1. (10 points) a. (6 points) Draw a word-wide HEXADECIMAL content of memory cells corresponding to the following sequence of assembler directives:

```
ORG $3700
A DS.W 2
P EQU 20
V1 DC.B 10,45
V2 DC.L $40302010
V3 DC.B P-2
V4 DS.L 3
```

<table>
<thead>
<tr>
<th>Address [Hex]</th>
<th>Content &lt;15:0&gt; [Hex]</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>3700</td>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>3702</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3704</td>
<td>A, 3D</td>
<td>V1</td>
</tr>
<tr>
<td>3706</td>
<td>4030</td>
<td>V2</td>
</tr>
<tr>
<td>3708</td>
<td>2010</td>
<td></td>
</tr>
<tr>
<td>3710</td>
<td>0018</td>
<td>V3</td>
</tr>
<tr>
<td>3712</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3714</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3716</td>
<td></td>
<td>V4</td>
</tr>
<tr>
<td>3718</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3720</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3722</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

b. (4 points) The Motorola 68000 microprocessor has:

(i) ___3 (byte, word, longword)___ sizes for data operations

(ii) ___A7___ functioning as the stack pointer

(iii) register size is ___32___ bits

(iv) status bit N represents ___negative__, It is set when ___the leftmost bit of the operand is 1__.
2. (1 point) __Communication__ is the hardest problem.

3. (1 point) __Overflow__ occurs when a number can’t be represented in a computer.

4. (1 point) The addressing mode used when the contents of an address register specify the address of the operand is known as __register indirect__ addressing.

5. (15 points) Write a subroutine using 68K assembly that sums the elements of a word array. Assume that parameters for subroutine `void sum_array(int *a, int n, int *sum)`, the starting address, array size, and the address of sum are prepared on the stack in the main program. Use registers for local variables in the subroutine.

    ```assembly
    Sum_array  MOVEM.L  A0-A1/D0-D1,-(A7)
    MOVEA.L    14(A6),A0  
    MOVE.W     12(A6),D0  
    MOVEA.L    8(A6),A1   
    ADD.W      D0,D0     
    CLR.W      (A1)      
    Loop       MOVE.W    (A6,D0),D1 
    ADD.W      D1,(A1)   
    SUBQ.W     #2,D0     
    BNE        Loop      
    MOVEM.L    (A7)+,A0-A1/D0-D1 
    ```

6. (1 point) The addressing mode used when the operand appears in the instruction itself is known as __immediate__ addressing.

7. (1 point) The Motorola 68000 instruction set is a __two__ operand instruction set.

8. (20 points) For the given fraction of an assembly language program:

    (a) MOVE.L #1, D1  12 cycles
    (b) L2 MOVE.L -2(A6),D0  16 cycles
    (c) MULU.L D0,D1  70 cycles
    (d) ADDQ.W #1,-2(A6)  16 cycles
    (e) L1 ADDQ.W #1,D1  4 cycles
    (f) CMP.W #20,D1  8 cycles
    (g) BHI.S L2  8 cycles not taken/10 cycles taken
    (h) RTS  16 cycles

   a. (10 points) Find the total execution time of the given program on an 8 MHz 68000 microprocessor.

The number cycles required for each instruction is shown alongside the instructions. Now, how many times is the loop executed? D1 is the loop control variable that is initialized to 1 with the first instruction. Assuming −2(A6) is initialized to 1, DO will start at 1.

<table>
<thead>
<tr>
<th>Value of D1</th>
<th>Value of D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial</td>
<td>1</td>
</tr>
<tr>
<td>First iteration CMP</td>
<td>2</td>
</tr>
<tr>
<td>Second iteration CMP</td>
<td>5</td>
</tr>
<tr>
<td>Third iteration CMP</td>
<td>16</td>
</tr>
</tbody>
</table>
Fourth iteration CMP

Of the four iterations, three times the branch is taken and the last time the branch is not taken. Instructions (a) and (h) are executed only once. Three times instruction (g) takes 10 cycles to execute while it takes 8 cycles the fourth time. So, the total number of cycles is 12(a) + 16(h) + 3(16(b) + 70(c) + 16(d) + 4(e) + 8(f) + 10(g)) + (16(b) + 70(c) + 16(d) + 4(e) + 8(f) + 8(g)) = 12 + 16 + 3(124) + 122 = 28 + 372 + 122 = 522 cycles

Execution time = Number of cycles * time/cycle = 522 cycle * 125 ns = 65.25 us

b. (5 points) Calculate the average CPI (number of clocks per instruction). Number of cycles = 522, Number of instructions = 2 + 4(6) = 26, CPI = 522/26 = 20.08

c. (5 points) Calculate the MIPS rate

\[
MIPS = \frac{\# \text{Instructions}}{\text{Time} \times 10^6} = \frac{26}{65.25 \times 10^{-6} \times 10^6} = 0.86
\]

