```
1   // Eugene Ngo
2   // 4/20/2023
3   // CSE 469
4   // Lab 2, Task 1 and 2
5
6   /* dmem is a more traditional, albeit very uninteresting, random access 64 word x 32 bit
    per word memory.
7   ** This module is also written in RTL, and likely strongly resembles your own register file
    except for a
8   ** few minor differences. The first is that there is only a single read port, compared to
    the register
9   ** file's two read ports. The other difference is that the dmem is also byte aligned, and
    therefore
10  ** discards the bottom two bits of the address when doing a read or write.
11  */
12
13  // clk - system clock, same as the processor
14  // wr_en - write enable, allows the wr_data to overwrite the 32 bit word stored in
    memory[addr]
15  // addr - the location to which you intend to read or write from
16  // wr_data - the 32 bit data word which you intend to write into memory
17  // rd_data - the data currently stored at memory[addr]
18  module dmem (
19      input  logic        clk, wr_en,
20      input  logic [31:0] addr,
21      input  logic [31:0] wr_data,
22      output logic [31:0] rd_data
23  );
24
25      logic [31:0] memory [63:0];
26
27      // asyncrhnous read
28      assign rd_data = memory[addr[31:2]]; // word aligned, drop bottom 2 bits
29
30      // syncrhonous gated write
31      always_ff @(posedge clk) begin
32          if (wr_en) memory[addr[31:2]] <= wr_data; // word aligned, drop bottom 2 bits
33      end
34
35  endmodule
```