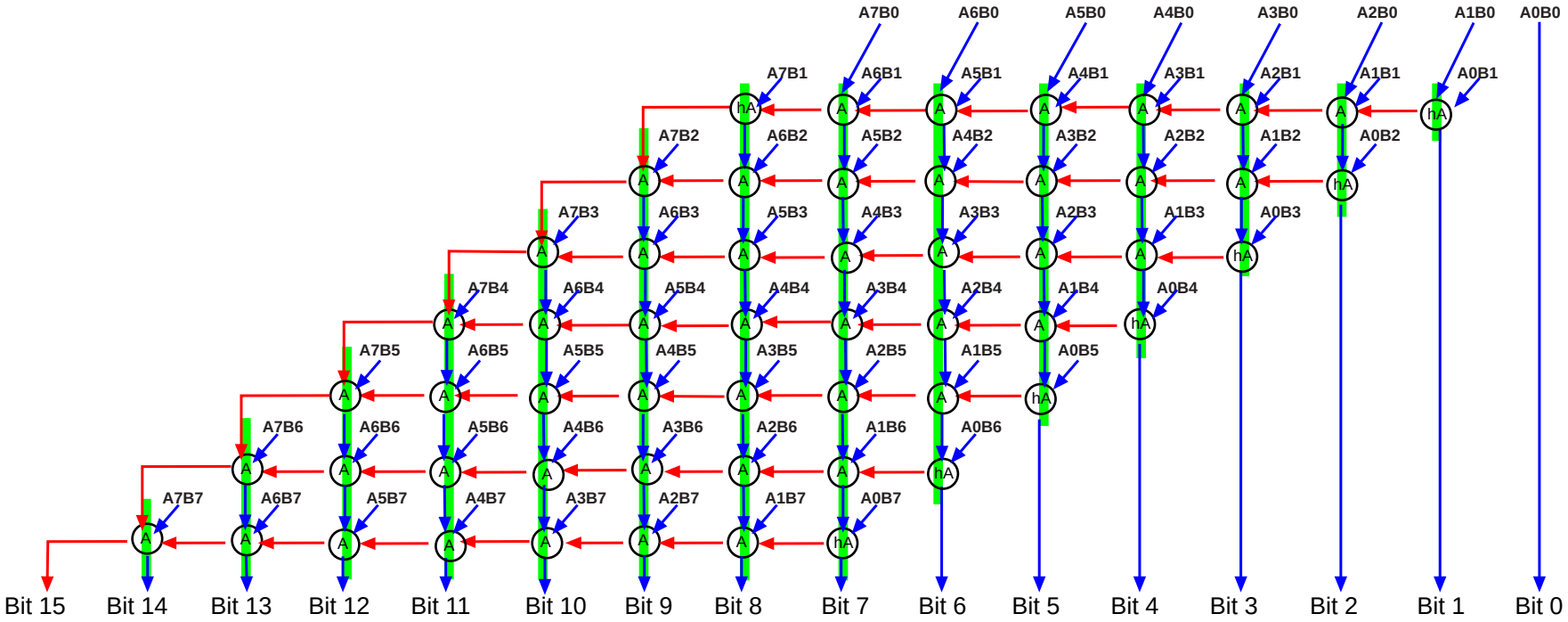
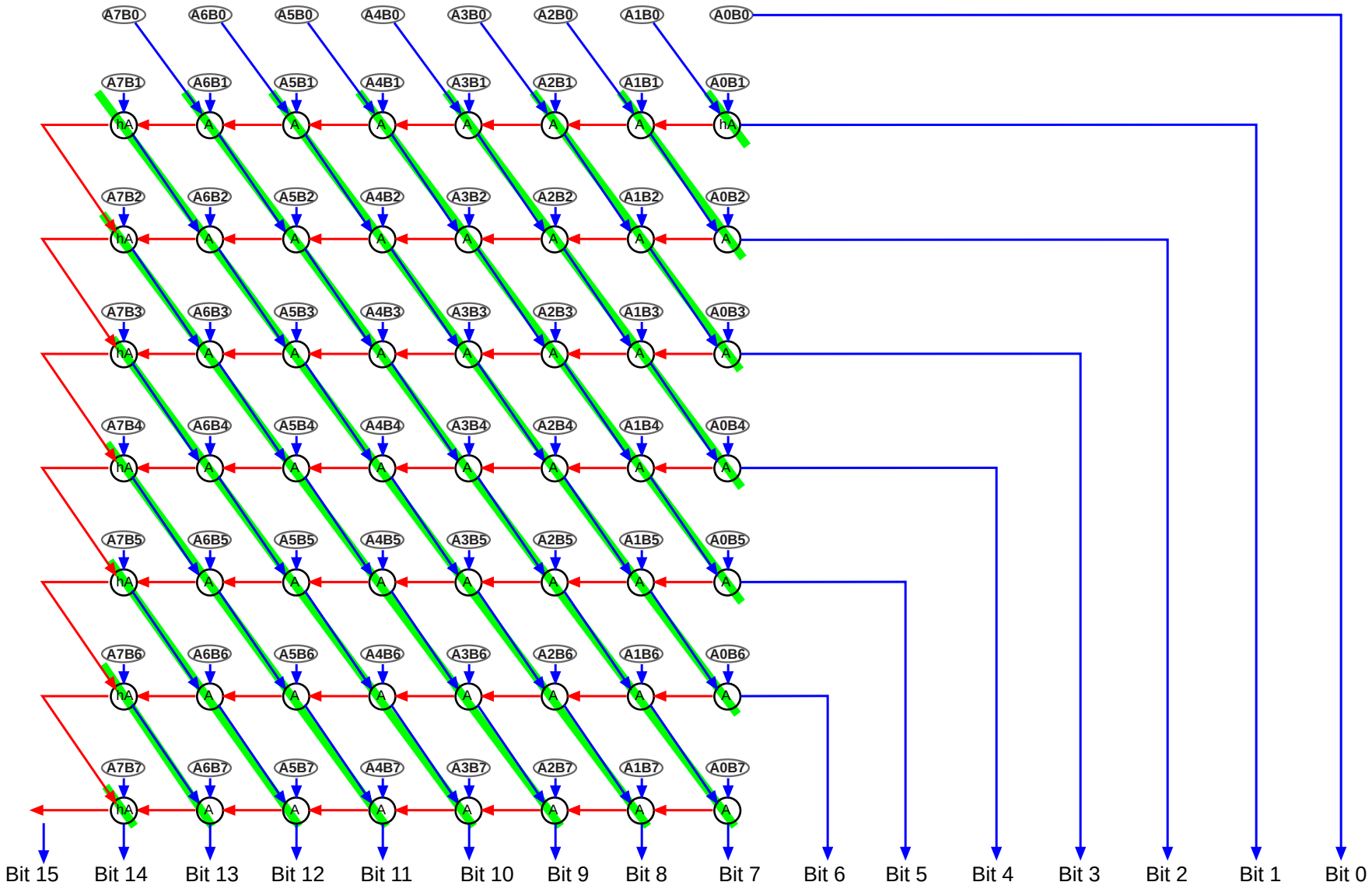
Sandbox 7

