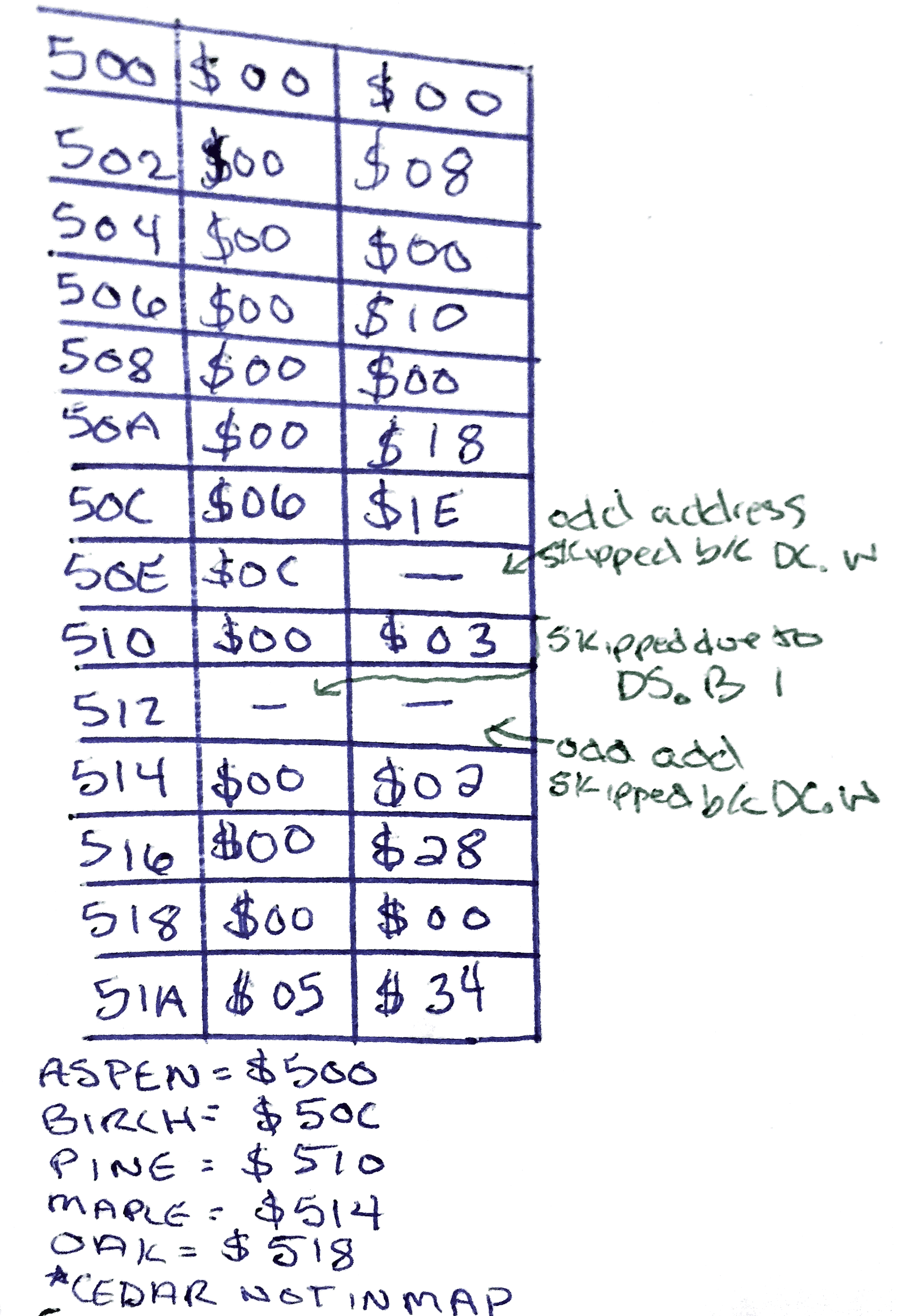
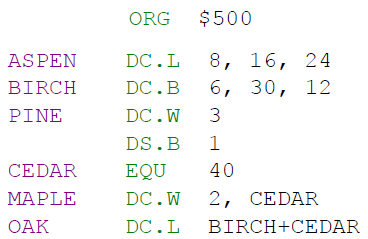
**Assembler directives**  
<LABEL> EQU <VALUE>  
<LABEL> ORG <VALUE>  
<LABEL> DC <VALUE>  
<LABEL> DS <VALUE>  
<LABEL> END <VALUE>

