	10	5	15	15	15	100

Consider the following code.

```
uint16_t SI[32][32];
uint64_t RI[16384][16384]; // in 2GB image memory
uint64_t cost, best_cost=MAX_COST;
int xguess=8192; int yguess=8192;
int oldx=8192; int oldy=8192;
int newx=8192; int newy=8192;
int x, y, xoff, yoff, dx, dy;
while (true) {
// A
  for (y=0;y<32;y++)
    for (x=0;x<32;x++)
      SI[y][x]=getPixel(); // reads from camera
// B
  for (yoff=-15; yoff<16;yoff++)
    for (xoff=-15; xoff<16;xoff++)  {
      cost=0;
      for (y=0;y<32;y++)
        for (x=0;x<32;x++)
           cost+=DIST(SI[y][x],RI[yguess+y+yoff][xguess+x+xoff]);
      if (cost<best_cost)
        {
          newx=xguess+xoff;
          newy=yguess+yoff;
          best_cost=cost;
        }
      }
// C
  for (y=0;y<32;y++)
    for (x=0;x<32;x++)
      RI[newy+y][newx+x]=UPDATE(SI[y][x],RI[newy+y][newx+x]);
// D
  dy=newy-oldy;
  dx=newx-oldx;
  yguess=newy+dy;
  xguess=newx+dx;
  oldy=newy;
  oldx=newx;
  newy=yguess;
  newx=xguess;
  best_cost=MAX_COST;
  }
```

We start with a baseline, single processor system as shown.



- Base processor can execute one instruction per cycle and runs at 1 GHz.
- Base processor has a local memory that holds 64KB with single cycle access for 64b data.
- 2GB image memory can perform one operation (read or write) every 20 cycles that transfers 2048b of data.
  - Reading or writing less than 2048b still costs 20 cycles.
  - For the processor, assume you have a macro READ2048 that will initiate a 2048b read from the 2GB Image Memory into the processor local memory and a macro WRITE2048 that will initiate a 2048b write to the image memory from the processor local memory.
  - For the pipelined accelerator (on later questions) assume you have a data mover that can similarly initiate 2048b block transfers from image memory into an associated FIFO and another data mover than can initiate a 2048b transfer from an associated FIFO memory to the image memory.
- Function DIST is a macro that contains 10 primitive operations (instructions)
  - critical path is 4 primitive operations
  - value returned from DIST is a 16b value
- Function UPDATE is a macro that contains 100 primitive operations (instructions)
  - critical path is 15 primitive operations
  - value returned from update is a 64b value
- getPixel() can be called once every 3 cycles and delivers a single, 16b pixel value.
- Assume you store SI in local memory.
- RI only fits in the 2GB image memory.
- Assume scalar (non-array) variables can live in registers.
- You may ignore loop and conditional overheads in processor runtime estimates.

1. Estimate time to perform one iteration of the outer while loop body on a single processor for the code as shown, taking each reference to RI as a separate read or write to the memory.
(note: for this and all estimates, two significant figures is sufficient.)

A	$32 \times 32 \times (3+1)$	4096
B	$32 \times 32 \times 31 \times 31 \times (20+1+10+1+2+2)$ (potentially a few more 1's for update on new best)	35M
C	$32 \times 32 \times (20+1+100+20+2)$	143K
D	9	9
		35M

Processing Time Estimate	35ms

2. What is the lower bound for the processing time to perform one loop body of the outer while loop just considering the 2GB memory and taking each reference to RI as a separate read or write to the memory.

A	none	0
B	$32 \times 32 \times 31 \times 31 \times 20$	20M
C	$32 \times 32 \times (2 \times 20)$	40K
D	none	0
		20M

Memory Lower Bound Estimate	20ms

3. What is the lower bound for the processing time of the outer while loop body just considering the computational operations (computational resource bound)?

A	none	0
B	$32 \times 32 \times 31 \times 31 \times (10+1+2+2)$ (potentially a few more 1's for update on new best)	15M
C	$32 \times 32 \times (100+2)$	102K
D	9	9
		15M

Computational Lower Bound Estimate	14ms

4. What is the latency (critical path) lower bound for the computations in loop B? Assume all the data is available. (This question is about the computation, so the answer will not include any time for reading data out of memory. Previous and subsequent questions ask you about limits reading data from memory.)

