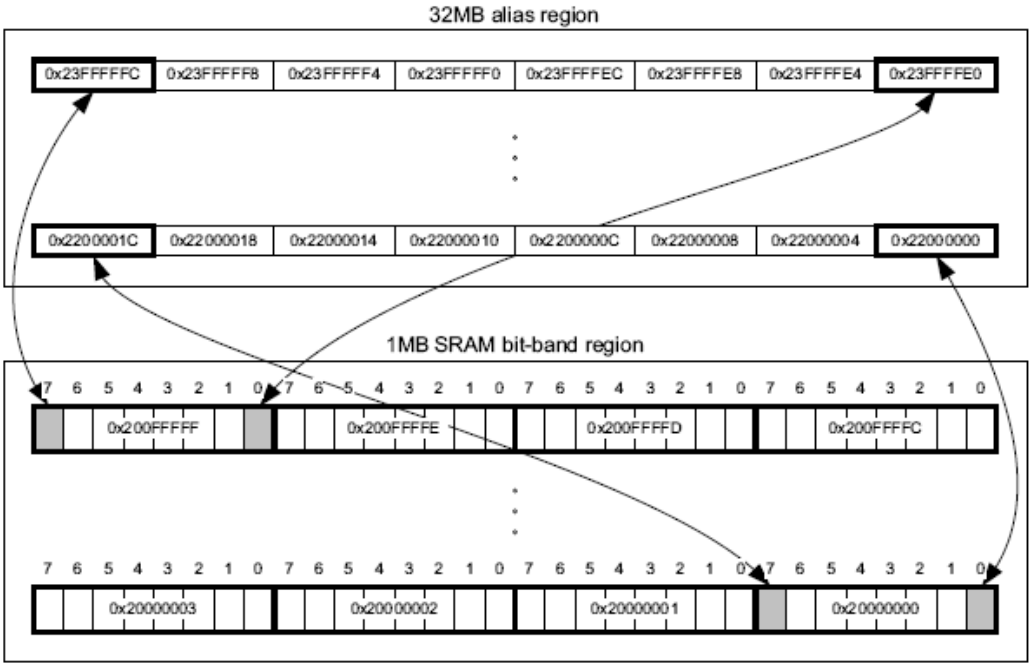
CE2107 Microprocessor System Design and Development

Tutorial 2 (with Solutions)

Signals and Interfacing

1. a) Explain how the bit banding is performed and show why 32MB of the address space is needed for the mapping of 1MB of bits.

Solution:



1MB = 8 Mb.

Each bit will be accessed by a 32-bit word, i.e. 4 Bytes

8Mb = 8M x 4Bytes = 32 Mbytes

b) Show the codes that can be used to set b[3] (i.e. 4th bit) of address location 0x20000001.

Solution:

b[3] of 0x20000001 is the 4th bit, and hence 12 bits from b[0] of 0x20000000.

The address should be offset from 0x22000000 by 11 words, or 44 bytes (i.e. 2Ch), which is hence equal to 0x2200002C.

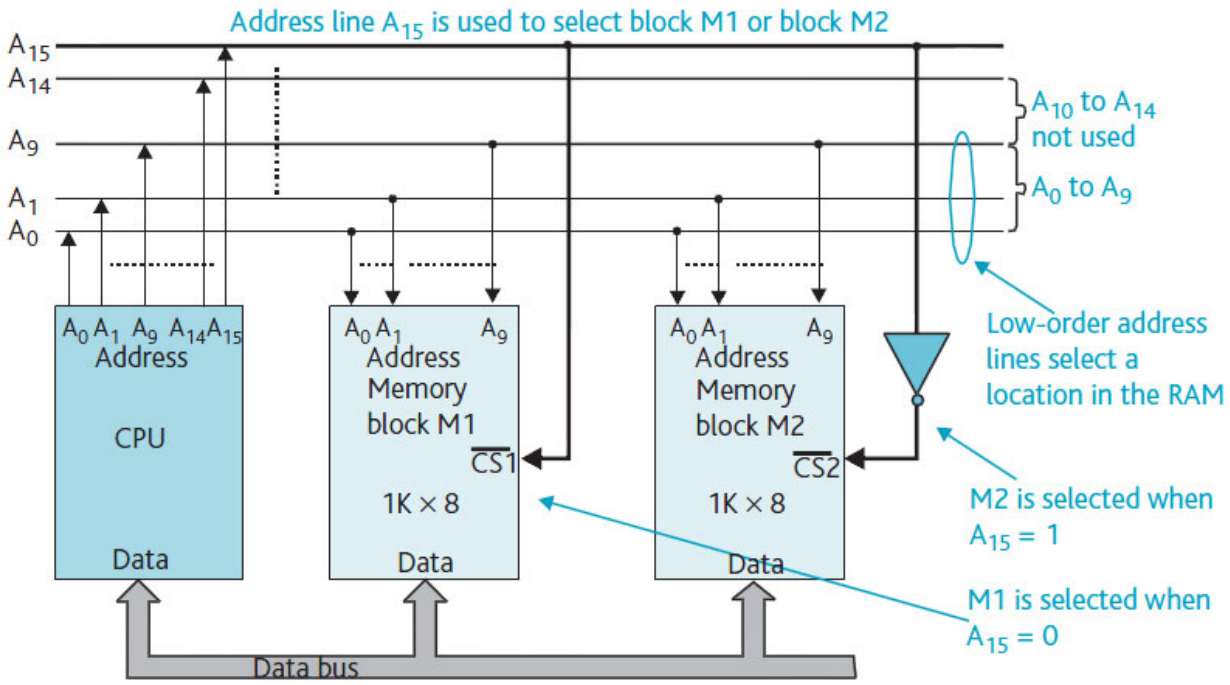
The codes can be as follows:

```
LDR   R0, =0x2200002C
MOV   R1, #0x1
STR   R1, [R0]
```

