Vector2

← vector1 (/wiki/vector1)

vectorgates → (/wiki/vectorgates)

A 32-bit vector can be viewed as containing 4 bytes (bits [31:24], [23:16], etc.). Build a circuit that will reverse the *byte* ordering of the 4-byte word.

```
aaaaaaaabbbbbbbbccccccccdddddddd => ddddddddccccccccbbbbbbbbaaaaaaaa
```

This operation is often used when the endianness (https://en.wikipedia.org/wiki/Endianness) of a piece of data needs to be swapped, for example between little-endian x86 systems and the big-endian formats used in many Internet protocols.

Module Declaration

```
module top_module(
    input [31:0] in,
    output [31:0] out );
```

Hint...

Write your solution here

```
1  module top_module(
2      input [31:0] in,
3      output [31:0] out );//
4
5      // assign out[31:24] = ...;
6
7  endmodule
8
```

Submit | Submit (new window)

Upload a source file...

← vector1 (/wiki/vector1)

vectorgates → (/wiki/vectorgates)

Problem Set Contents

- Getting Started
- **Verilog Language**
  - Basics
  - **Vectors**
    - ✓ Vectors (/wiki/vector0)
    - ✓ Vectors in more detail (/wiki/vector1)
    - ○ **Vector part select (/wiki/vector2)**
    - ○ Bitwise operators (/wiki/vectorgates)
    - ○ Four-input gates (/wiki/gates4)
    - ○ Vector concatenation operator (/wiki/vector3)
    - ○ Vector reversal 1 (/wiki/vectorr)
    - ○ Replication operator (/wiki/vector4)
    - ○ More replication (/wiki/vector5)
  - Modules: Hierarchy
  - Procedures
  - More Verilog Features
- Circuits
- Verification: Reading Simulations
- Verification: Writing Testbenches