The University of Alabama in Huntsville
Electrical & Computer Engineering Department
CPE/EE 421/521 01
Test 1 Solution
Fall 2004

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
```

Address [Hex]	**Content <15:0> [Hex]**	**Comment**
3700		A
3702		
3704	A, 3D	V1
3706	4030	V2
3708	2010	
3710	0018	V3
3712		V4
3714		
3716		
3718		
3720		
3722		

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.

```
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.

	Value of D1	Value of D0
Initial	1	1
First iteration CMP	2	2
Second iteration CMP	5	3
Third iteration CMP	16	4

Fourth iteration CMP 65 5

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

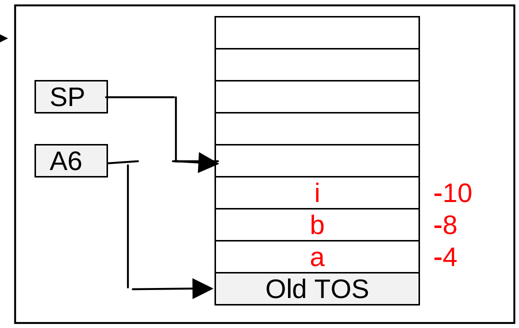
$$MIPS = \frac{\#\,Instructions}{Time \times 10^6} = \frac{26}{65.25 \times 10^{-6} * 10^6} = 0.86$$

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

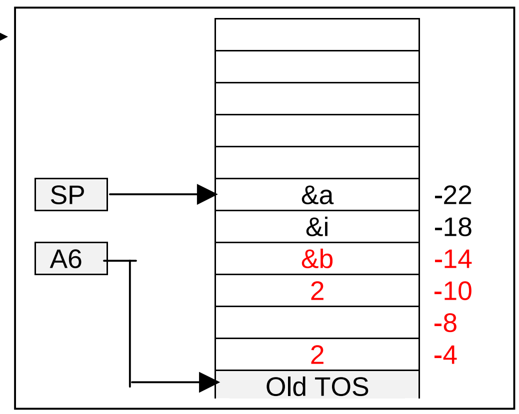
```
int power (long unsigned int *base, unsigned int *exponent,
               long unsigned int *product);
int main()
{
   long unsigned int a, b;
   unsigned int i;
   a = 2;
   i = 2;
   power(&a, &i, &b);
   return 0;
}
int power (long unsigned int *base, unsigned int *exponent,
                  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:

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

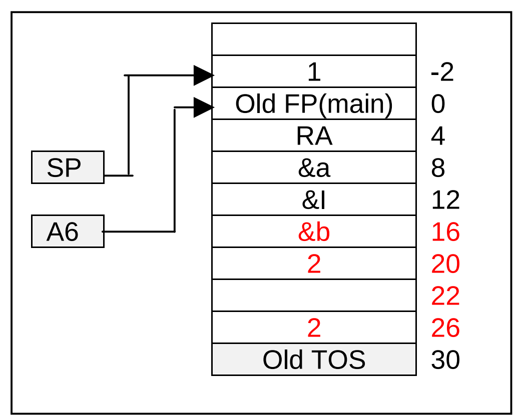



```
      MOVEQ.L     #2,D1
      MOVE.L      D1,-4(A6)
      MOVE        #2,-10(A6)
      PEA.L       -8(A6)
      PEA.L       -10(A6)
      PEA.L       -4(A6)
```

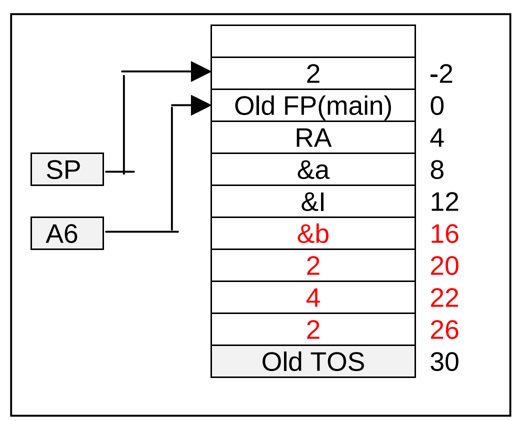



