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

ESE532, Spring 2017 — Final — Monday, May 1

- Exam ends at 11:00AM; begin as instructed (target 9:00AM)
- Problems weighted as shown.
- Calculators allowed.
- Closed book = No text or notes allowed.
- Show work for partial credit consideration.
- 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:

Problem 1 (35 pts) (20–30 minutes)					Problem 2 (35 pts) (40–50 minutes)				Problem 3 (30 pts) (30–40 minutes)								
a	b	c	d	e	a	b	c	d	a	b	c	d	e	f	g	h	Total
5	5	5	10	10	5	5	20	5	3	6	3	3	3	3	6	3	100

1. Consider the following computation:

```
int x[256], y[256], w[256][256], s[3];

while (true) {
  for (i=0;i<256;i++) { // loop A
     x[i]=input();
     y[i]=0;
    }
  for (i=0;i<256;i++)   // loop B
   for (j=0;j<256;j++)
     y[j]+=x[i]*w[i][j];
  for (i=0;i<256;i++)   // loop C
     s[2]=max(y[i],s[2]);
     s[1]=max(s[1],s[2]);
     s[0]=max(s[0],s[1]);
  for (i=0;i<2;i++)    // loop D
     output(s[i]);
}
```

- The initial input() provides a new input every 100 ns
- multiply is 5 ns operation, pipelineable to start one multiply every 1 ns
- local memory access (load, store) to w[][], x[], y[], s[] is 1 ns
- add and `max` are 1 ns operations
- ignore loop and indexing costs for this problem

(a) How many operations (load, store, add, max, multiply) in each labelled (A, B, C, D) for loop?

Loop	A	B	C	D
Operations				

(b) Where is the bottleneck in this computation?

(c) What is the Amdahl's law speedup if only the bottleneck is accelerated?

(d) What parallelism can be exploited in this task (both within and among loops)? Describe all applicable options where appropriate.

Loop	Parallelism Options
among loops	
A	
B	
C	
D	

(e) Describe how you would speedup this task so that it can consume one input every 100 ns, limited only by the input rate.

This page nearly blank for pagination.

(Feel free to use for answer to 1e, but it is probably not necessary.)

2. Consider the following computation:

```
int Image[1024][1024], Model[3][1024][1024], wpixel[1024][1024];
boolean mpixel[1024][1024];
for (y=0;y<1024;y++)
  for (x=0;x<1024;x++) {
     int pixel=Image[y][x];
     int M0=Model[0][y][x];
     int M1=Model[1][y][x];
     int M2=Model[2][y][x];
     mpixel[y][x]=f(pixel,M0,M1,M2); // 10 mpy, 6 adds
     int mupdate=g(pixel,M0,M1,M2); // 4 mpy, 10 adds
     int updateval=h(pixel,M0,M1,M2); // 16 mpy, 8 adds
     Model[mupdate][y][x]=updateval;
     }
for (i=0;x<1024;i++) {
  if (mpixel[0][i]) mpixel[0][i]=1 else mpixel[0][i]=0;
  if (mpixel[i][0]) mpixel[i][0]=1 else mpixel[i][0]=0;
  }
for (y=1;y<1024;y++)
  for (x=1;x<1024;x++) {
     int imax=max(wpixel[y-1][x-1],max(wpixel[y-1][x],wpixel[y][x-1]));
     if (mpixel[y][x]) wpixel[y][x]=imax+1; else wpixel[y][x]=0;
     }
int xmax=0;
int ymax=0;
int maxval=0;
for (y=1;y<1024;y++)
  for (x=1;x<1024;x++)
     if (wpixel[y][x]>maxval) {maxval=wpixel[y][x]; xmax=x; ymax=y;}
int sy=max(0,ymax-16);
int sx=max(0,xmax-16);
for (y=sy;y<sy+16;y++)
  for (x=sx;x<sx+16;x++)
    output(Image[y][x]);
```

- Main memory is 256 M 32b ints; has a read and write latency of 100 ns, but can stream sequential data at 1 ns per cycle for blocks up to 512 words.
  `streamIn(MainAddr,LocalAddr,n)` – copy n≤512 32b ints to local memory in 100+n ns.
  `streamOut(LocalAddr,MainAddr,n)` – copy n≤512 32b ints to main memory in 100+n ns.
- Local memory is 4K 32b ints and has a read/write latency of 1 ns.
- multiply, add, max, compare each take 1 ns.
- As written Pixel, Model, mpixel, and wpixel live in main memory.
- Ignore loop and indexing costs for this problem.

