define文

- Verilogも他の言語と同様、なるべくコード中に直接数を書かないようにするのがお勧めです。define文はこのために使います。
- 記述が読みやすく、変更が容易になります。
- C言語同様、基本的には文頭、module文の前に置きます。
- C言語と違って＃ではなく、バックシングルコーテーションを用いる
  - この記号はめったに使わないし、見にくいので困ったもの
  - 引用の際もバックシングルコーテーションが必要！
- 例

```
`define DATA_W 16
`define SEL_W 3
`define ALU_THA `SEL_W'b000
`define ALU_THB `SEL_W'b001
`define ALU_AND `SEL_W'b010
`define ALU_OR `SEL_W'b011
```

前に定義したSELを引用してさらに定義している

define文の利用

- シングルバックコーテーションを使って引用
- 例

```
input [`DATA_W-1:0] a,b ;
assign y = s==`ALU_THA? a;
           s==`ALU_THB? b;
           s==`ALU_AND? a&b: a+b ;
```

include文で、よく使う定義をひとまとめにしておいて、複数のファイルで共有

- 例　def.hというファイルの中に

```
`define ENABLE 1
`define DISABLE 0
```

と書いておいて、複数のファイルの先頭で

```
`include “def.h”
```

として定義を共有すれば、同じように使える

複数の定義ファイルで同じ文字列に違った定義をしないように注意！

エラーにならず、後の定義が先の定義を上書きする場合もある

parameter文

- define文とは違ってモジュール文の先頭ではなく、文中に式の形で定義します

parameter STEP=10;

- defineと違ってバックシングルクォーテーションが不要
- 引用でもバックシングルクォーテーションが不要
- モジュールを呼び出す時に上位階層のモジュールからパラメータ値の設定ができる→パラメタライズ記述
  - 格好いいがここではやらない
- こっちの方が便利じゃん、なんでdefine文使うの？

  `include文による共有が格好良くできないから （やればできるんだけど、、）
  parameter文はパラメタライズのために使うのが望ましいから
  なんとなく、、、