```
      JSR         power
      CLR         D0
      UNLK        A6
      RTS

*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)

D0	01234567	D1	89ABCDEF	D2	0001002D	D3	ABCD7FFF
D4	33449127	D5	AAAAAAAA	D6	ABCD0003	D7	55555555
A0	00007020	A1	00007028	A2	0001000A	A3	00007034
A4	00010020	A5	0000FFFA	A6	00010000	A7	00010080
		XNZVC	b10100			PC	00004000

Main memory (all values are hex unless otherwise noted)

007000	AE	007020	5A	010000	DD	010020	DC
007001	F2	007021	AD	010001	B2	010021	25
007002	32	007022	99	010002	00	010022	15
007003	77	007023	92	010003	15	010023	17
007004	89	007024	79	010004	76	010024	29
007005	90	007025	33	010005	19	010025	39
007006	1A	007026	97	010006	92	010026	49
007007	AE	007027	14	010007	26	010027	2D
007008	EE	007028	79	010008	17	010028	B2
007009	F1	007029	E7	010009	14	010029	62
00700A	F2	00702A	00	01000A	23	01002A	81
00700B	A4	00702B	0A	01000B	E7	01002B	21
00700C	AE	00702C	88	01000C	19	01002C	45
00700D	88	00702D	18	01000D	92	01002D	18
00700E	AA	00702E	82	01000E	19	01002E	31
00700F	E4	00702F	79	01000F	54	01002F	D9
007010	7E	007030	2B	010010	45	010030	AA
007011	8D	007031	17	010011	99	010031	77
007012	9C	007032	46	010012	15	010032	78
007013	C4	007033	9E	010013	43	010033	AE
007014	B2	007034	FC	010014	25	010034	EA
007015	12	007035	FF	010015	76	010035	34
007016	39	007036	77	010016	89	010036	25
007017	90	007037	60	010017	17	010037	17
007018	00	007038	21	010018	81	010038	15
007019	89	007039	42	010019	17	010039	14
00701A	14	00703A	55	01001A	4E	01003A	17
00701B	01	00703B	EA	01001B	72	01003B	F9
00701C	3D	00703C	61	01001C	33	01003C	8A
00701D	77	00703D	81	01001D	23	01003D	0F
00701E	89	00703E	C9	01001E	E1	01003E	F2
00701F	9A	00703F	AA	01001F	CD	01003F	E5

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
```

X	N	Z	V	C
1	0	0	0	0

D0 = 01234579
A1 = 00007029

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

X	N	Z	V	C
1	1	0	0	0

(A1) = 79E7, A1 = 0000702A
(A3) = FCFF, A3 = 00007036

(c) `CLR.L    5(A3,D6.W)`

X	N	Z	V	C
1	0	1	0	0

A3 = 00007034, D6.W = 0003
MEM[703C] = 00
MEM[703D] = 00
MEM[703E] = 00
MEM[703F] = 00

(d) `BEQ #100`

X	N	Z	V	C
1	0	1	0	0

PC = #100

(e) `AND.B    6(A4),$1001C`

6(A4) = MEM[2046] = 49
$1001C = MEM[1001C] = 33
MEM[1001C] = 01

X	N	Z	V	C
1	0	0	0	0

(f) `BCHG.W   #10, D3`

D3 = ABCD7DFF

(g) `ROXR.B   #4, D5`

X	N	Z	V	C
1	0	0	0	0

D5 = AAAAAABA

X	N	Z	V	C
1	1	0	0	1

(h) `ADD.W    $FE(A2), (A3)+`

X	N	Z	V	C
1	0	0	0	1

$FE(A2) = 10008
A3 = 7036
MEM[10008] = 1714
MEM[7034] = FCFF
MEM[7034] = 1413
A3 = 7036