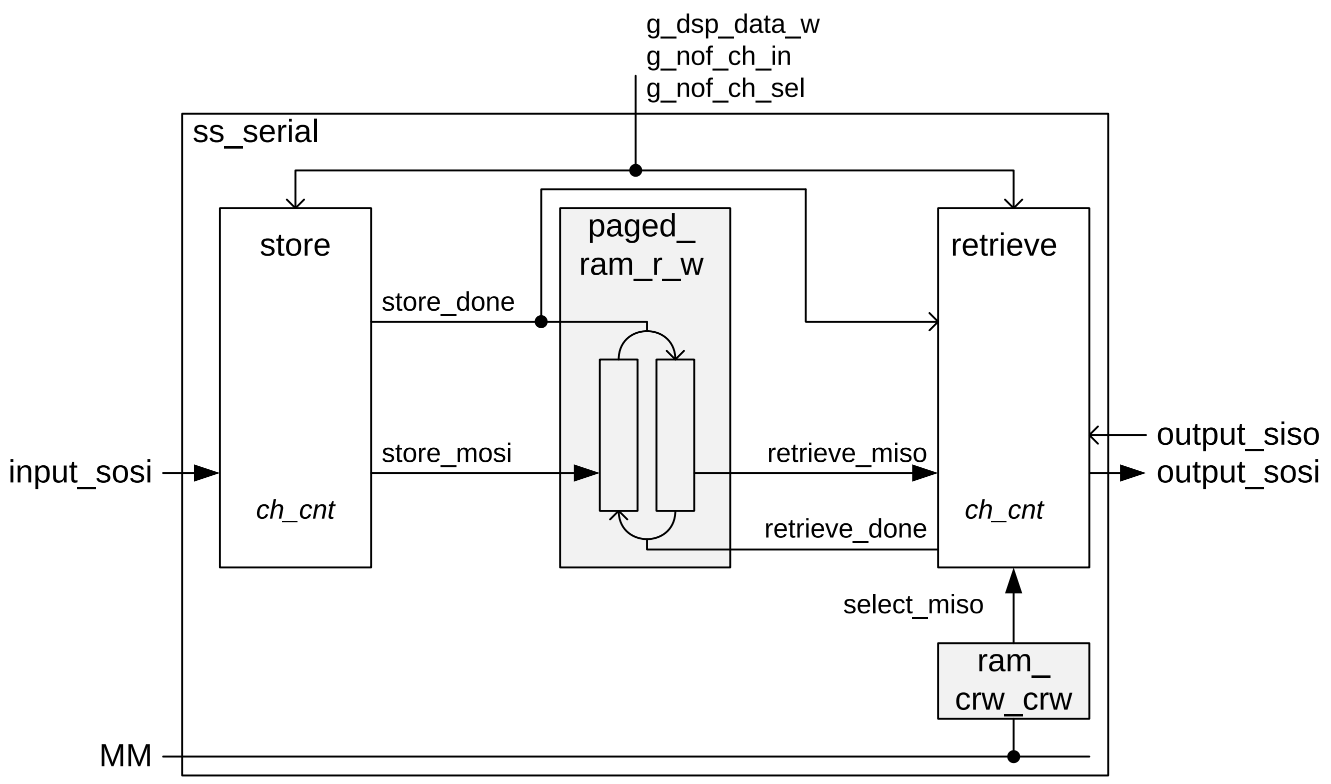
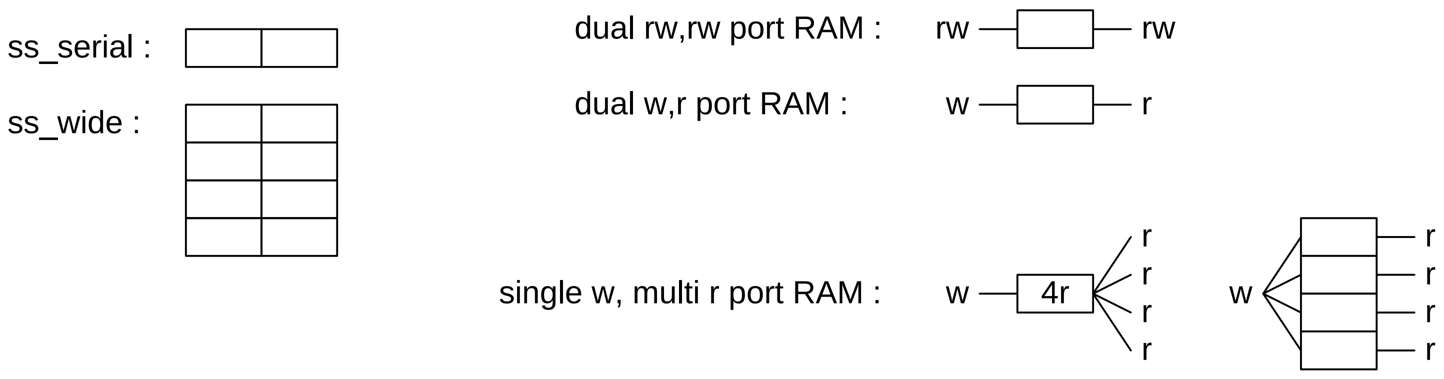
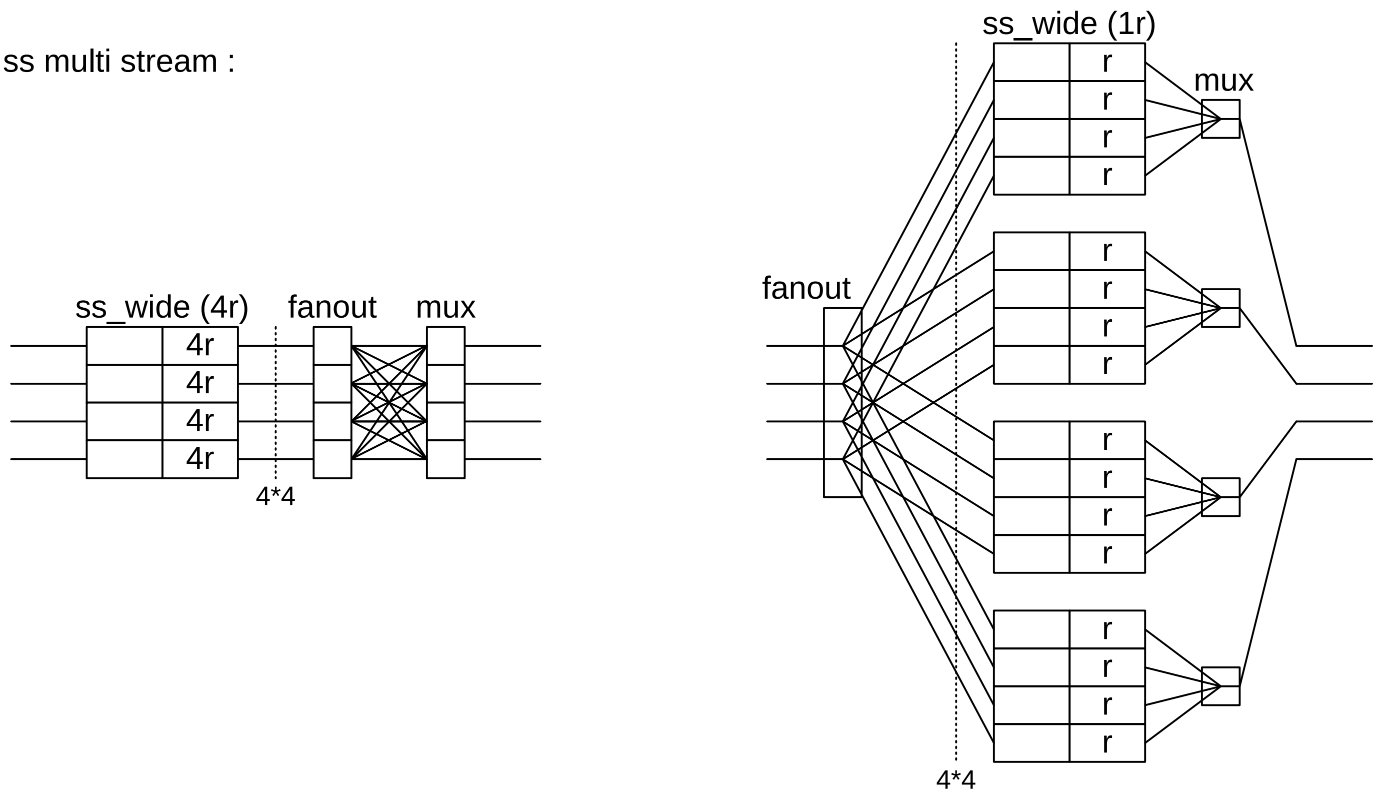
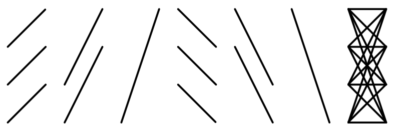
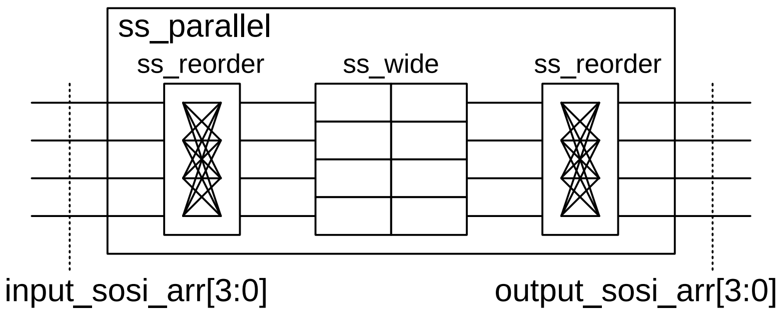