**Memory Map**  
Draw a memory map &specify values of labels  


CEDAR = 4010 = $28 (not actually stored in memory map)

**Errors in code**

**a. ADDQ.L #12, D6**   
Immediate data must be in range 1 to 8.

**b. SWAP A4**   
Address register direct not supported. Argument must be data register direct.

**c. CLR.L A2**   
CLR does not support address register direct.

**d. MOVEA.B #4, A3**   
MOVEA does not support byte operations.

**e. ANDI D4, D5**   
ANDI supports only immediate value as source argument. Also, size was not specified.

**f. LEA.B #4, A3**   
LEA supports only longword operations. g. EOR.W (A2)+, D4 Source operand must be data register direct.

**h. RTS.B**   
RTS does not support byte operations. It is unsized.

**i. ASL.L #9, D3**   
ASL with immediate source operand only permits values of 1 to 8. 9 is out of range.

**j. BRA.B 2741**   
The BRA.B instruction is okay as long as the displacement value (the jump range) can fit into an 8-bit two’s complement value. This is called a ‘short branch’. For this BRA.B instruction to work, it must be present in a location no more than 128 bytes from location 2741. Otherwise, the assembler will report an error.

**Branch Instructions**  
Bcc : branch to label on condition cc true   
BRA : branch to label unconditionally   
DBcc Dn, : test condition cc, decrement, and branch to “label”

Translate Code

If T is 5

Then X←4

Else Y←6

End If

\*One possible solution, assuming T, X, and Y are longwords:

CMP.L #5,T

BNE else

MOVE.L #4,X

BRA onward

else MOVE.L #6,Y

onward ...

**Addressing Modes 68k**  
Immediate addressing   
 Ex) MOVE.L #100, D0  
Absolute (direct) addressing   
 Ex) MOVE.L D3, $1234  
Register direct addressing   
 Ex) MOVE.L A0,D0  
Indirect addressing   
 EX) MOVE.W #5,(A0)   
 MOVE.L #10, (A2)+   
 MOVE.L 12(A4), D2

**Stack**

Push:

MOVE.W D0, -(A7)

MOVEM.L D0-D7/A0-A7, -(A7)

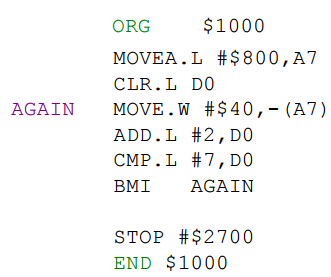
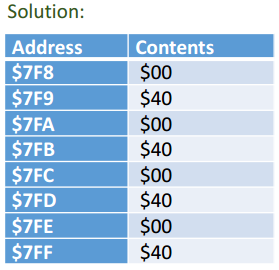
Pop:

MOVE.L (A7)+, D0

MOVEM.L (A7)+, D0-D7/A0-A7

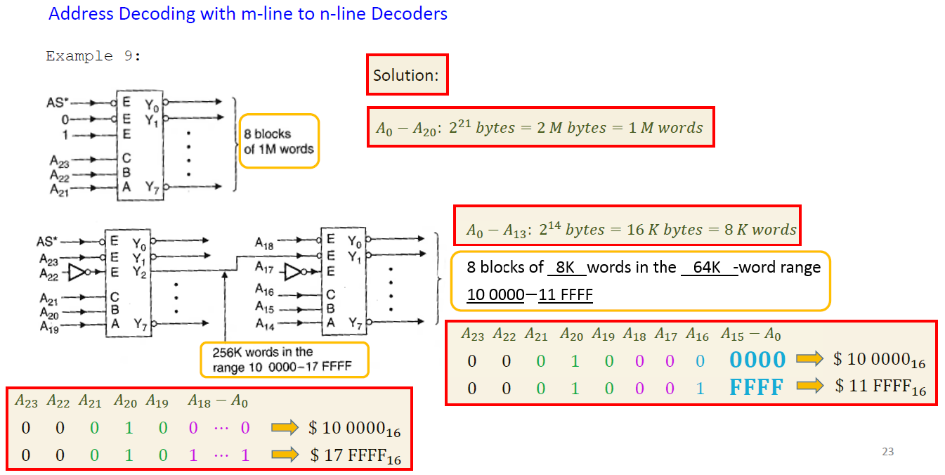
**Loop Example**

Draw the stack memory map for the following program. How many times does each program jump back to label AGAIN

**Subroutine**  
A subroutine is called by **BSR** or **JSR**   
Return from subroutine: **RTS**

**Iteration**   
A repetition of a set of instructions (block) using a loop. Terminates when the loop-continuation condition fails.

**Recursion**   
Repeats the code by calling itself. Terminates when a base case is recognized.

**Cache Memory**  
Cache is a small high-speed memory.

Stores data from some frequently used addresses (of main memory)

**Cache hit:** Data found in cache. Results in data transfer at maximum speed.

**Cache miss**: Data not found in cache. Processor loads data from next layer down in memory hierarchy and copies into cache. This results in extra delay, called miss penalty, which is number of clocks to process a cache miss.

**Hit ratio** = proportion of accesses that hit by the cache.

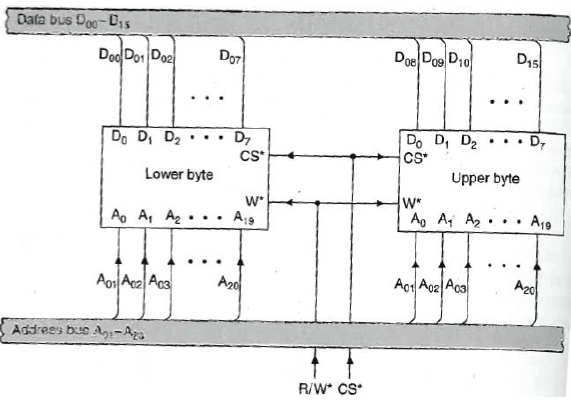
**Miss ratio** = 1 −Hit ratio

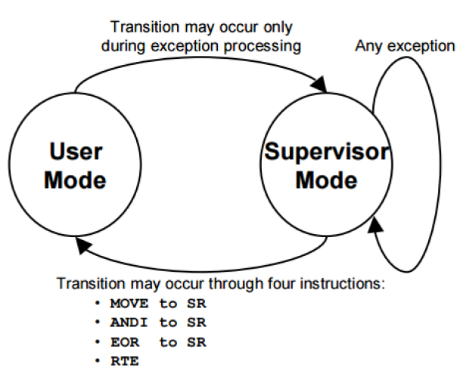
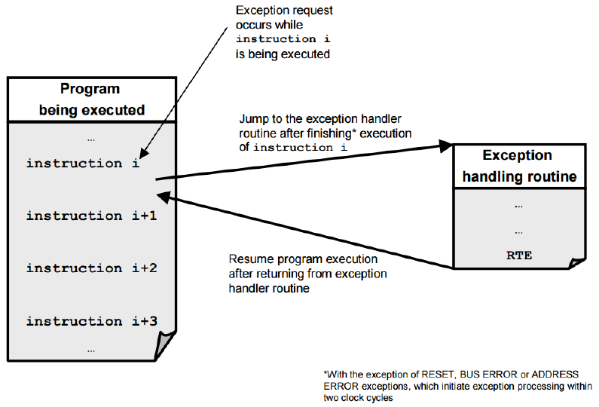
**Average memory access time** = Hit time + Miss rate x Miss penalty