2D Pipelined Array Multiplier

8X8 multiply array

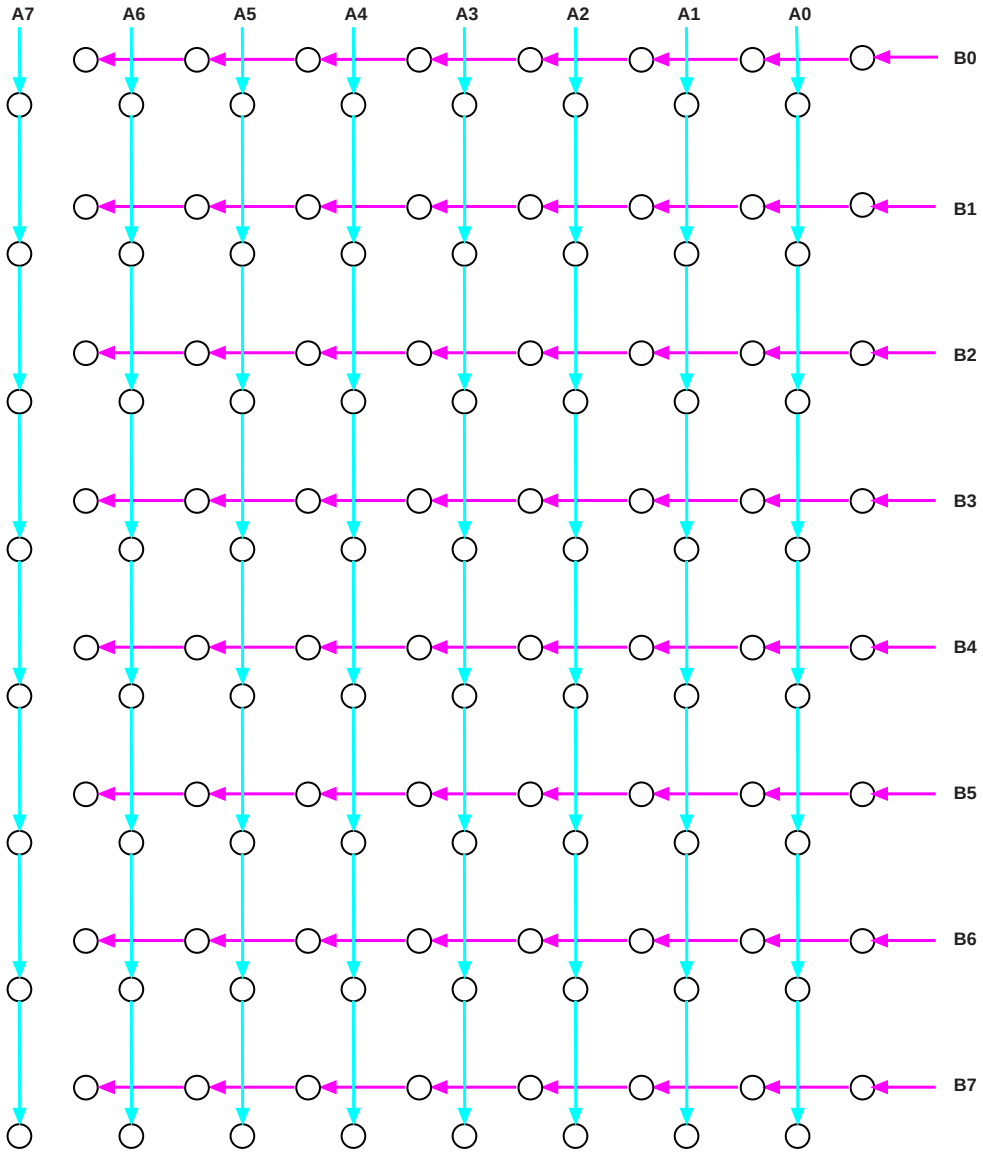


8X8 multiply array squared up



8X8 multiply array 2D pipeline structure

A

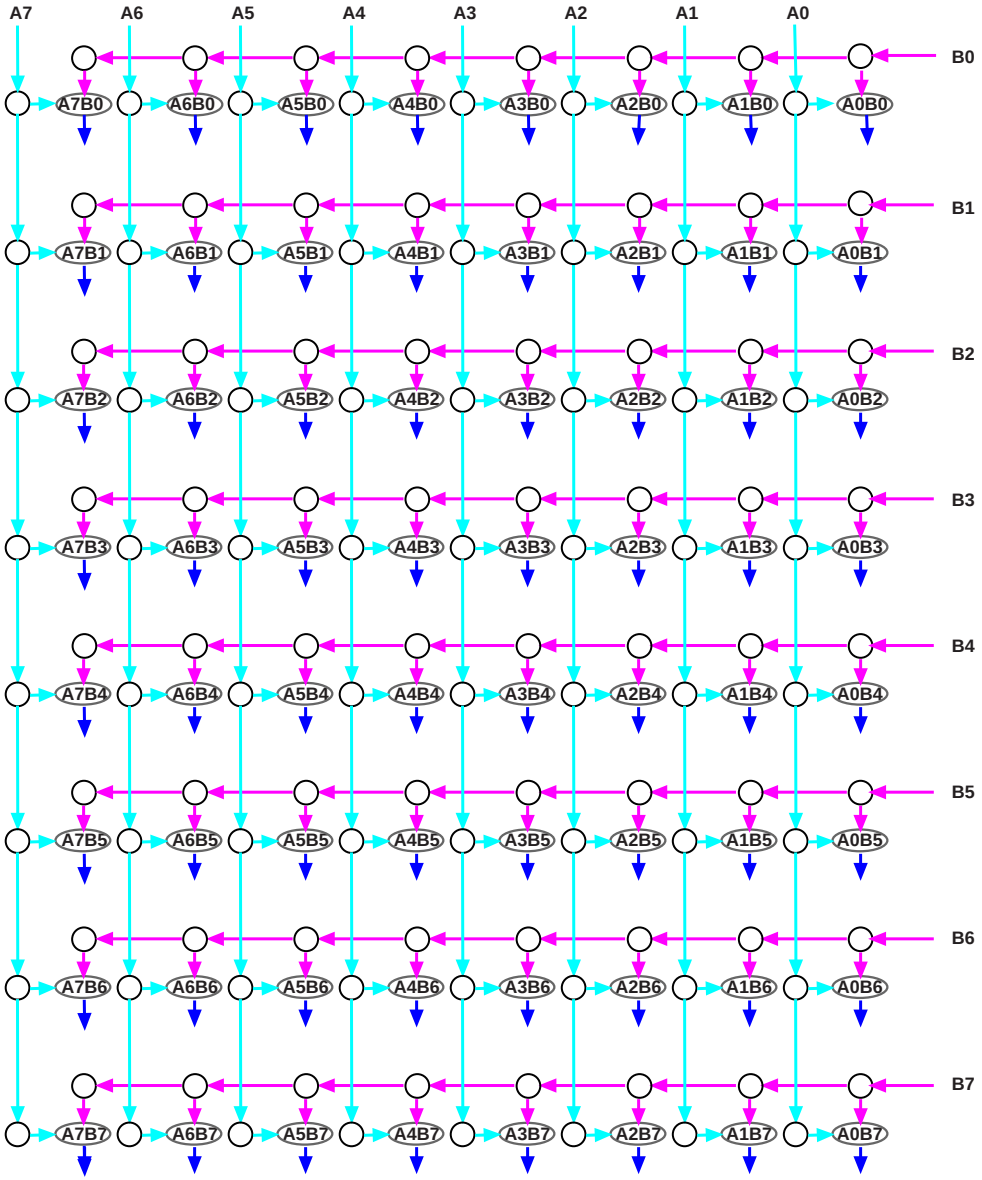


B

Verilog
unsigned8/TwoD_mult8U2.v
make mul8U2

```
mul8(A, B ->  ){
token [7-0]{1:0} A, B;
token [15-0]{1:0} C;
token [7-0][7-0]{1:0} Bbuf, Abuf;

path Abuf/i{