Compute all 1M DIST in parallel	4
Compute cost sum as associative reduce	$\log(32 \times 32) = 10$
Compute min over all xoff, yoff as an associative reduce (alternately, could consider this a subtract followed by a mux select)	$\log(32 \times 32) = 10$ (20)
Total	24

Latency (critical path) Lower Bound Estimate	24 ns

If we consider the selection of the best cost as a sequential operation in order to guarantee that we pick the smallest (yoff,xoff) of a given cost, then instead of doing a min reduce, we might then have a sequential selection of best. That would be 14 for the DIST and cost sum followed by 32×32 for the best update, for a total of 1024+14=1038. But, we could still select the smallest (yoff,xoff) without sequentializing the whole computation; that will lead to a critical path of 24 or 34 cycles.

5. What is the lower bound on the number of reads and writes necessary from the 2GB memory for one iteration of the loop body of the outer while loop? Exploit the full width of the memory and assume you store and reuse values from the processor's local memory. Assuming you can achieve this, what is the lower bound for the processing time to perform one loop body of the outer while loop just considering the 2GB memory operations (i.e., revise your answer to question 2).

Reads	128
Writes	32
Memory Lower Bound Estimate	3200 ns

The entire region we need to read in RI for a single outer-loop-body is $(32+32)\,(32+32)$=4K 64b values (32KB). That is 64 rows, where each row is a contiguous set of $64 \times 8 = 512$B. Each read (or write) is 2048b or 256B. So, we need 2 reads per row, or $2 \times 64 = 128$ reads. When we go to write, it's 32 rows of 32 values. We only need one write per row, for a total fo 32 writes.

6. Describe how you would use the processor's local memory block to achieve the lower bound above. With this change, estimate the time to perform one iteration of the outer while loop body on a single processor.

Add a loop before B that reads in all the values needed for the B loop into a variable RI_local. This takes up half of the 64KB local memory. This is done in 2 reads per row, starting with &RI[yguess-15][xguess-15]. B now works only on RI_local. C also works on RI_local. After C add a post loop to write the updated 32 RI_local rows back to RI in the 2GB image memory.

A	$32 \times 32 \times (3+1)$	4096
Bpre	128×20	2560
B	$32 \times 32 \times 31 \times 31 \times (1+1+10+1+2+2)$ (potentially a few more 1's for update on new best)	17M
C	$32 \times 32 \times (1+1+100+1+2)$	105K
Cpost	32×20	640
D	9	9
		17M

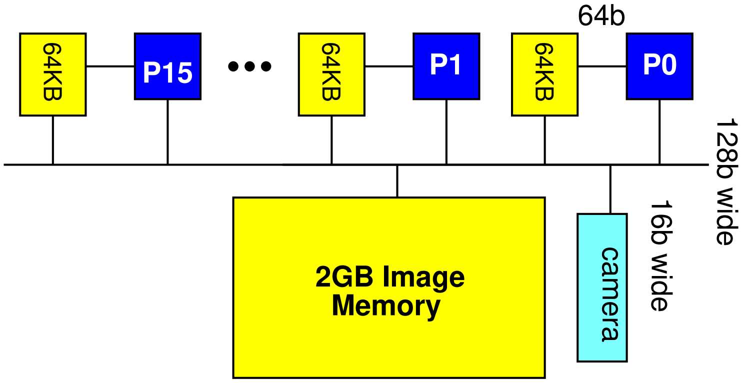
Processing Time	16ms

7. Working from this memory-optimized, sequential version, if you only speed up one of the labeled code segments (A, B, C, D), which one should you speedup and what is the Amdahl's Law limit on the speedup you can achieve for the outer-loop body?

$$Speedup = \frac{A + Bpre + B + C + Cpost + D}{A + Bpre + C + Cpost + D} = \frac{16M + 114K}{114K} = 147 \qquad (1)$$

Speedup Which (circle)	A [B] C D
Upper Bound Speedup	150

8. Building on your memory solution and assuming you have 16 identical processors, describe how you would assign tasks to processors to accelerate this computation. Estimate the throughput achievable in outer-loop-bodies per second on the 16 processor task mapping. Assume it is possible to broadcast to all 16 processors or specify for a read-response from the Image Memory to go to all 16 processors. Assume you have a facility to synchronize on a rendevous point amoung processors (e.g., barrier) that will allow the task set to continue on the cycle after the last processor arrives at the synchronization point. As part of your answer, identify what operations can be run concurrently and what operations must be sequentialized.



