Tolaesh Mengeste

CSS 422 Hard Ware

**Q1. Convert the following 68K assembly language instructions to the machine codes. (2 points)**

1) MOVE.W        D1, $0000A000

Size is byte = **0011**

Destination is  **$0000A000**

Register for destination 000

Mode for destination 111

Source is **D1**

Registration for source 001

Mode for source. 000

Therefore, together we will have

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | **0** | **1** | **1** | 000 | 111 | 000 | 001 |

When we change to Hexadecimal number, we get **33C1**

The destination address is **33C1 0000A000**

2) MOVE.B        $42A7, (A1) +

Size is byte = **0011**

Destination is (**A1)+**

Register for destination 001

Mode for destination 011

Source is **$42A7**

Registration for source 000

Mode for source. 111

Therefore, together we will have

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | **0** | **1** | **1** | 001 | 011 | 111 | 000 |

When we change to Hexadecimal number, we get **12F8**

The destination address is **12F8 $42A7**

3) ADD.L            D7, D0

Opcode = **1101**

Destination is **D0**

Register for destination 000

Opmode is

<ea> + D0 ->D0 010

Effective address **D7**

Registration 000

Mode 111

Therefore, together we will have

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | **0** | **1** | **1** | 000 | 010 | 111 | 000 |

When we change to Hexadecimal number, we get **DO87**

There for the machine language is **DO87**

4) MOVEA.L       D3, A0

Size is long = **0010**

Destination is **A0**

Register for destination 000

Mode for destination 001

Source is **D3**

Registration for source 000

Mode for source. 011

Therefore, together we will have

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | **0** | **1** | **1** | 000 | 001 | 011 | 000 |

When we change to Hexadecimal number, we get **2043**

The destination address is **2043**

**Q2. For each of the operations below, evaluate the value in D0 and the state of the CCR after completing the operation.**

1. ASL.B #2, D0

**XNZVC=00000 and D0=$C1A8E392**

**After I executed the operation by make the XNZVC and D0, I found that**

**D0 = FFFFFFC and XNVZVC = 11001**

1. ASL.L #5, D0

**XNZVC=00000 and D0=$C1A8E392**

**After I executed the operation by make the XNZVC and D0 as give, I found that D0 = FFFFFF0 and XNVZVC = 11001**

1. LSR.B #4, D0

**XNZVC=00000 and D0=$C1A8E392**

**After I executed the operation by make the XNZVC and D0 as give, I found that D0 = FFFFF0F and XNVZVC = 10001**

1. ROR.W #2, D0

**XNZVC=00000 and D0=$C1A8E392**

**After I executed the operation by make the XNZVC and D0 as give, I found that D0 = FFFFF0F and XNVZVC = 10001**

**Q3. Pattern Finding and Cumulative program. (5 points)**

**Q4. Decode a floating-point number. (5 points)**