  dualbuf(A/i -> Abuf/i/0);
  dualbuf(B/i -> Bbuf/i/0);
  path Abuf/i/j-1{
    dualbuf(Abuf/i/j -> Abuf/i/j+1);
    dualbuf(Bbuf/i/j -> Bbuf/i/j+1);
}
// sink the output of each pipeline
(Abuf/i/7 ->  );
(Bbuf/i/7 ->  );

close A <- Abuf/i/0/#
close B <- Bbuf/i/0/#
}
}
```

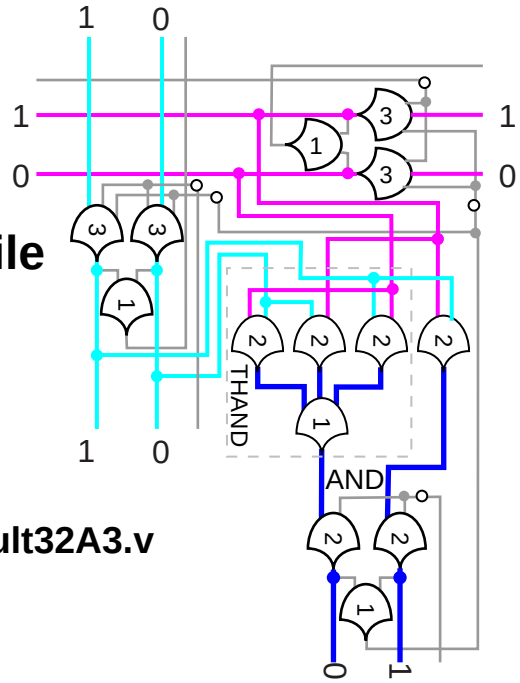
8X8 multiply array 2D pipeline structure with partial products added