(a) With no memory streaming operations or local memories,

i. estimate runtime

ii. identify bottleneck (which loop? memory or compute?) and support your answer using your runtime estimate

(b) Rewrite the code to localize and stream data You may combine loops where you find it beneficial.

i. Identify the local variables you define and how they are laid out in the local memory:

Address		
begin	end	Variable

ii. Show how the code is revised to use these local variables and stream fetch operations.

(continue 2.b.ii, showing your revised code.)

(c) What is the runtime of your optimized design?

(d) Where is the bottleneck now?

This page intentionally left nearly blank for pagination.
(or, additional code and calculations)

3. Consider a function from $A00$, $A01$, $A10$, $A11$, $B0$, $B1$ to $B2$, $B3$:

$$\begin{aligned}
t0 &= \frac{A00}{A10} && (1)\\
t1 &= \frac{A01}{A11} && (2)\\
t2 &= t1 * B1 && (3)\\
t3 &= B0 - t2 && (4)\\
t4 &= t1 * A10 && (5)\\
t5 &= A00 - t4 && (6)\\
t6 &= t0 * B1 && (7)\\
t7 &= B0 - t6 && (8)\\
t8 &= t0 * A11 && (9)\\
t9 &= A01 - t8 && (10)\\
t10 &= \frac{t3}{t5} && (11)\\
t11 &= \frac{t7}{t9} && (12)\\
t12 &= A20 * t10 && (13)\\
t13 &= A21 * t11 && (14)\\
t14 &= A30 * t10 && (15)\\
t15 &= A31 * t11 && (16)\\
B2 &= t12 + t13 && (17)\\
B3 &= t14 + t15 && (18)
\end{aligned}$$

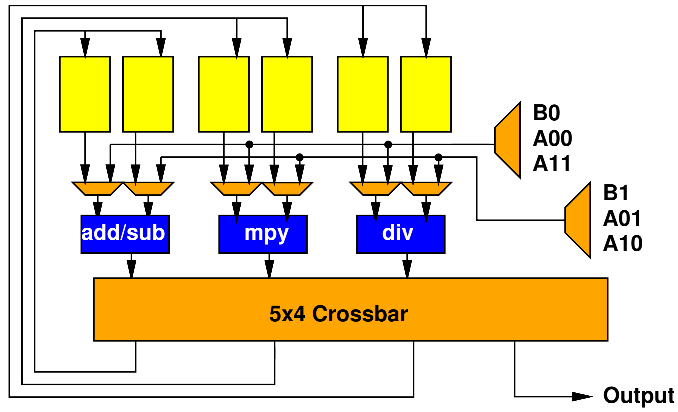
Assume:

- $A00$, $A01$, $A10$, $A11$, $B0$, $B1$ available on inputs at beginning of cycle
- output $B2$, $B3$ on designated output port
- $A20$, $A21$, $A30$, $A31$ already in operator memories; you choose which
- add/subtract, multiply, divide are single-cycle operations
- add/subtract unit costs 1 units of area
- multiply unit costs 10 units of area
- divide unit costs 10 units of area
- memory bank costs 5 units of area
- $i \times o$ crossbar costs $0.5 \cdot i \cdot o$ units of area
- word-wide pipeline register costs 0.5 units of area
- 2 or 3 input mux is 1 unit of area

(a) What is the critical path bound for this computation?

(b) Show a pipelined datapath for this operation.

(c) Estimate the area for the pipelined datapath.



(d) What is the resource bound for this computation on a VLIW datapath with a single add/subtract unit, a single multiplier, and a single divider (as shown)?

(e) Schedule the computation on the VLIW datapath with a single add/subtract unit, a single multiplier, and a single divider (as shown) to minimize computation cycles.
Mark each "operator" with the variable computed on the operator on that cycle; mark each "input" with the variable being stored into the data memories on each cycle (note: only one value can be stored into the data memories associated with an operator on each cycle).

Cycle	add/sub		multiply		divide		mux0	mux1	output
	operator	input	operator	input	operator	input			
0									
1									
2									
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									
13									
14									
15									

(f) Estimate the area of the VLIW datapath with a single add/subtract unit, a single multiplier, and a single divider (as shown).

(g) Using no more than 100 units of area, provision a customized VLIW datapath for this unit – how many operators of each type? total area?

Operator	add/sub	mpy	div	mux2 or mux3	Area
Number					

(h) Justify your choice of operators.

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.