```verilog

/*
   Modular 2^depth x width register file with 2 read ports and 1 write port.
   Read port 1 accesses r_data1 according to address r_addr1.
   Read port 2 accesses r_data2 according to address r_addr2.
   The single write port writes w_data to address w_addr.
*/

/*
   DESIGN DECISION:
   An internal register array was chosen rather than instatiating 16 register.v entities,
   since register arrays are implemented using on-board FPGA memory, rather than costly FFs.
*/

module reg_file #(
   // for modularity
   parameter depth = 4,
   parameter width = 32
   )(
   input clk, wr_en,
   input [depth-1:0] r_addr1, r_addr2, w_addr,
   input [width-1:0] w_data,
   output [width-1:0] r_data1, r_data2
   );

   // Internal memory array consisting of 2^depth, width-lengthed registers.
   // This definition enforces a right-to-left increasing bit significance, and an up-to-down addressing scheme.
   reg [width-1:0] reg_array [0:2**depth-1];

   // Default all contents to 0.
   genvar i;
   generate
   for (i=0; i<2**depth; i = i+1) begin : init_loop
      initial reg_array[i] = 32'b0;
   end
   endgenerate

   // r_data1/2 are immediately available as r_addr1/2 is presented to the RF
   // Two read addresses are supported by the register file should we choose to implement a 3-bus design in later phases.
   assign r_data1 = reg_array[r_addr1];
   assign r_data2 = reg_array[r_addr2];

   // Data writes are synchronous to clk with wr_en high.
   always @(posedge clk) if (wr_en) reg_array[w_addr] = w_data;

endmodule

```