Feedback on Quiz # 0 (Formative)

The **Quiz # 0** was composed of **10** questions, which were randomly selected from a Question Bank of 20 questions. Answers and feedback comments for all of the questions are given below:

Q01	Consider the binary number $(11101010110)_2$ What would be its equivalent representation in the octal number system?
(a)	**$(3526)_8$**
(b)	$(6526)_8$
(c)	$(7254)_8$
(d)	$(7252)_8$

Feedback:
To obtain the octal equivalent, we take numbers in groups of 3, from right to left as:

011 101 010 110

3 5 2 6 = $(3526)_8$

Q02	Consider the octal number $(17317)_8$ What would be its equivalent representation in the hexadecimal number system?
(a)	**$(1ECF)_{16}$**
(b)	$(1DBF)_{16}$
(c)	$(7B3C)_{16}$
(d)	$(F678)_{16}$

Feedback:
To convert an octal number to its decimal equivalent follow these steps:

$(17317)_8$

= 001 111 011 001 111 (groups of 3 binary numbers from right to left)

= 001 1110 1100 1111 (groups of 4 binary numbers from right to left)

= 1 E C F

Therefore, the answer is $(1ECF)_{16}$.

Q03	The range of numbers that can be stored in a ten bit register (using 2's complement) is ____________________?
(a)	**-512 to +511**
(b)	-511 to +512
(c)	-1024 to +1023
(d)	-1023 to +1024
(e)	

Feedback:

The minimum number that we can represent in 10-bits is 1000000000, where the most significant bit represents the -ve sign. To find out its equivalent value in decimal, we can take 2's complement.

Take 1's complement of 1000000000 = 0111111111

Add 1 to 0111111111 = 0111111111+1 = $(1000000000)_2$ = $(512)_{10}$

So we can represent $(-512)_{10}$.

The maximum value that we can represent in 10-bits = 0111111111 = $(511)_{10}$

In general, for n bits, the range is: -2^{n-1} --> $2^{n-1} - 1$

Q04	The range of numbers that can be stored in an eight bit register (using 2's complement) is ____________________?
(a)	-128 to +127
(b)	-127 to +128
(c)	-256 to +255
(d)	-255 to +256
(e)	

Feedback:

The minimum number that we can represent in 8-bits is 10000000, where the most significant bit represents the -ve sign. To find out its equivalent value in decimal, we can take 2's complement.

Take 1's complement of 10000000 = 01111111

Add 1 to 01111111 = 01111111+1 = $(10000000)_2$ = $(128)_{10}$

So we can represent $(-128)_{10}$.

The maximum value that we can represent in 8-bits = 01111111 = $(127)_{10}$

In general, for n bits, the range is: -2^{n-1} --> $2^{n-1} - 1$

Q05	If the clock speed of a processor is increased from 1GHz to 2GHz:
(a)	**The number of instructions executed per second will increase but by less than 2 times**
(b)	The number of instructions executed per second will be doubled
(c)	The number of instructions executed per second will stay the same
(d)	The CPU will start fetching instructions from the memory at double speed

Feedback:

The impact of clock speed on instruction execution is significant but it will not be exactly twice, as many factors, such as pipelining, branch instructions, procedure calls, will lower the CPU performance. Therefore, increasing the clock speed from 1GHz to 2GHz will increase the number of instructions executed per second, but this will be less than 2 times.

Q06	To reduce the memory access time, we generally make use of:
(a)	Solid State Drive (SSD)
(b)	**Cache Memory**
(c)	Higher Capacity RAMs
(d)	More CPU Registers

Feedback:

By using the cache memory, we can reduce the speed of memory access by a factor of approximately 10.

Q07	Which of the following statements are relevant to the Instruction Fetch (IF) stage?
(a)	**The Program Counter (PC) is incremented**
(b)	The value in the Memory Data Register is written to the address in the Memory Address Register
(c)	The data values are retrieved from memory
(d)	**The instruction at the Memory Address Register is read from RAM**

Feedback:

The PC contains the address of the memory location that has the next instruction which has to be fetched. This address is then passed from the PC to the memory address register via the address bus.

The instruction at the memory location (address) contained in MAR is then read from RAM and placed into the control unit. The value in the PC is then incremented by 1 so that it now points to the next instruction to be fetched.

Q08	Which of the following statements are relevant to the Instruction Decode (ID) stage?
(a)	**The source and destination operand addresses are computed**
(b)	The instruction at the Memory Address Register is read from RAM
(c)	The Program Counter value is copied to the Memory Address Register
(d)	**Decoder sets up the operation in the ALU**
Feedback: In the instruction decode (ID) stage, the instruction is decoded by the control unit. The decoder pulls apart the instruction and determines what operation will be set up in the ALU. The source and destination operand addresses are computed.	

