Complete the following table (2) below for each instruction based on the memory contents in table 1.

Assume that before each instruction that:

Register PC = $1000
Register SP = $0280
Register R5 = $0001
Register R6 = $0002
Register R7 = $0202
Register R8 = $0200
Register R9 = $FFFF

Table 1

Memory address	Memory contents	Label
$0200	$00	
$0201	$01	
$0202	$10	VAR1
$0203	$04	
$0204	$05	
$0205	$04	
$0206	$10	
$0207	$02	
….	…	…

Table 2

Instruction	Addressing Mode	Affected Reg	Contents of Reg	Effective Address
MOV.W R5, R6				
MOV.B R5, R6				
MOV.W #0xFFFF, R6				
MOV.W 2(R7), R6				
MOV.W 2(R7), 2(R8)				
MOV.W VAR1, R6				
MOV.W &VAR1, R6				
MOV.B @R7, R6				
MOV.B @R7+, R6				

Problem 2

Write a subroutine that moves 16 bytes of data from a source range in the memory map to a destination range. Assume that R5 has the first address of source range and that R5 + 15 is the final address in the source range. R6 – similarly for the destination - has the initial destination address and R6 + 15 is the final destination address.

Problem 3

Write a subroutine that takes a 16-bit unsigned number passed in register R5, squares the number and returns the result in R5. The subroutine must check that the input value is small enough that the square is less than 256 and if not A should be cleared as an indication of overflow. You can use the multiplier to perform the square.

Problem 4

Write a short program that will light up one of the eight LEDs on Port 2 (assume a board is configured with all port 2 pins tied to LEDS) and have the light race from right to left and then left to right – seemingly bouncing back and forth. The program should run forever.