Verilog
unsigned8/TwoD_mult8U3.v
make mul8U3

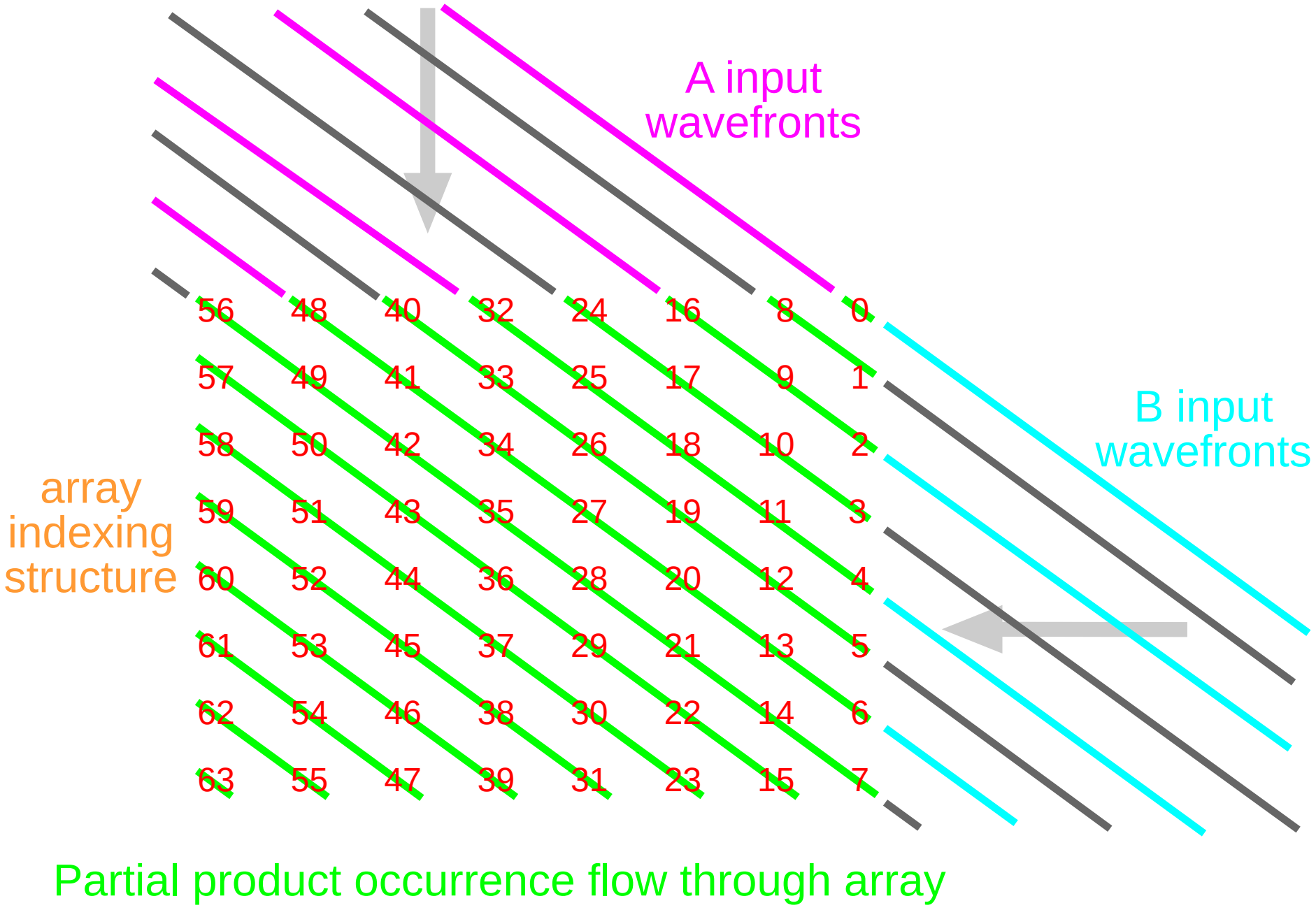
```
mul8(A, B ->  ){
token [7-0]{1:0} A, B;
token [15-0]{1:0} C;
token [7-0][7-0]{1:0} Bbuf, Abuf;

path Abuf/i{
  dualbuf(A/i -> Abuf/i/0);
  dualbuf(B/i -> Bbuf/i/0);
  path Abuf/i/j-1{
    dualbuf(Abuf/i/j -> Abuf/i/j+1);
    dualbuf(Bbuf/i/j -> Bbuf/i/j+1);
    PP(Abuf/i/j, Bbuf/i/j -> PP/i/j);
    // sink each partial product
    (PP/i/j ->  );
}
close A <- Abuf/i/0/#
close B <- Bbuf/i/0/#
}
}
```

