Chris Thomas

Set #1 (Due Wed Oct 12)

1) Von Neumann vs Harvard architectures.

a) How does a Von Neumann architecture differ from a Harvard architecture?
In a Von-Neumann architecture the same memory and bus are used to store both data and instructions. The Harvard architecture stores machine instructions and data in separate memory units that are connected by different busses.

b) In what way is the PIC32MZEC MIPS processor a Von Neumann architecture from a hardware engineer's perspective?
It is a von Neumann architecture from a hardware engineers' perspective because it does not have overlapping addressing.

c) In what way is the PIC32MZEC MIPS processor a Harvard architecture from a hardware engineer's perspective?
It is a Harvard architecture from a hardware engineers' perspective because it has dedicated busses for instructions and data.

2) Product cost.

a) Why is the number of chips in a product usually the most important factor in the product's overall cost?
The number of chips (and other parts) is often the primary consideration in recurring costs, as it drives the other recurring cost factors.

b) When might using a processor that has more memory than we need lead to a less expensive system?
When that processor has peripherals, we will need built into the chip so the expense of space is reduced.

c) When, however, might not using the memory on a chip efficiently lead to a more expensive system? There are two ways this can happen, give both.
If you are not using the memory efficiently it could lead to you thinking you need a chip with more memory than what is needed.

3) The following instructions could be used to write to OC1CON (address: `0xBF844000`)

```
lui $v0, 0xBF84
sw $v1, 0x4000($v0)
```

because the sw instruction adds the (sign extended) 16 bit offset (`0x4000`) in the instruction to the value placed in the $v0 register (0xBF840000) by the lui instruction in order to form the address that is written to. (Note that the sw instruction sign extends its offset since, in general, we want to be able to reference addresses both above and below the address in $v0.) Lets suppose that we want to write some code to write to AD1CON2 (address: `0xBF84B004`) instead. Our first attempt yields:

```
lui  $v0, 0xBF84
sw  $v1, 0xB004($v0)
```

a) Remembering that the sw instruction will sign extend its 16 bit offset before adding it to $v0, what address actually gets written to?

0xBF83

b) How should the value in the lui instruction be corrected so that the right address gets written to? Give the corrected value.

0xBF85

c) In general, for what kinds of target addresses should this correction be applied?

Signed target addresses.

4) How, specifically, do the PIC32MZ ECG, ECH and ECM family members differ from each other?
The EGC doesn't have CAN 2.0B and only the ECM has Crypto.
5) What advantages does having multiple peripheral clocks give the PC32MZ processors over a processor that only has one peripheral clock?
Having multiple peripheral clocks allows you to maximize performance while controlling power consumption.
6) Processors that have many on-chip peripherals will likely need to share I/O pins between peripherals. Those that do not have a PPS typically assign several peripherals to each pin.

a) How does having a fixed assignment of several peripherals to an I/O pin restrict the system designer?
Having a fixed assignment of several peripherals to an I/O is restricting since it limits choice. Also, since it is fixed you cannot have an I/O device hooked up somewhere else that it may be needed.
b) What does the PPS (Peripheral Pin Select) function of MZ processors do?
PPS allows the developer to choose from a number of output and input pins to connect to the digital peripheral.
c) So, what main advantage does a processor that has a PPS have over one that does not have this feature?
Flexibility and puts more control in the designer's hands.

7) The data sheet informs us that program flash memory (PFM) can be accessed

without any wait states (additional SYSCLK clock cycles added to a fetch (i.e. an instruction fetch) from PFM) if SYSCLK <= 83MHz and ECC is not enabled for the PFM. If ECC is enabled, the PFM can be accessed without wait states only if SYSCLK <= 66MHz. From this information we can compute (closely enough) what the actual access time of the PFM is. Note that when there are no wait states, instructions are fetched one per SYSCLK cycle from the PFM. With each wait state added, there will be an additional clock cycle.

a) What is the PFM access time in ns if PFM ECC is not enabled? Give your answer in ns to 3 decimal points.

12.048ns

b) What is the PFM access time in ns if PFM ECC is enabled? Give your answer in ns to 3 decimal points.

15.152ns

c) Why is the access time longer if ECC is enabled? What advantage is ECC?

The ECC is error correct code it takes more time since it does any extra check for accuracy. The advantage is reducing errors.

d) If the SYSCLK frequency is 150MHz, how many wait states are required assuming ECC is enabled? **__2__** How many are required if ECC is not enabled? **__1__**

e) What things (there are several) can be done in PIC2MZ processors to speed up access to instructions?

Enable ECC instruction cache and enable prefetch module for devices with L1 CPU cache.

Note that RAM accesses in PIC32MZ processors don't need any wait states no matter what the SYSCLK frequency is.