A: Read in image from P0 and broadcast values to all processors.
Bpre: Perform the Bpre on P0, broadcasting values to all processors.
B: Split among all 16 processor by outer loop yoff. Processor p performs loop instances yoff=$2p-15$ and $2p+1-15$.
Barrier synchronization on completion of B loops. Must complete B loops before can perform C.
P0 gathers up the best cost and associated newx, newy from the 16 processors and computes the overall best cost and newx, newy.
C: split among 16 processors by outer loop y. Processor p performs loop instances y=$2p$ and $2p+1$.
Barrier synchronization on completion of C loops.
Cpost: Have each processor, in sequence, write its RI updates.
D: Perform calculations on P0.

Throughput (outer-loop-bodies/s)	940

(This page intentionally left mostly blank for pagination and answer space.)

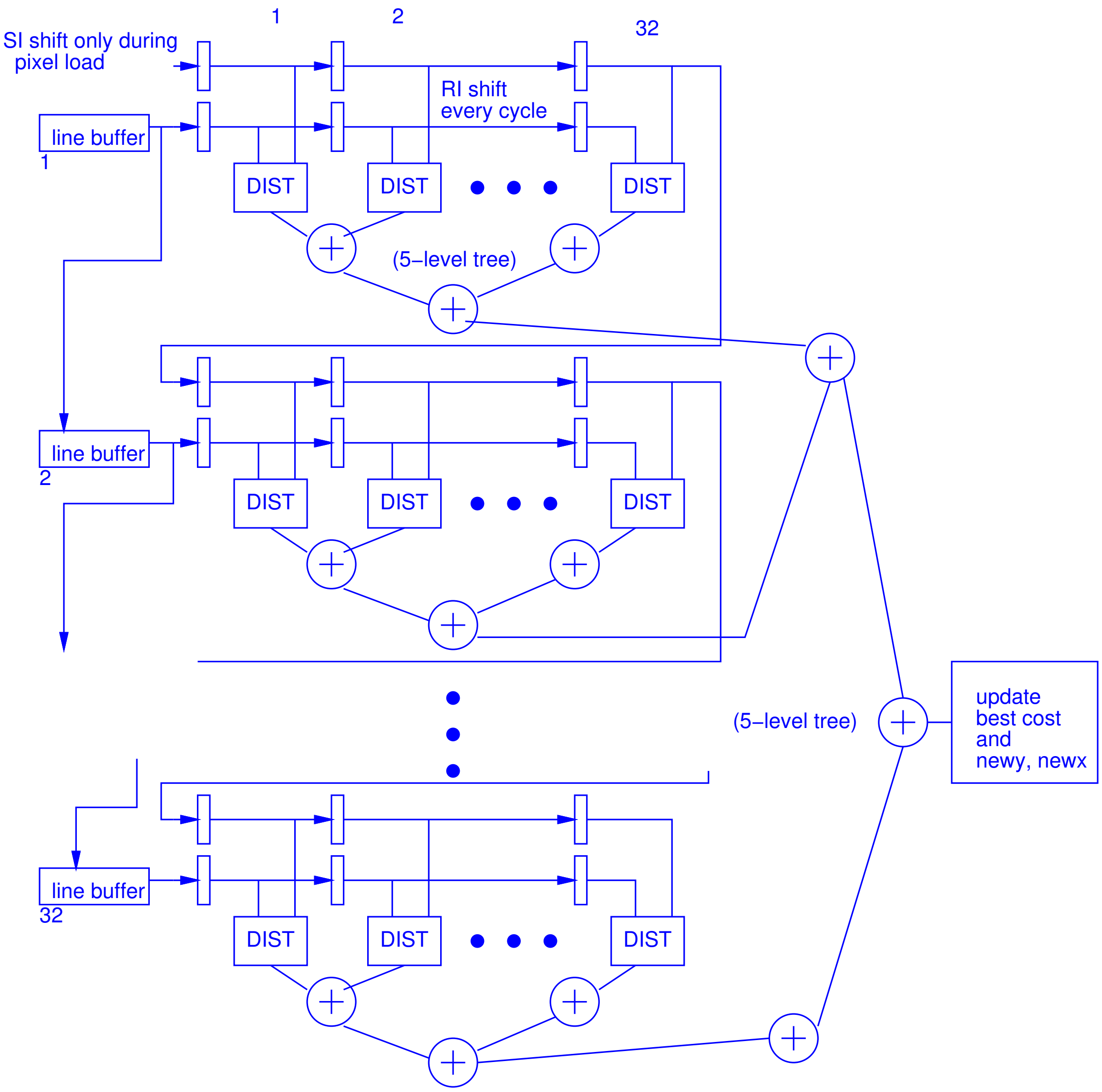
A	$32 \times 32 \times (3+1)$	4096
Bpre	128×20	2560
B	$2 \times 32 \times 31 \times 31 \times (1+1+10+1+2+2)$ (potentially a few more 1's for update on new best)	1M
Bmin	16×10 crude approximation for calculate best	160
C	$2 \times 32 \times (1+1+100+1+2)$	6720
Cpost	32×20	640
D	9	9
		1.06M