Array Tile

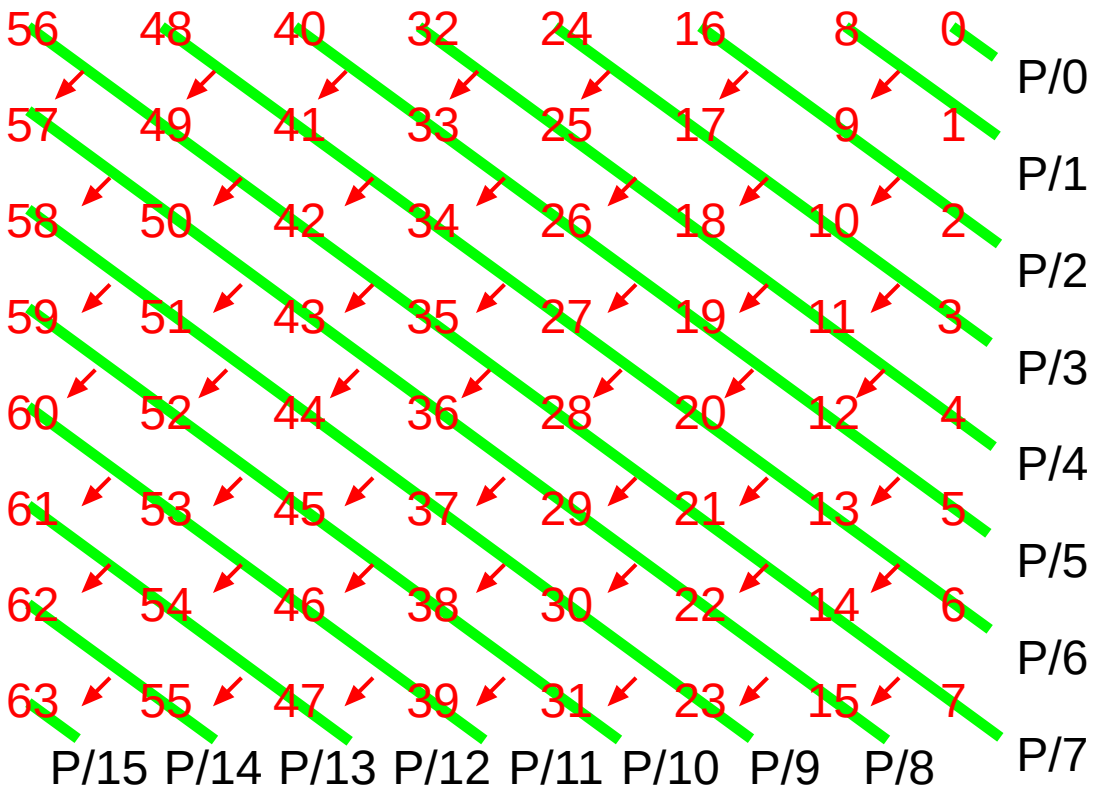


Verilog
unsigned32/TwoD_mult32A3.v
make mul32A3

8X8 multiply array 2D pipeline structure
input flow and partial product flow

