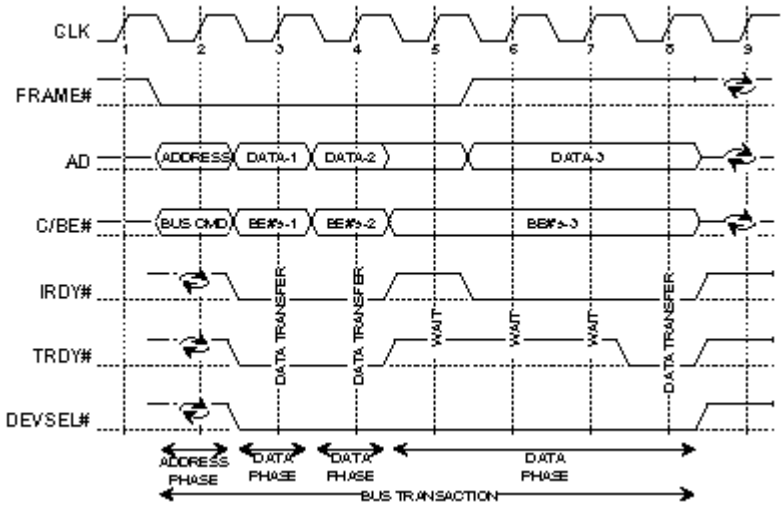
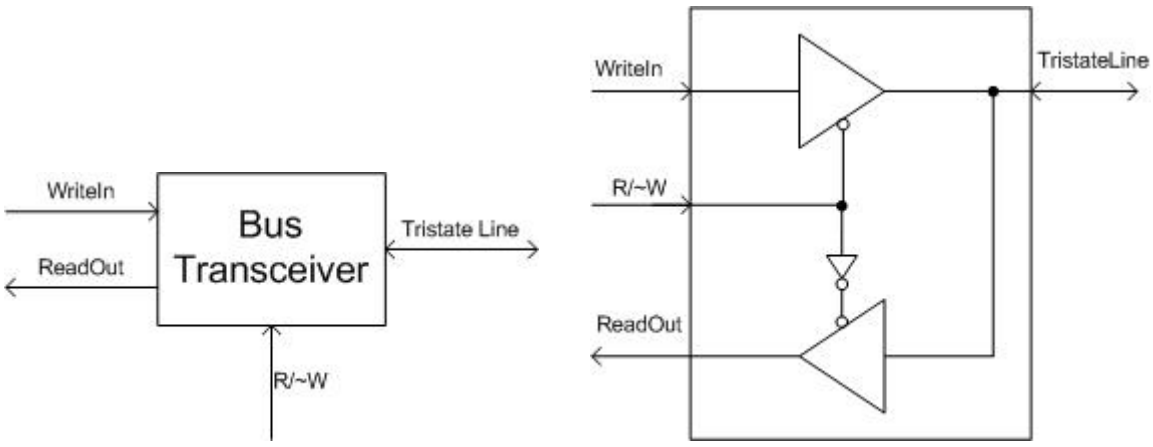
COMP3211/COMP9211 Computer Architecture
Week 6 Tutorial Exercises Solutions

Andhi Janapsatya
andhij@cse.unsw.edu.au

Q01. [S3.10] Draw and explain a timing diagram for a PCI write operation similar to Stalling, Figure 3.23 (Figure 1 in your assignment).



Q02. Describe in VHDL and simulate a device that writes to or reads from a tri-stated bus line. The interface of the device should be as follows:




```
library IEEE;
use IEEE.std_logic_1164.all;

entity bus_trcv is
  port (        WriteIn       : in    std_logic;
                ReadOut       : out   std_logic;
                RnW           : in    std_logic;
```

```
                TristateLine : inout std_logic);
end bus_trcv;

architecture behav of bus_trcv is
begin
  TristateLine <= WriteIn when RnW = '0' else 'Z';
   ReadOut <= TrisateLine when RnW = '1' else 'Z';
end behav;
```

Q03. [P&H Example p.665] Suppose we have a system with the following characteristics:

1. A memory and bus system supporting block access of 4 to 16 32-bit words.

2. A 64-bit synchronous bus clocked at 200 MHz, with each 64-bit transfer taking 1 clock cycle, and 1 clock cycle required to send an address to memory.