9. (25 points) Represent the state of the stack and values of SP and A6 during the execution of the following program:

```c
int power (long unsigned int *base, unsigned int *exponent, long unsigned int *product);

int main()
{
    long unsigned int a, b;
    unsigned int i;
    a = 2;
    i = 2;
    power(&a, &i, &b);
    return 0;
}

int power (long unsigned int *base, unsigned int *exponent, long unsigned int *product)
{
    unsigned int i = 1;
    *product = 1;
    while (i <= *exponent)
    {
        *product = *product * *base;
        i++;
    }
    return 0;
}
```
Code generated by the cross compiler is given below:

```assembly
*5 int main()
* Variable a is at -4(A6)
* Variable b is at -8(A6)
* Variable i is at -10(A6)
LINK A6,#-10

MOVEQ.L #2,D1
MOVE.L D1,-4(A6)
MOVE #2,-10(A6)
PEA.L -8(A6)
PEA.L -10(A6)
PEA.L -4(A6)
JSR power
CLR D0
UNLK A6
RTS

*18 int power (long unsigned int *base,
unsigned int *exponent, long unsigned int *product)
* Parameter base is at 8(A6)
* Parameter exponent is at 12(A6)
* Parameter product is at 16(A6)
* Variable i is at -2(A6)
LINK A6,#-2
MOVE #1,-2(A6)
MOVEQ.L #1,D1
MOVEA.L 16(A6),A4
MOVE.L D1,(A4)
BRA L1
L2 MOVEA.L 16(A6),A4
MOVE.L (A4),D1
MOVEA.L 8(A6),A0
MULU.L (A0),D1
MOVE.L D1,(A4)
ADDQ #1,-2(A6)
L1 MOVEA.L 12(A6),A4
MOVE -2(A6),D1
CMP (A4),D1
BLS.S L2
CLR D0
UNLK A6
RTS
```
68000 Registers (all values are hex unless otherwise noted)

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>D0</td>
<td>D1</td>
<td>D2</td>
<td>D3</td>
<td>D4</td>
<td>D5</td>
<td>D6</td>
</tr>
<tr>
<td>01234567</td>
<td>89ABCDEF</td>
<td>0001002D</td>
<td>ABCD7FFF</td>
<td>33449127</td>
<td>AAAAAAAA</td>
<td>ABCD0003</td>
</tr>
<tr>
<td>A0</td>
<td>A1</td>
<td>A2</td>
<td>A3</td>
<td>A4</td>
<td>A5</td>
<td>A6</td>
</tr>
<tr>
<td>00007028</td>
<td>00001000A</td>
<td>A7</td>
<td>00010080</td>
<td>00010020</td>
<td>0000FFFA</td>
<td>00010000</td>
</tr>
<tr>
<td>XNZVC</td>
<td>b10100</td>
<td>PC</td>
<td>00004000</td>
<td>b10100</td>
<td>00004000</td>
<td></td>
</tr>
</tbody>
</table>

Main memory (all values are hex unless otherwise noted)

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>007000</td>
<td>007020</td>
<td>007030</td>
<td>007040</td>
<td>007050</td>
<td>007060</td>
<td>007070</td>
</tr>
<tr>
<td>007001</td>
<td>007021</td>
<td>007031</td>
<td>007041</td>
<td>007051</td>
<td>007061</td>
<td>007071</td>
</tr>
<tr>
<td>007002</td>
<td>007022</td>
<td>007032</td>
<td>007042</td>
<td>007052</td>
<td>007062</td>
<td>007072</td>
</tr>
<tr>
<td>007003</td>
<td>007023</td>
<td>007033</td>
<td>007043</td>
<td>007053</td>
<td>007063</td>
<td>007073</td>
</tr>
<tr>
<td>007004</td>
<td>007024</td>
<td>007034</td>
<td>007044</td>
<td>007054</td>
<td>007064</td>
<td>007074</td>
</tr>
<tr>
<td>007005</td>
<td>007025</td>
<td>007035</td>
<td>007045</td>
<td>007055</td>
<td>007065</td>
<td>007075</td>
</tr>
<tr>
<td>007006</td>
<td>007026</td>
<td>007036</td>
<td>007046</td>
<td>007056</td>
<td>007066</td>
<td>007076</td>
</tr>
<tr>
<td>007007</td>
<td>007027</td>
<td>007037</td>
<td>007047</td>
<td>007057</td>
<td>007067</td>
<td>007077</td>
</tr>
<tr>
<td>007008</td>
<td>007028</td>
<td>007038</td>
<td>007048</td>
<td>007058</td>
<td>007068</td>
<td>007078</td>
</tr>
<tr>
<td>007009</td>
<td>007029</td>
<td>007039</td>
<td>007049</td>
<td>007059</td>
<td>007069</td>
<td>007079</td>
</tr>
<tr>
<td>007010</td>
<td>007030</td>
<td>007040</td>
<td>007050</td>
<td>007060</td>
<td>007070</td>
<td></td>
</tr>
<tr>
<td>007011</td>
<td>007031</td>
<td>007041</td>
<td>007051</td>
<td>007061</td>
<td></td>
<td></td>
</tr>
<tr>
<td>007012</td>
<td>007032</td>
<td>007042</td>
<td>007052</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>007013</td>
<td>007033</td>
<td>007043</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>007014</td>
<td>007034</td>
<td>007044</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>007015</td>
<td>007035</td>
<td>007045</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>007016</td>
<td>007036</td>
<td>007046</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>007017</td>
<td>007037</td>
<td>007047</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>007018</td>
<td>007038</td>
<td>007048</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>007019</td>
<td>007039</td>
<td>007049</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00701A</td>
<td>00703A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00701B</td>
<td>00703B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00701C</td>
<td>00703C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00701D</td>
<td>00703D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00701E</td>
<td>00703E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00701F</td>
<td>00703F</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