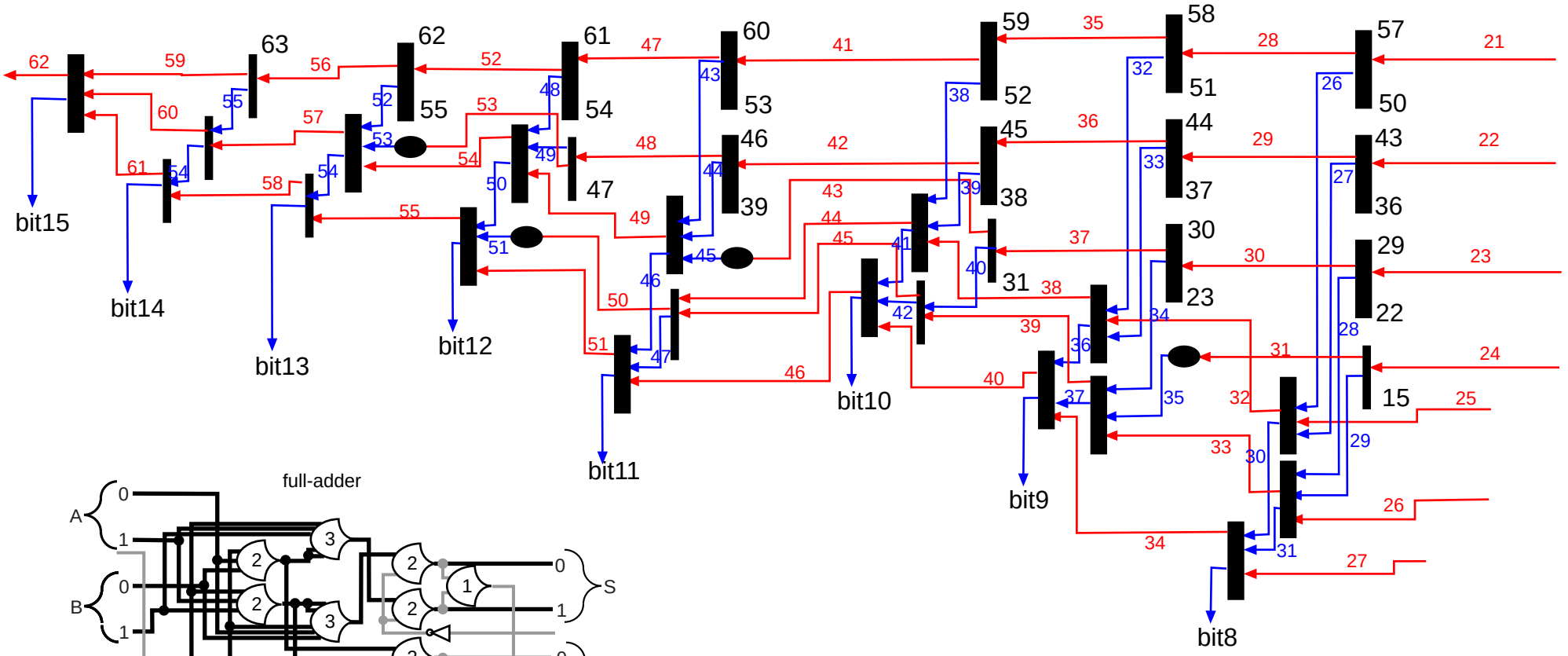


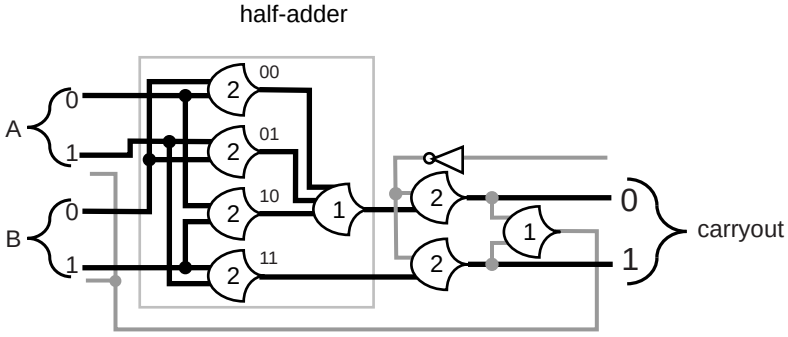
8X8 multiply array 2D pipeline structure carry flow and product flow