3. Two clock cycles needed between each bus operation. (Assume the bus is idle before an access.)

4. A memory access time for the first four words of 200 ns; each additional set of four words can be read in 20 ns. Assume that a bus transfer of the most recently read data and a read of the next four words can be overlapped.

Find the sustained bandwidth and the latency (time to complete the operation) for a read of 256 words for transfers that use 4-word blocks and for transfers that use 16-word blocks. Also compute the effective number of bus transactions per second for each case. Recall that a single bus transaction consists of an address transmission followed by data.

[Hint: The answer appears in P&H, pp.665 - 666]

Operation to be done:

256 32-bit words transfer with 4 word blocks.

Bus speed = 200MHz

Bus clock cycle = 1/200M = 5ns

- 1 clock cycle to send the address to memory.
- 200ns / (5ns/cycle) = 40 clock cycles to read memory
- 2 clock cycles to send data from memory (since 4 words = 128 bit and data bus is only 64 bit wide)
- 2 idle clock cycles between this transfer and the next

This gives a total of 45 clock cycles to transfer 4 words of data.

Number of bus trans. needed = 256/4 = 64 trans.

Transfer time = 64 * 45 cycles = 2880 clock cycles.

Latency = 2880 * 5ns = 14400 ns.

Bandwidth = 256 * 4 bytes / 14400ns = 71.11MB/sec.

Number of bus trans. per sec. = 64 trans. / 14400ns = 4.44 M trans./sec.

To transfer 256 32bit words with 16 word blocks

- 1 cycle required to send address
- 40 cycle required to read the first 4 words from memory
- 2 cycles to send data of the block and at the same time read of the next four words is started
- 2 idle cycles between transfers.

Total number of cycle needed to read each 16 word blocks = 57 cycles.

Latency = 4560ns

Bandwidth = 224.56MB/sec.

Number of bus trans. per sec. = 3.52M trans./sec.

Q04. [P8.12] The example above demonstrates that using larger block sizes results in an increase in the maximum sustained bandwidth that can be achieved. Under what conditions might a designer tend to favour smaller block sizes? Specifically, why would a designer choose a block size of 4 instead of 16 (assuming all characteristics are as identified in the example)?

A smaller block size would be preferable if the data that needs to be transferred exists in small blocks of 4 words. If a 16 words block transfer is to be used, for every memory access, the remaining 12 words in each transfer is wasted.

Q05. [P7.1] Describe the general characteristics of a program that exhibits very little temporal and spatial locality with regard to data accesses. Provide an example program (pseudocode is fine).

A program that does not or very rarely perform an operation on the same variable more than once.

Example:

```
for (p = 0; p < 100; p++) {
  for (q = 0; q < 100; q++) {
    a[q][p] = b[q][p];
  }
}
```

Q06. [P7.2] Describe the general characteristics of a program that exhibits very high amounts of temporal locality but very little spatial locality with regard to data accesses. Provide an example program (pseudocode is fine)

A program that perform operation on the same variable within a small amount of time but these variables are not next to each other in an array.

Example:

```
for (p = 0; p < 100; p++) {
  for (q = 0; q < 100; q++) {
    a[p] = b[q][p];
  }
}
```

Q07. [P7.3] Describe the general characteristics of a program that exhibits very little temporal locality but very high amounts of spatial locality with regard to data accesses. Provide an example program (pseudocode is fine).

A program that perform operation on arrays.

Example:

```
for (p = 0; p < 1000; p++) {
    a[p] = 2 * p;
}
```