Hit Time << Miss Penalty

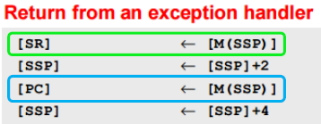
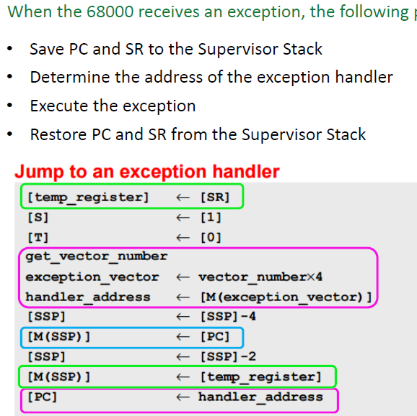
Example 11: A microcomputer has 1𝑀-word of RAM implemented with 1M × 8 chips(1M locations)

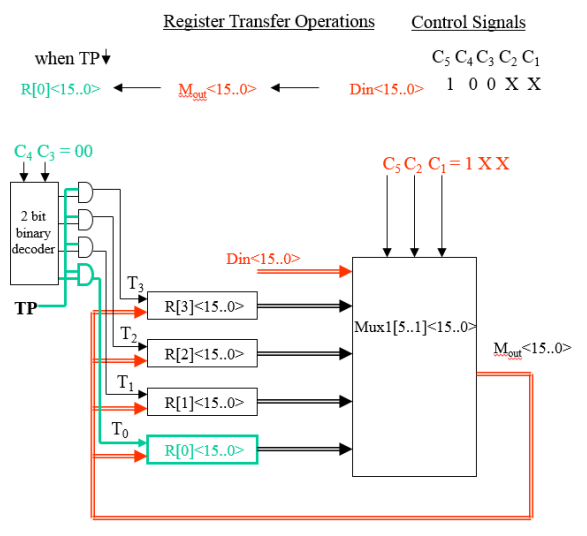
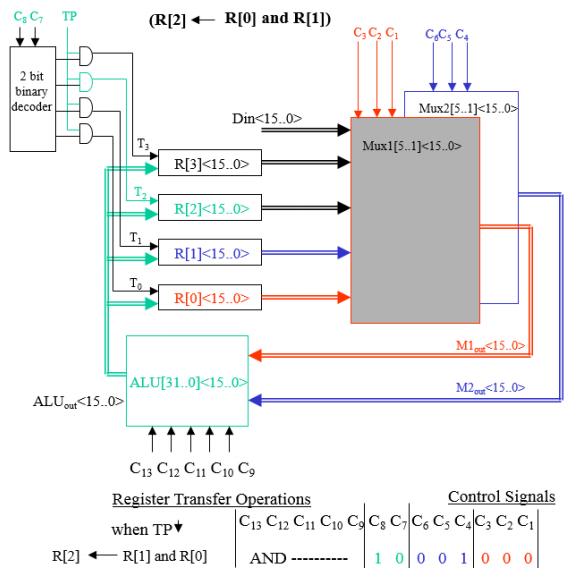
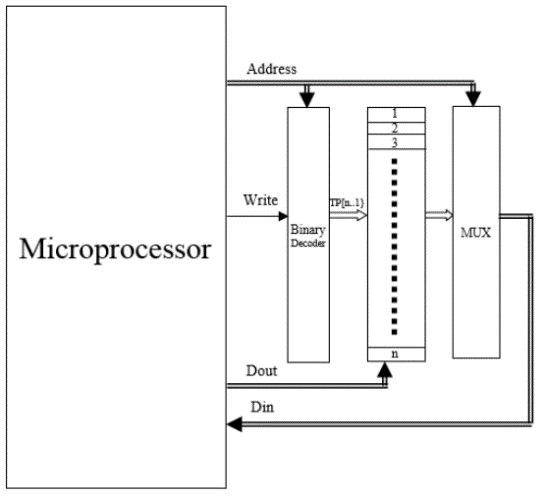
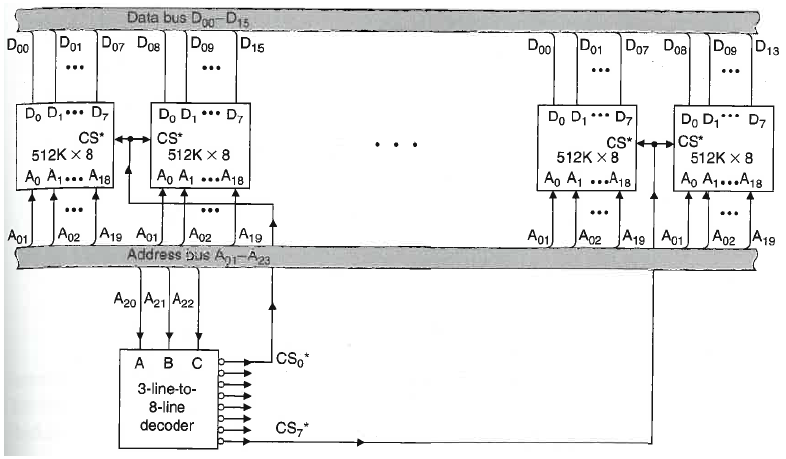
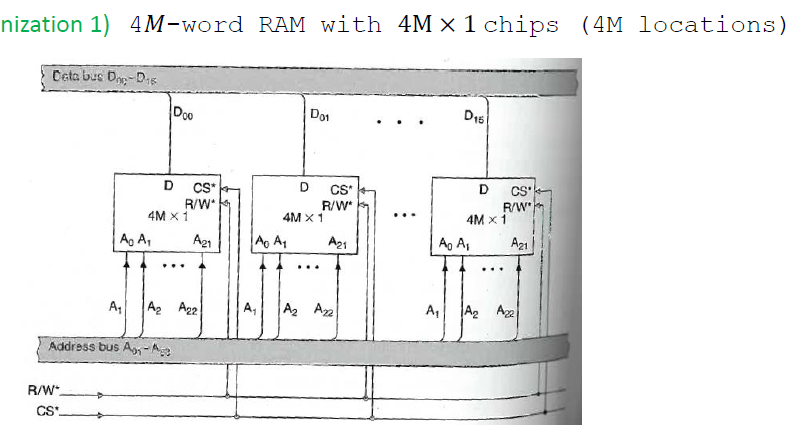
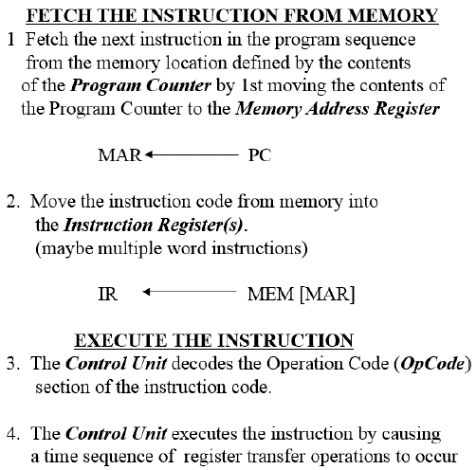
Draw the data bus and address bus connections.





**Exceptions**





**Exception processing**

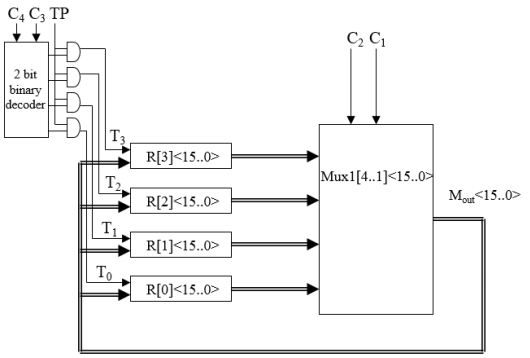
Consider the design of a 4M-word memory. The designer may select either 16 4𝑀×1 chips, or 16 512𝐾×8 chips.  
(Memory organization 1) 4𝑀-word RAM with 4M × 1 chips (4M locations)  
Draw the data bus and address bus connections.Ex contuied below

**Interrupts**

**Arithmetic Unit Register Transfer-Design**

**Adding a RAM memory to the Microprocessor**

**Data Input to MUX Output to Register Transfer**



**Register to MUX Output to Register Transfer**