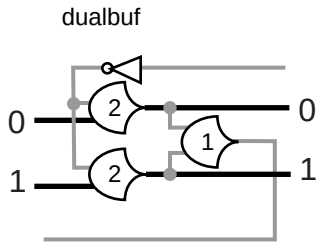


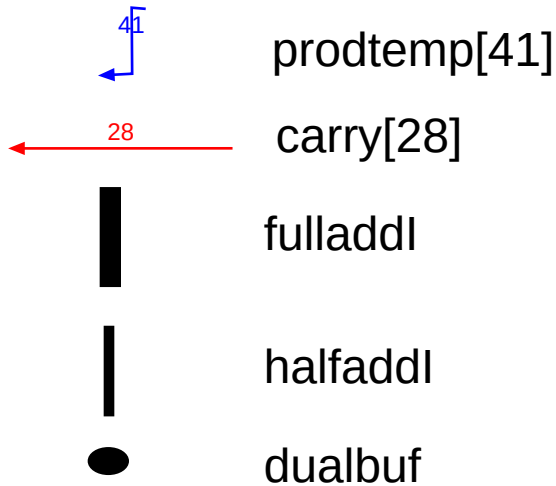
Adder forest - unsigned - high order bits

Verilog
unsigned8/TwoD_mult8U4.v
make mul8U4

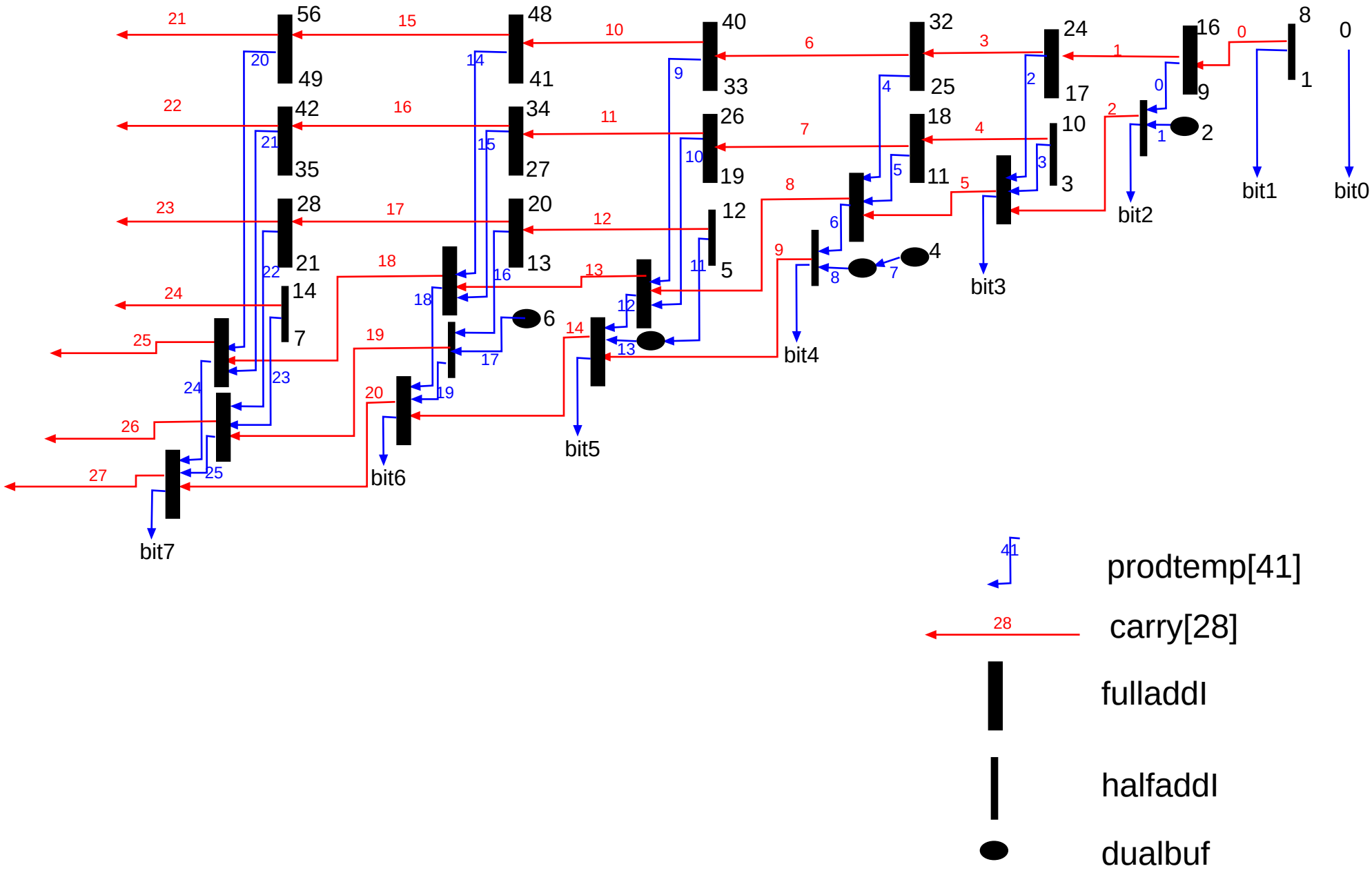