Q09	Which of the following statements are false:
(a)	MIPS R4000 has a modified Harvard architecture
(b)	**In the von Neumann architecture, the memory is organised in two separate blocks**
(c)	A Harvard machine has separate memory units for data and program
(d)	**A von Neumann machine can access program memory and data memory simultaneously**
Feedback: In a system with a von Neumann architecture, the memory is organised in a single block, therefore programs and data are stored in the same memory (unlike the Harvard architecture) and cannot be accessed at the same time. This means that a CPU cannot simultaneously read an instruction or write data from/to the memory.	

Q10	Which of the following stages are not valid in the Fetch/Execute cycle?
(a)	**Register Return (RR)**
(b)	Instruction Decode (ID)
(c)	Instruction Fetch (IF)
(d)	Data Fetch (DF)
(e)	**Data Store (DS)**
Feedback: Register Return (RR) and Data Store (DS) are not valid fetch execute stages. The last stage is Result Return (RR), which returns the result to the memory.	

Q11	What is the result of the following MIPS instruction: **and $4, $5, $0**
(a)	We don’t know
(b)	**Store 0 into register $4**
(c)	Store $(101)_2$ into register $4
(d)	Store $(9)_{10}$ into register $0
Feedback: Register $0 holds only zeroes, therefore the result of the logical AND operation $5 & $0 will be 0, regardless of the value stored in $5.	

Q12	What is the result of the following MIPS instruction: **sub $30, $15, $10**
(a)	Store $(5)_{10}$ in register $30
(b)	**We don’t know**
(c)	Store $(15)_{10}$ in register $10
(d)	Store $(101)_2$ in register $30
Feedback: The instruction subtracts the contents of register $10 from the content of register $15 and stores the result in register $30. Therefore, we cannot tell the result of the instruction without knowing the contents of the registers.	

Q13	What is the value stored in ‘z’ after the execution of the following MIPS assembly code, if y=6? lw $5, &y li $6, 5 blt $5, $6, L1 addi $7, $5, 4 j L2 L1: addi $7, $5, 2 L2: sw $7, &z

(a)	z = 2
(b)	**z = 10**
(c)	z = 4
(d)	z = 8

Feedback:

The blt (branch on less than) instruction compares two registers and jumps to L1 if the content of register $5 is less than the content of register $6. The equivalent Java code is an if-else statement:

```
if (y < 5) z = y + 2;
   else z = y + 4;
```

Knowing that y = 6, the result will be z = y + 4 = 6 + 4 = 10.

Q14 What is the value stored in ‘z’ after the execution of the following MIPS assembly code, if x=5?

```
li $6, 1
li $7, 2
lw $8, &x
L1:  bgt $7, $8, L2
       mult $6, $6, $7
       addi $7, $7, 1
       j L1
L2: sw $6, &z
```

(a)	z = 24
(b)	z = 6
(c)	z = 720
(d)	**z = 120**

Feedback:

The bgt (branch on greater than) instruction compares two registers and jumps to L2 if the content of register $7 is greater than the content of register $8. The equivalent Java code is a for loop:

```
 z = 1;
for (i = 2; i  <= x; i++){
   z = z * i;
}
```

Knowing that x = 5 and the index i starts from 2, the result will be z = 1 * 2 * 3 * 4 * 5 = 120.

Q15	A processor is operating at 30MHz. Each instruction takes a minimum of 6 cycles to execute. The processor has a six stage pipeline. If a program starts execution at time 0, what is the theoretical maximum number of instructions that will have completed their execution at the end of 1 millisecond?
(a)	30,000
(b)	5,000
(c)	4,995
(d)	**29,995**
Feedback: Speed = 30MHz = 30,000,000 cycles/second = 30,000 cycles / millisecond With pipelining: Number of clock cycles taken by the first instruction = 6 cycles. After the first instruction has completely executed, one instruction comes out per cycle. So, number of cycles taken by each remaining instruction = 1 cycle. Number of executed instructions = First Instruction (1) + remaining instructions (30,000 cycles/ms – 6 cycles) = 1 + 29,994 = 29,995 instruction/ms In practice, there are factors that mean that this theoretical maximum is never reached.	

Q16	A processor is operating at 20MHz. Each instruction takes a minimum of 5 cycles to execute. The processor has a five stage pipeline. If a program starts execution at time 0, what is the theoretical maximum number of instructions that will have completed their execution at the end of 1 millisecond?
(a)	20,000
(b)	4,000
(c)	3,996
(d)	**19,996**
Feedback: Speed = 20MHz = 20,000,000 cycles/second = 20,000 cycles / millisecond With pipelining: Number of clock cycles taken by the first instruction = 5 cycles. After the first instruction has completely executed, one instruction comes out per cycle. So, number of cycles taken by each remaining instruction = 1 cycle. Number of executed instructions = First Instruction (1) + remaining instructions (20,000 cycles/ms – 5 cycles) = 1 + 19,995 = 19,996 instruction/ms In practice, there are factors that mean that this theoretical maximum is never reached.	