2. A microprocessor with 16-bit address bus needs to be interfaced with two 1Kx8 SRAMs.
   a) With the help of a diagram, show how partial address decoding can used for interfacing the microprocessor with the two SRAMs.
   b) Discuss the advantages and disadvantages of your address decoding design.

Solution:

(a)

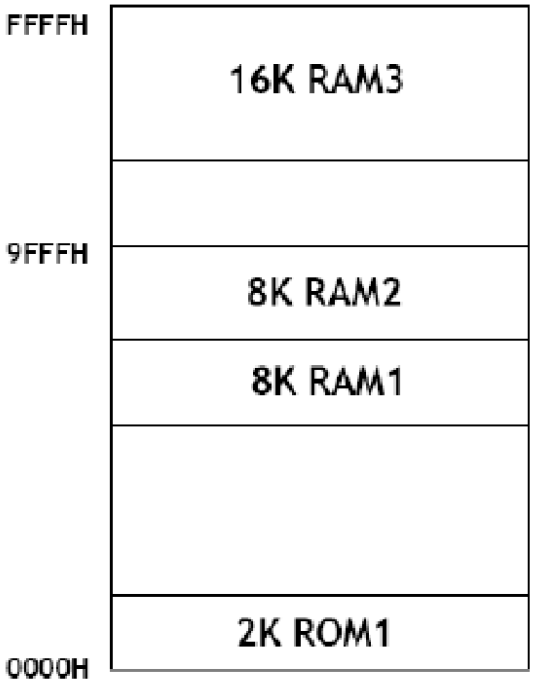


(b)

Advantage: Small address decoder logic.

Disadvantage: Prevents full use of the microprocessor's address space.

3. A microprocessor system has the memory map as shown below.
   a. Provide the starting and ending addresses for each memory device in hexadecimal format.
   b. Give the address decoding logic.
   c. Draw the address decoding circuit and provide hardware connections for the external memory interfacing.
   (Hint: Consider using a 3-to-8 decoder together with discrete logic gates)



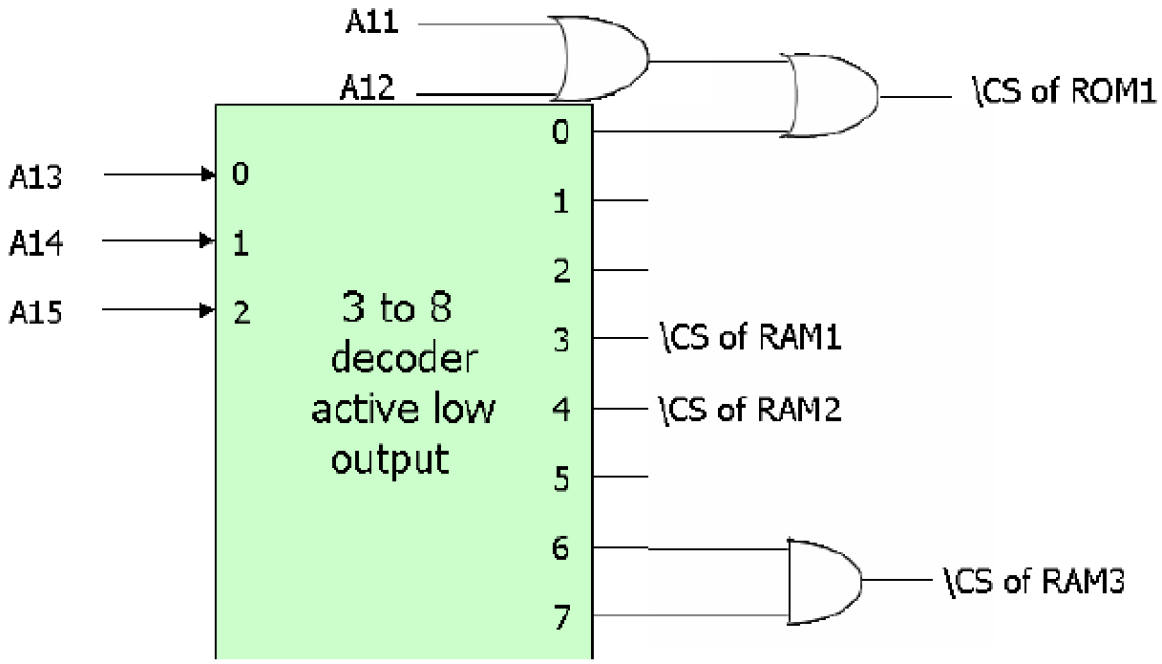
Solution:

2K ROM1	0000H	07FFH
8K RAM1	6000H	7FFFH
8K RAM1	8000H	9FFFH
16K RAM3	C000H	FFFFH

The following is the address decoding table (x indicates don't care):

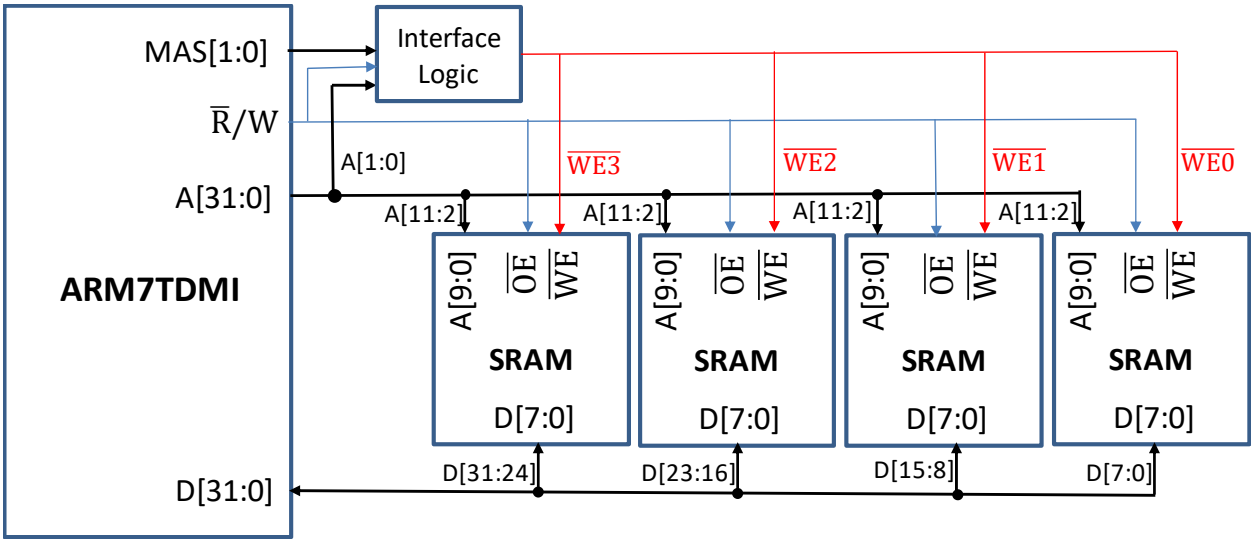
A15	A14	A13	A12	A11	Chip Select
0	0	0	0	0	ROM1
0	1	1	X	X	RAM1
1	0	0	X	X	RAM2
1	1	X	X	X	RAM3

The address decoding circuit is as below:



4. This question is based on the memory interfacing diagram that is covered in the lecture notes. The microprocessor is interfaced to a 4Kx8 memory that is constructed using four 1Kx8 SRAMs.

   Design the interface logic that will allow you to **write** to the correct SRAM chips. MAS[1:0] can be used to determine if a byte/half-word/word is to be written to the SRAM based on the table below.