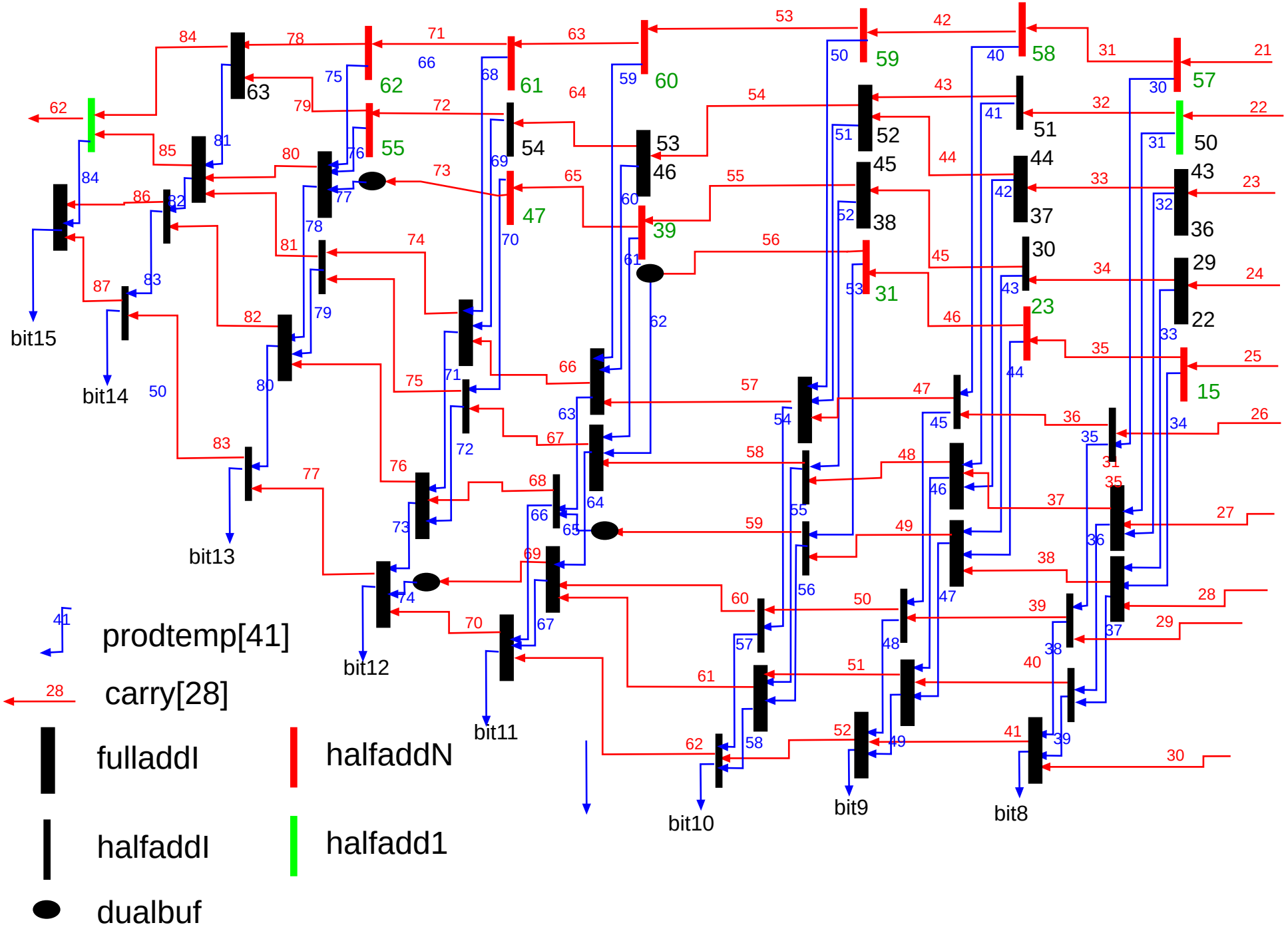




Adder forest - unsigned - low order bits



Adder forest - Baugh-Wooley 2s complement - high order bits



Adder forest - Baugh-Wooley 2s complement - low order bits