9. Describe how to build a pipeline for the B loop that allows the computation in this loop to execute in a little over 1024 cycles (1024 cycles plus the time to drain the pipelines). You may include customized local memories for data storage. Draw pipelined structure (but you won't be able to show every element). Counting each primitive operation as 1 unit and each KB of memory used as 1 unit, estimate the area required for this pipelining. (To simplify accounting for this problem, we will assume register cost is negligible.)

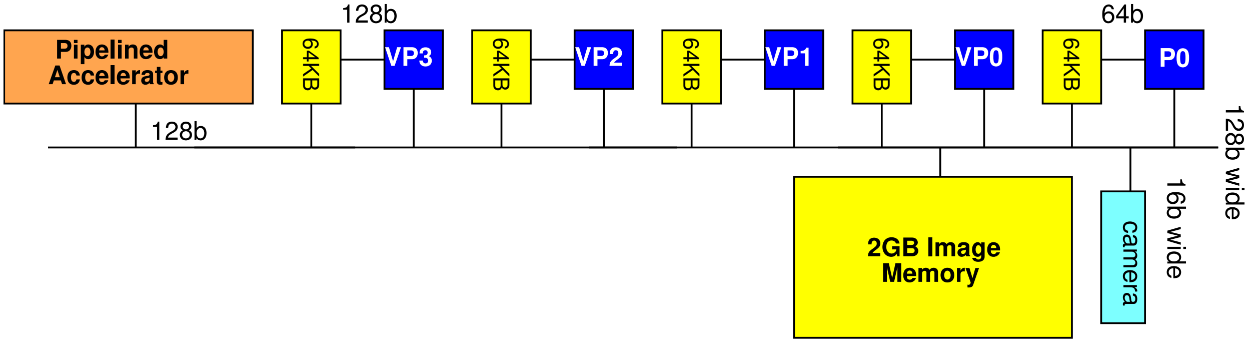
Fully unroll and pipeline the inner two loops (y and x) so that the datapath can start the computation of the cost for a unique (xoff, yoff) pair every cycle. This requires fully partitioning SI, so there is a register for each element of the SI array. It requires a shift register configuration for the active portions of the RI row, and line buffers for the portion for the portion of the row that won't be used again until the next yoff. The additional cycles beyond the 1024 cycles to initiate each (xoff,yoff) position are the ones to drain the pipeline—4 for DIST, 10 for the sum reduce for the cost add, plus 1 for the best cost update.

Compute hardware is 10 operations for DIST plus 1 for the add for each of the 32×32 unrolled inner loop bodies, or 11K. The best cost update is less than 10 primitive operators. We need at most 32 line buffers holding 64×8 Bytes or 16KB plus another 32 lines worth of memory totaling another 16KB. With care, we can probably use only 32×8 line buffers with the other 32 values in the shift registers and loading the other 32 lines just as needed. Nonetheless, even using all 32KB, this only adds another 32 units of area, which remains a second order area term.

