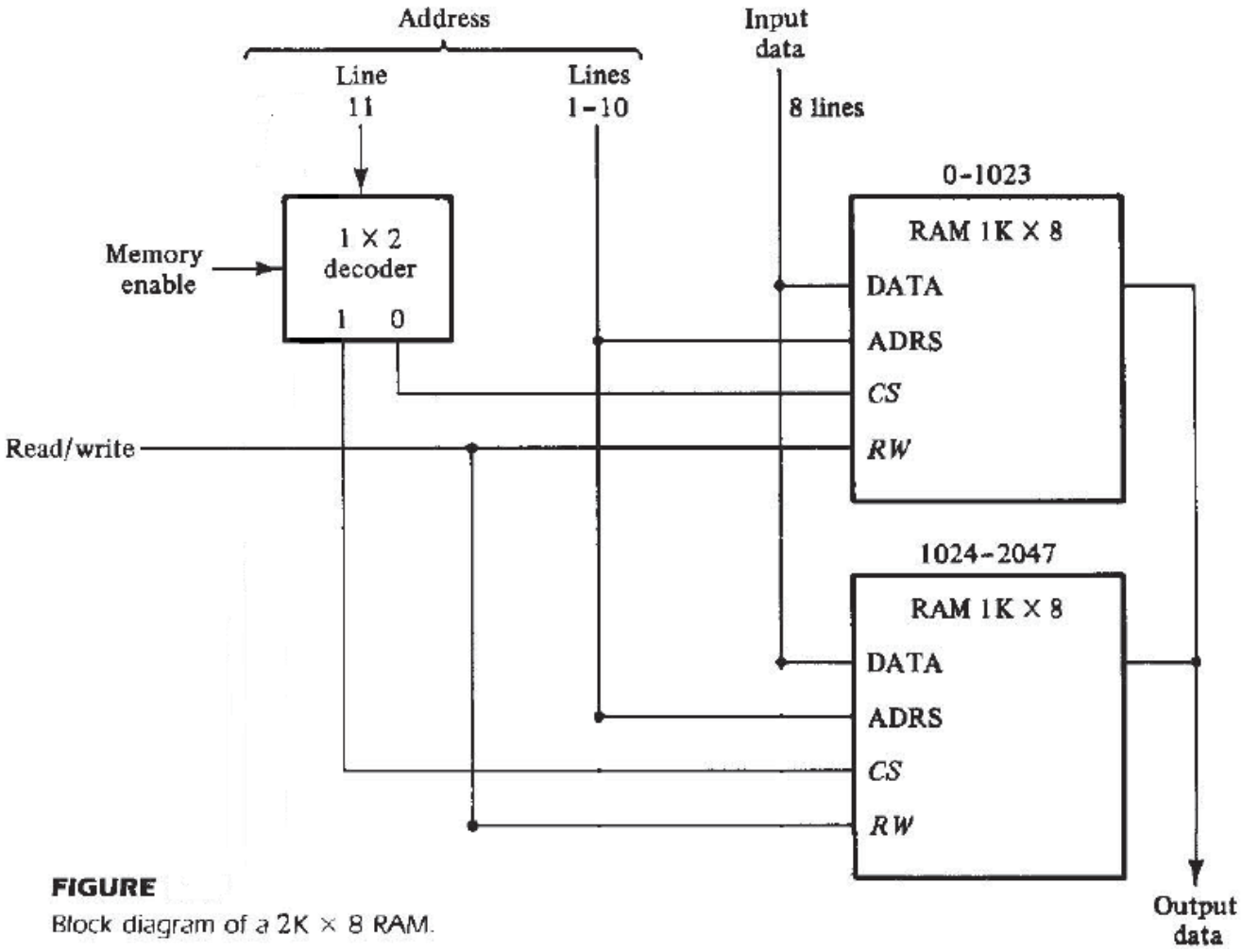
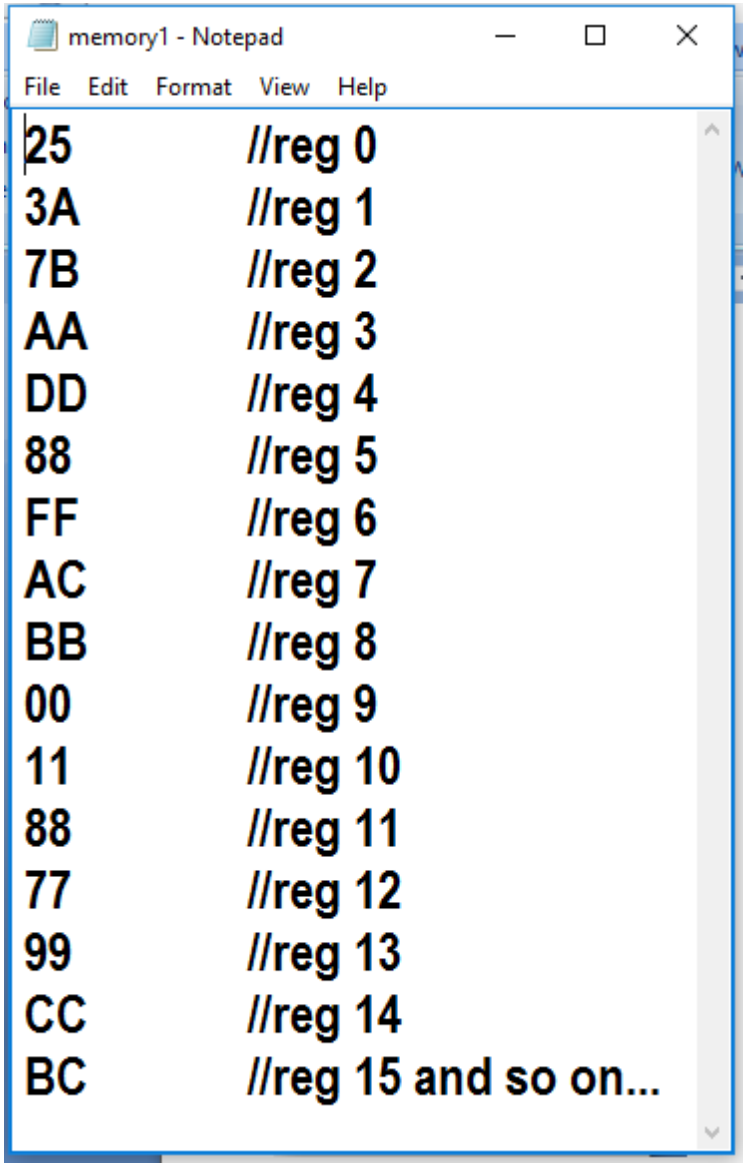


FIGURE
Block diagram of a 2K × 8 RAM.

```
module Memory_2Kx8 (addr, CS, RW, idata, odata);

     input CS, RW;
     input [10:0] addr;
     input [7:0] idata;
     output [7:0] odata;

     wire [7:0] odata1, odata2;

     wire D0, D1;

     assign D0 = CS && ~addr[10]; //Output D0 of the 2x1 decoder
     assign D1 = CS && addr[10];  //Output D1 of the 2x1 decoder

     RAM1 rc1 (addr[9:0], D0, RW, idata, odata1);
     RAM2 rc2 (addr[9:0], D1, RW, idata, odata2);

     assign odata = (CS && RW)?(odata1 | odata2):8'bz;

endmodule
```

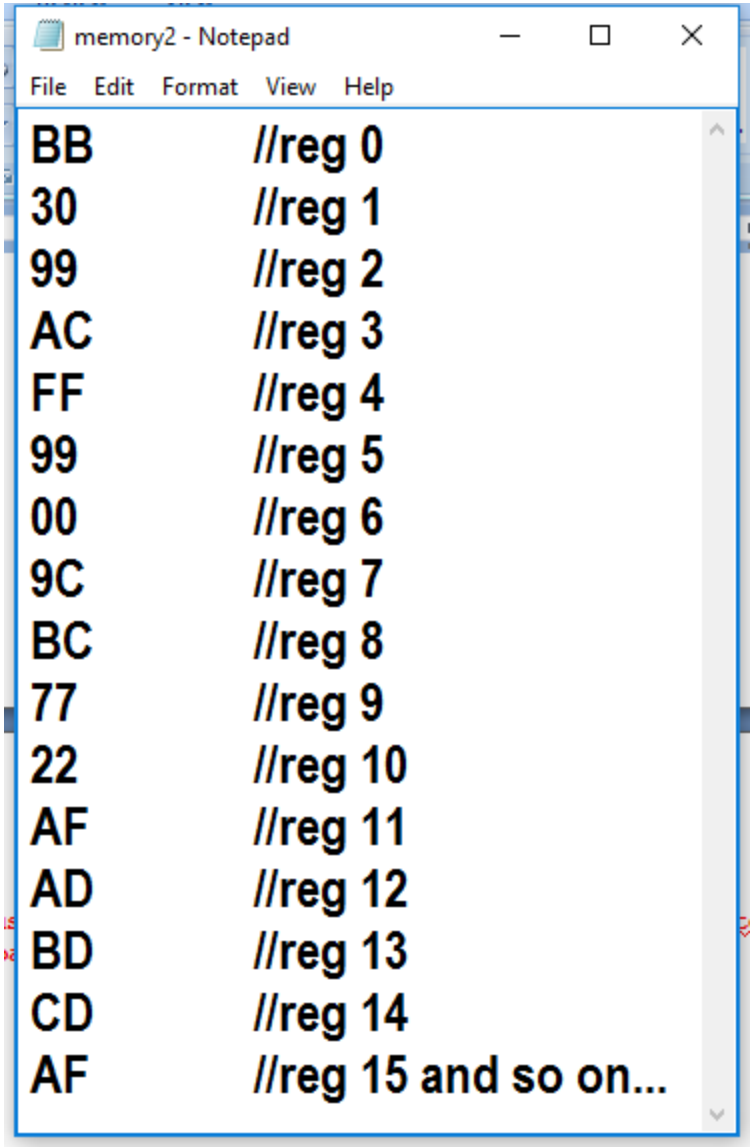
```
module RAM1 (addr, CS, RW, idata, odata);

     input CS, RW;
     input [9:0] addr;
     input [7:0] idata;
     output [7:0] odata;
     reg [7:0] d_out;
     reg [7:0] Mem1 [0:1023];

     assign odata = (CS && RW)?d_out:8'b0;

     always @(addr or idata or CS or RW)
               if (CS && !RW)
                    Mem1 [addr] = idata;