Q17	A 32-bit computer system has 4 GB of memory installed in it, which is represented by addresses $(00000000)_{16}$ - $(FFFFFFFF)_{16}$. However, the system programmer has been told that they can only use memory from $(4A400000)_{16}$ to $(B9C00000)_{16}$. The memory from $(00000000)_{16}$-$(4A400000)_{16}$ is unavailable and the memory above $(B9C00000)_{16}$ is reserved for the Operating System, hence cannot be used. How much memory is available to the system programmer?
(a)	1644 MB
(b)	**1784 MB**
(c)	1848 MB
(d)	1988 MB

Feedback:

```
  B9C00000
- 4A400000
---------------
  6F800000
```

$(6F800000)_{16} = (1870659584)_{10}$

1870659584 / (1024*1024) = 1784 MB

Q18	A 28-bit computer system has 256 MB of memory installed in it, which is represented by addresses $(0000000)_{16}$ - $(FFFFFFF)_{16}$. However, the system programmer has been told that he can only use memory from $(A8B0000)_{16}$ to $(E7B0000)_{16}$. The memory from $(0000000)_{16}$-$(A8B0000)_{16}$ is unavailable and the memory above $(E7B0000)_{16}$ is reserved for the Operating System, hence cannot be used. How much memory is available to the system programmer?
(a)	**63 MB**
(b)	193 MB
(c)	64 MB
(d)	65536 KB
(e)	**64512 KB**

Feedback:

```
  E7B0000
- A8B0000
----------
  3F00000
```

$(3F00000)_{16} = (66060288)_{10}$

66060288/ (1024*1024) = 63 MB (== 64512 KB)

<table>
<tr><td>Q19</td><td>A hypothetical computer stores real numbers in floating point format in 8-bit words.
The first bit is used for the sign of the number (1 is negative), the second bit for the sign of the exponent, the next two bits for the magnitude of the exponent, and the next four bits for the magnitude of the mantissa.
Represent e = 2.718 in the 8-bit floating point binary format.
(You can assume that we don't need to add offset to the exponent part)</td></tr>
<tr><td>(a)</td><td>**(00010110)$_2$**</td></tr>
<tr><td>(b)</td><td>(00011011)$_2$</td></tr>
<tr><td>(c)</td><td>(00010101)$_2$</td></tr>
<tr><td>(d)</td><td>(00100101)$_2$</td></tr>
<tr><td colspan="2">**Feedback:**
$(2)_{10} = (10)_2$
Finding $(0.718)_{10} = (?)_2$
0.718 × 2 = **1**.436
0.436 × 2 = **0**.872
0.872 × 2 = **1**.744
0.744 × 2 = **1**.488   (we will round this bit in the next step)
$(0.718)_{10} \approx (0.1011)_2$
$(0.101)_2 \approx (0.110)_2$ (We round the last bit, so we keep only three bits here, which will become 4 bits in the normal form)
$(2.718)_{10} \approx (10.110)_2 = (1.0110)_2 \times 2^1$
(You can see above that once we convert the number into standard form, we get four bits after the binary point)
Sign number bit = 0
Sign exponent bit = 0
Bits in exponent = 01
Bits in mantissa = 0110
The final floating point representation is $(00010110)_2$.</td></tr>
</table>

Q20	A hypothetical computer system stores real numbers in floating point format using 8-bit words. The first bit is used for the sign of the number (1 is negative), the second bit is used for the sign of the exponent, the next two bits for the magnitude of the exponent, and the next four bits for the magnitude of the mantissa. Convert the floating point representation given by binary string $(10100111\ )_2$ into its equivalent decimal representation? (You can consider that there is no offset added to the exponent part)
(a)	**-5.75**
(b)	-2.875
(c)	-5.875
(d)	-2.75

Feedback:

The number is 10100111

Sign number bit = 1, so sign of number is negative.

Sign exponent bit = 0, so sign of exponent is positive

Bits in exponent = 10

$(\ 10\ )_2 = 1 \times 2^1 + 0 \times 2^0 = (2)_{10}$

Bits in mantissa = 0111, so it can be written in standard form as: 1.0111×2^2

which is equivalent to = 101.11

$(\ 101.11\ )_2 = 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2}$

$= (5.75)_{10}$

As the sign of number is negative, the binary string represents

$= (-5.75)_{10}$