10. (30 points) What is the effect of applying each of the following 68000 instructions assuming the initial conditions shown? Represent modified internal registers, memory locations and condition codes.

(a) MOVE.B (A1)+, D0
CMPI.B #$5A, D0

\[
\begin{array}{cccccc}
\text{X} & \text{N} & \text{Z} & \text{V} & \text{C} \\
\hline
1 & 0 & 0 & 0 & 0 \\
\end{array}
\]

D0 = 01234579
A1 = 00007029

(b) CMPM.W (A1)+, (A3)+

\[
\begin{array}{cccccc}
\text{X} & \text{N} & \text{Z} & \text{V} & \text{C} \\
\hline
1 & 1 & 0 & 0 & 0 \\
\end{array}
\]

(A1) = 79E7, A1 = 0000702A
(A3) = FCFF, A3 = 00007036
(c) \texttt{CLR.L 5(A3,D6.W)}

\begin{align*}
A3 &= 00007034, \quad D6.W = 0003 \\
\text{MEM}[703C] &= 00 \\
\text{MEM}[703D] &= 00 \\
\text{MEM}[703E] &= 00 \\
\text{MEM}[703F] &= 00
\end{align*}

\begin{tabular}{|c|c|c|c|c|c|}
\hline
X & N & Z & V & C \\
\hline
1 & 0 & 1 & 0 & 0 \\
\hline
\end{tabular}

(d) \texttt{BEQ \#100}

\begin{align*}
\text{PC} &= \#100
\end{align*}

\begin{tabular}{|c|c|c|c|c|c|}
\hline
X & N & Z & V & C \\
\hline
1 & 0 & 1 & 0 & 0 \\
\hline
\end{tabular}

(e) \texttt{AND.B 6(A4), \$1001C}

\begin{align*}
6(A4) &= \text{MEM}[2046] = 49 \\
\$1001C &= \text{MEM}[1001C] = 33 \\
\text{MEM}[1001C] &= 01
\end{align*}

\begin{tabular}{|c|c|c|c|c|c|}
\hline
X & N & Z & V & C \\
\hline
1 & 0 & 1 & 0 & 0 \\
\hline
\end{tabular}

(f) \texttt{BCHG.W \#10, D3}

\begin{align*}
D3 &= \text{ABCD7DFF}
\end{align*}

\begin{tabular}{|c|c|c|c|c|c|}
\hline
X & N & Z & V & C \\
\hline
1 & 0 & 1 & 0 & 0 \\
\hline
\end{tabular}

(g) \texttt{ROXR.B \#4, D5}

\begin{align*}
D5 &= \text{AAAAAAABA}
\end{align*}

\begin{tabular}{|c|c|c|c|c|c|}
\hline
X & N & Z & V & C \\
\hline
1 & 0 & 1 & 0 & 0 \\
\hline
\end{tabular}

(h) \texttt{ADD.W \$FE(A2), (A3)+}

\begin{align*}
\$FE(A2) &= 10008 \\
A3 &= 7036 \\
\text{MEM}[10008] &= 1714 \\
\text{MEM}[7034] &= \text{FCFF} \\
\text{MEM}[7034] &= 1413 \\
A3 &= 7036
\end{align*}

\begin{tabular}{|c|c|c|c|c|c|}
\hline
X & N & Z & V & C \\
\hline
1 & 1 & 0 & 0 & 0 \\
\hline
\end{tabular}

\begin{tabular}{|c|c|c|c|c|c|}
\hline
X & N & Z & V & C \\
\hline
1 & 0 & 0 & 0 & 0 \\
\hline
\end{tabular}