Access type	MAS[1:0]	A[1:0]	Little-endian
Word	10	XX	**D[31:0]**
Halfword	01	0X	**D[15:0]**
	01	1X	**D[31:16]**
Byte	00	00	**D[7:0]**
	00	01	**D[15:8]**
	00	10	**D[23:16]**
	00	11	**D[31:24]**

Solution:

Create a truth table for the interface logic.

	nRW	MAS[1]	MAS[0]	A[1]	A[0]	$\overline{WE3}$	$\overline{WE2}$	$\overline{WE1}$	$\overline{WE0}$
Read operation	0	X	X	X	X	1	1	1	1
None (MAS = 11)	1	1	1	X	X	1	1	1	1
Write word	1	1	0	X	X	0	0	0	0
Write half-word	1	0	1	0	X	1	1	0	0
Write half-word	1	0	1	1	X	0	0	1	1
Write byte	1	0	0	0	0	1	1	1	0
Write byte	1	0	0	0	1	1	1	0	1
Write byte	1	0	0	1	0	1	0	1	1
Write byte	1	0	0	1	1	0	1	1	1

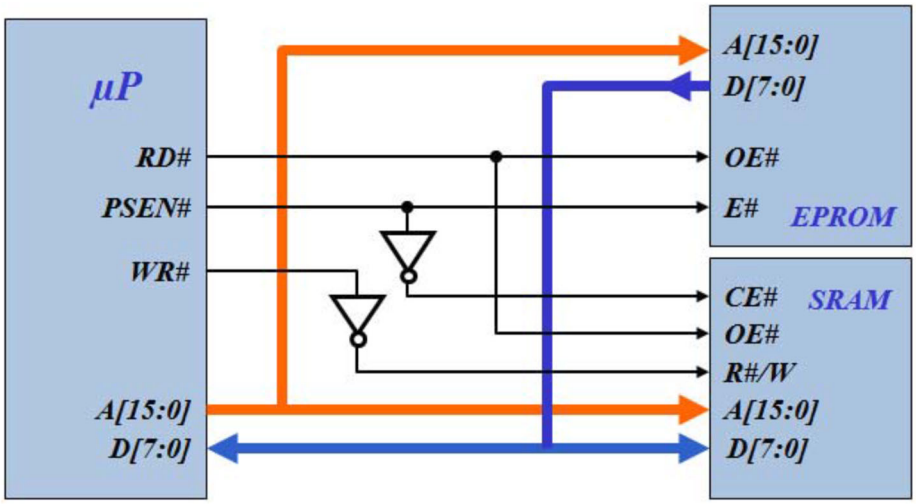
Using K-Map, find the Boolean expression of the write enable signals.

$$\overline{WE3} = \overline{MAS[1]}.\overline{A[1]} + MAS[1].MAS[0] + \overline{MAS[1]}.\overline{MAS[0]}.\overline{A[0]} + \overline{nRW}$$
$$\overline{WE2} = \overline{MAS[1]}.\overline{A[1]} + MAS[1].MAS[0] + \overline{MAS[1]}.\overline{MAS[0]}.A[0] + \overline{nRW}$$
$$\overline{WE1} = \overline{MAS[1]}.A[1] + MAS[1].MAS[0] + \overline{MAS[1]}.\overline{MAS[0]}.\overline{A[0]} + \overline{nRW}$$
$$\overline{WE0} = \overline{MAS[1]}.A[1] + MAS[1].MAS[0] + \overline{MAS[1]}.\overline{MAS[0]}.A[0] + \overline{nRW}$$

Optional

5. The following diagram shows the memory interface of a SRAM and an EPROM to a microprocessor that uses a memory selection signal PSEN# (Program Strobe Enable, active low) when accessing external memory. Specifically, PSEB# is asserted when the microprocessor is accessing the program code (rather than data).

   a. Describe the operation of the memory system with the microprocessor.
   b. A SRAM typically consumes higher power when its CE# is asserted (e.g. 100µA vs 2µA.) What is the disadvantage of this memory interface design and how it can be improved?



Solution:

A microprocessor system needs non-volatile memory like EPROM to permanently store the program code, and uses the SRAM to store data during execution.

1. The design of the system hence contains one EPROM, which is enabled when the PSEN# is asserted during fetching of the program code. By having the NOT gate between the PSEN# and CE# of SRAM, the SRAM is automatically disabled when the microprocessor is accessing the EPROM for program.

   The SRAM is hence enabled whenever the EPROM is not selected. However, the SRAM will only be access when the microprocessor performs a write, or a non-EPROM read, which by default will enable the OE# or R# of the SRAM in the design.

2. As the SRAM is always enabled during non-EPROM operation, power is continuously consumed even when it is not accessed. To reduce the power consumption (from 100uA to 2uA), the SRAM should be disabled except during non-EPROM read/write accessed (indicated by the PSEN# signal). A more elaborate decoding scheme is hence needed to only enable the SRAM when it is accessed, such as based on the logic

CE# = (PSEN#)# + (RD#.WR#)