          always @(addr or CS or RW)
               if (CS && RW)
                    d_out = Mem1 [addr];

        initial
               $readmemh ("memory1.dat", Mem1);

endmodule
```

//memory1.dat. The data file must be in the same directory as design and testbench. Otherwise you will need to specify the complete path.



```
module RAM2 (addr, CS, RW, idata, odata);

     input CS, RW;
     input [9:0] addr;
     input [7:0] idata;
     output [7:0] odata;
     reg [7:0] d_out;
     reg [7:0] Mem2 [0:1023];

     assign odata = (CS && RW)?d_out:8'b0;

     always @(addr or idata or CS or RW)
               if (CS && !RW)
                    Mem2 [addr] = idata;
          always @(addr or CS or RW)
               if (CS && RW)
                    d_out = Mem2 [addr];

        initial
               $readmemh ("memory2.dat", Mem2);

endmodule
```

//memory2.dat. The data file must be in the same directory as design and testbench. Otherwise will you need to specify the complete path.



```
module tst_Memory_2Kx8 ();

     reg [10:0] address;
     reg CE, RW;
     reg [7:0] data_in;
     wire [7:0] data_out;

     Memory_2Kx8 m1 (
        .addr(address),
        .CS (CE),
        .RW(RW),
        .idata(data_in),
        .odata(data_out));

     initial begin
              $display("Reading from Memory (already initialized/populated using $readmemh)...");

               CE = 0;
               RW = 0;
               #5 address = 15;
               #5 RW = 1;
               #5 CE = 1;
               #10 address = 1;
               #10 address = 2;
               #10 address = 3;
               #10 address = 1024;
               #10 address = 6;
               #10 address = 1025;
               #10 address = 1030;
               #10 address = 8;
               #10 address = 9;
               #10 address = 10;
               #10 address = 11;
               #10 address = 0;
```

```
            #10 address = 14;
            #10 address = 13;
            #10 address = 12;

            #500 $finish;
    end

endmodule
```