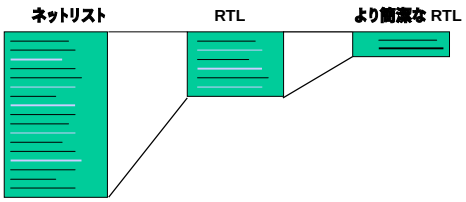
こんな話も聴けたりします！
これからのデザインの必須アイテム＝新しいRTL設計検証言語ってどうよ？

SystemVerilog 3.0

より良いデザイン記述は検証しやすい記述をも意味する



より簡潔に書かれた合成可能なRTL は
一般に 1/3 - 1/5 の記述量の削減となる

SystemVerilog 記述例(1)・構造体

```
typedef struct {
      byte R,G,B;
      } RGB;

const RGB BLUE = {0,0,255};
RGB Frame[639:0][479:0];
Frame[x][y] = BLUE;
```

定数リテラル
二次元アレイ
コピー

SystemVerilog 記述例(3) : always_*

- 設計意図を明確に区別
  - always_comb
  - always_latch
  - always_ff

```
always_comb begin
  tmp1 = a & b;
  tmp2 = c & d;
  y = tmp1 ! tmp2;
end
```
正しい使用法

```
always_comb
  if (en) q <= d;
```
順序回路が推測されるのでエラー

```
always_latch
  if (en) q <= d;
  else q <= q;
```
フィードバックが推測されるのでエラー

正しい使用法

センシティビティ・リストが不明瞭なのでエラー

SystemVerilog 記述例(2)・interface

Verilog95

SystemVerilog

1/3 以下へ記述量を削減
記述ミスによるインタフェイス誤りの低減