EECS 151/251A Homework 9

Due Tuesday, Apr 20nd, 2021

For this Homework

Please include a short (1-2 sentence) explanation with your answer, unless otherwise noted.

Problem 1: Unsigned Equality Checker

Consider a circuit that compares two N-bit unsigned integers, A and B, and generates a single output bit f, which is equal to 1 if and only if $A \leq B$.

There are two common ways to implement this function. One is to use subtractors to determine the smaller number, but a circuit optimized for the comparison will be simpler and potentially faster. Such a circuit could also be implemented by computing $(A > B)'$, but there is a way to implement this function directly.

(a) Derive the simplest circuit that achieves the direct implementation of the $A \leq B$ function. This circuit will have $O(N)$ cost and delay. Draw such a circuit for $N = 4$. Show your work.

(b) Derive a strategy for improving the performance of the circuit from part (a) to have $O(\log(N)$ delay and $O(N)$ cost. Draw a circuit illustrating the structure of your design for $N = 8$.

Problem 2: Fibonacci Computer

Given a 1024x32b single-port synchronous read and write memory block, we'd like to design a circuit to compute 1024 Fibonacci elements and store to the memory block using the *minimal* number of cycles. The algorithm description of the problem is as follows.

```
for (i = 2; i < 1024; i = i + 1)
  fib[i] = fib[i - 1] + fib[i - 2]
```

The first two locations of the memory block are initialized to 0 and 1, respectively. Draw the block diagram of your hardware circuit including both datapath (computation, read/write to the memory block) and control (setting up clock enables, memory write enable, and MUX selectors). You may use any logic gates, MUXes, arithmetic units, comparators, registers as you like.

Problem 3: Modulo Scheduling

The lead designer of the chip you are working on has asked you to implement the following operation,

$$Y_i = \text{shift\_l}_2(\text{shift\_l}_2(A_i) + \text{shift\_l}_2(B_i) + C_i)$$

You are given the following blocks to work with:

- A memory block with 1 synchronous read port and 1 synchronous write port
- A multiplier block pipeline to two stages
- Single-cycle shifter that shifts 2 bits to the left
- Single-cycle adder

A, B, and C are stored in memory, and Y should be written back to memory once the computation is done.

(a) Using modulo scheduling, show how you would optimally schedule the computation with your datapath.

(b) Draw an extended version of the hardware utilization chart covering at least 3 iterations of the calculation.

Problem 4: `strncmp` Accelerator

```
int strncmp( const char* str1, const char* str2, size_t num );
```

`strncmp` is a C function that compares up to `num` characters of the string `str1` to those in the string `str2`. The function begins comparing the first character of each string. If they are equal, the function continues with each following pair until the characters differ, until it reaches a terminating null character, or until `num` characters have been matched, whichever occurs first.

The function returns the following values:

< 0	The first character that does not match has a lower value in `str1` than in `str2`
0	The contents of both strings are equal
> 0	The first character that does not match has a greater value in `str1` than in `str2`

For this problem assume that the accelerator connects to a 1 byte-wide memory block with a single asynchronous memory read/synchronous write port, and the memory is initialized with two strings. The strings are stored in standard C format with one ASCII character per byte, terminated by a null character. The address of each of the two strings are inputs to the accelerator, and the accelerator will output the return value according to the return behavior detailed previously. The accelerator will assert both the return value and a corresponding "valid" bit once the accelerator finishes computation.

Design the accelerator with the best performance (total time to compare). Start with a simple (less cost) design, and then optimize for performance.

(a) Write a RT Language description of a simple version of the hardware.

(b) Draw the datapath and the state transfer diagram of the controller.

(c) Write a RT Language description for a performance-optimized design.

(d) Describe the revised datapath for performance optimization.