Area	11K


1
2
32
SI shift only during pixel load
RI shift every cycle
line buffer
1
DIST
DIST
DIST
(5–level tree)
line buffer
2
DIST
DIST
DIST
(5–level tree)
update best cost and newy, newx
line buffer
32
DIST
DIST
DIST

10. Describe the entire solution using the B pipeline above along with 4 vector and one non-vector processors. What can execute concurrently and what operations must be serialized? Estimate the throughput achievable in outer-loop-bodies per second. Assume the vector processors have a 128b-wide vector processing unit that can process 8 16b primitive operations per cycle and you can perform a perfect vector mapping of the UPDATE routine. Further assume you can transfer 128b in a cycle between the local memory and the vector processing unit.



P0: Runs A reading pixels and loading into SI shift register for B. (4096 cycles)
Stream read first half of RI window into B (and into local memories for VP0 through VP3). (1280 cycles)
Execute B, prefetching next row concurrently while operating at one row position (yoff); also read rows into VP0 through VP3 local memory. (1038+10 cycles)
After B completes, run C on VP0 through VP3.
Give 8 rows to each vector processor. $(8 \times 32 \times (100/8) = 3200)$
Have each VPi, in sequence, write its RI updates. (640)
Perform D on P0. (concurrent with RI writeback).
Total 10,264 cycles.

Throughput (outer-loop-bodies/s)	97K

With additional care to double-buffer the SI values (e.g. a separate shift-register from the storage register to use during the DIST computations), we can run the reading of the pixels for the next frame concurrent with all the rest of the computation. That would reduce the time to max(4096,6168). With a bit more care we can overlap parts of the RI writeback with the UPDATE computation.

Code of Academic Integrity

Since the University is an academic community, its fundamental purpose is the pursuit of knowledge. Essential to the success of this educational mission is a commitment to the principles of academic integrity. Every member of the University community is responsible for upholding the highest standards of honesty at all times. Students, as members of the community, are also responsible for adhering to the principles and spirit of the following Code of Academic Integrity.*

Academic Dishonesty Definitions

Activities that have the effect or intention of interfering with education, pursuit of knowledge, or fair evaluation of a students performance are prohibited. Examples of such activities include but are not limited to the following definitions:

A. Cheating Using or attempting to use unauthorized assistance, material, or study aids in examinations or other academic work or preventing, or attempting to prevent, another from using authorized assistance, material, or study aids. Example: using a cheat sheet in a quiz or exam, altering a graded exam and resubmitting it for a better grade, etc.

B. Plagiarism Using the ideas, data, or language of another without specific or proper acknowledgment. Example: copying another persons paper, article, or computer work and submitting it for an assignment, cloning someone elses ideas without attribution, failing to use quotation marks where appropriate, etc.

C. Fabrication Submitting contrived or altered information in any academic exercise. Example: making up data for an experiment, fudging data, citing nonexistent articles, contriving sources, etc.

D. Multiple Submissions Multiple submissions: submitting, without prior permission, any work submitted to fulfill another academic requirement.

E. Misrepresentation of academic records Misrepresentation of academic records: misrepresenting or tampering with or attempting to tamper with any portion of a students transcripts or academic record, either before or after coming to the University of Pennsylvania. Example: forging a change of grade slip, tampering with computer records, falsifying academic information on ones resume, etc.

F. Facilitating Academic Dishonesty Knowingly helping or attempting to help another violate any provision of the Code. Example: working together on a take-home exam, etc.

G. Unfair Advantage Attempting to gain unauthorized advantage over fellow students in an academic exercise. Example: gaining or providing unauthorized access to examination materials, obstructing or interfering with another students efforts in an academic exercise, lying about a need for an extension for an exam or paper, continuing to write even when time is up during an exam, destroying or keeping library materials for ones own use., etc.

* If a student is unsure whether his action(s) constitute a violation of the Code of Academic Integrity, then it is that students responsibility to consult with the instructor to clarify any ambiguities.