

これが高位合成Polyphonyだ！

Pythonで回路を記述

```
from polyphony import module, is_worker_running
from polyphony import Port
from polyphony import bit, bit32
```

```
@module
class Lchika:
    def __init__(self):
        self.clock = Port(bool, 'in')
        self.onBoardLED1 = Port(bool, 'out')
```

```
    self.append_worker(self.main)
```

```
def main(self):
    counter.bit32 = 0
    while is_worker_running():
```

Lchika.py

Polyphonyで
コンパイル
(高位合成)

Lchika_m.v

物理制約ファイル

```
IO_LOC "clk" 52;
IO_PORT "clk" IO_TYPE=LVC MOS18 PULL_MODE=UP;

IO_LOC "led1" 10;
IO_PORT "led1" IO_TYPE=LVC MOS18 PULL_MODE=UP DRIVE=8;
```

top.cst

SystemVerilog記述

```
module top(
    input wire clk,
    output wire led1
);
```

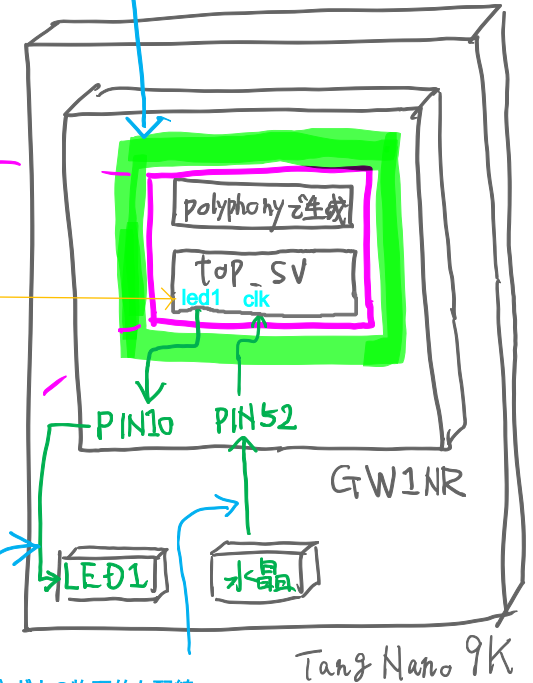
```
    Lchika_m inst(
        clock(clk),
        .onBoardLED1(led1)
    );
```

```
endmodule
```

top.SV

Polyphonyで作った
Lchika_m.vを実体化

SystemVerilogで書かれたモジュールのI/Oと物理的なFPGAチップの端子をどのようにつなぐのかを物理制約ファイル(top.cst)で指定する



ボード上の物理的な配線
(Schematicsで確認)