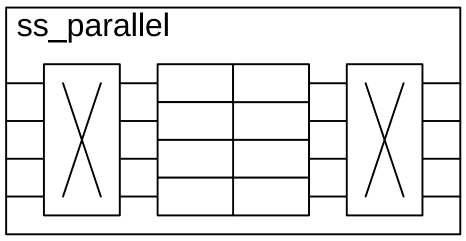


ss multi stream :

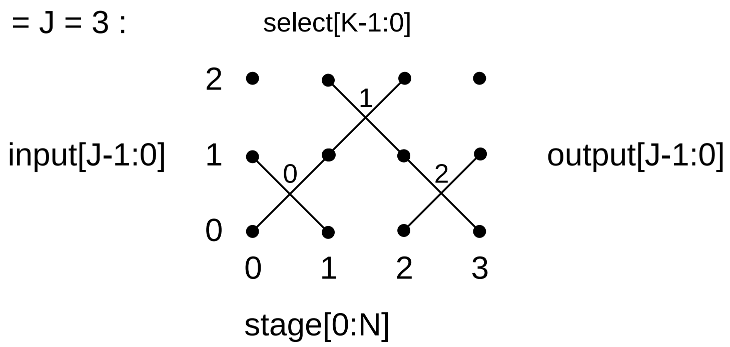




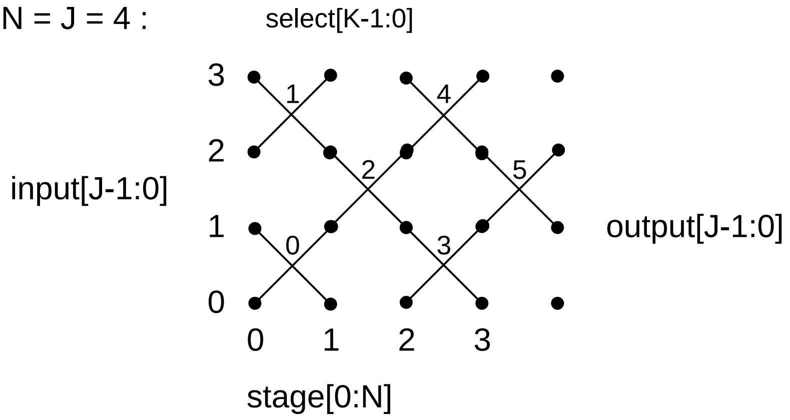




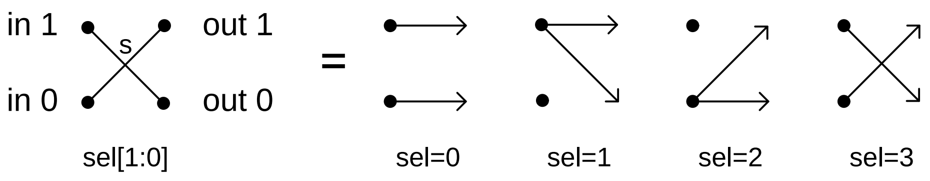
N = J = 3 :

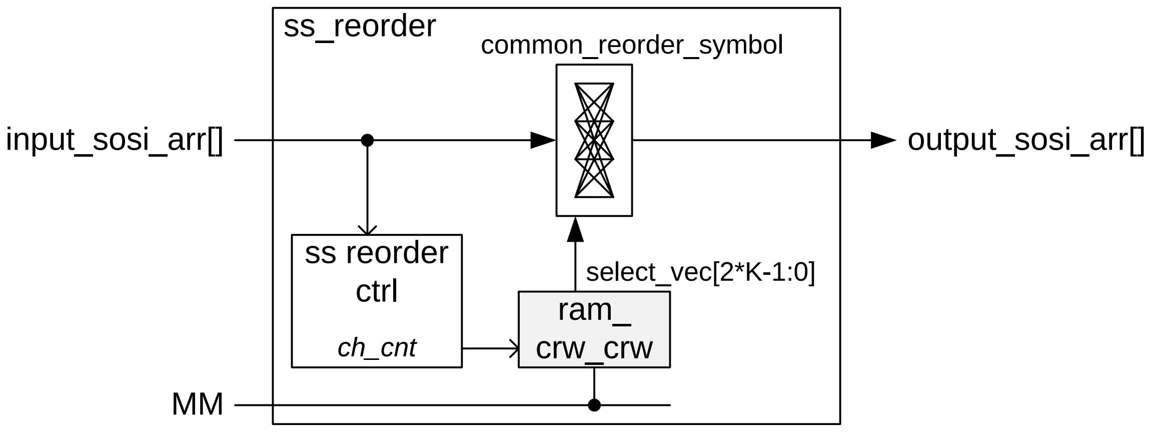


N = J = 4 :



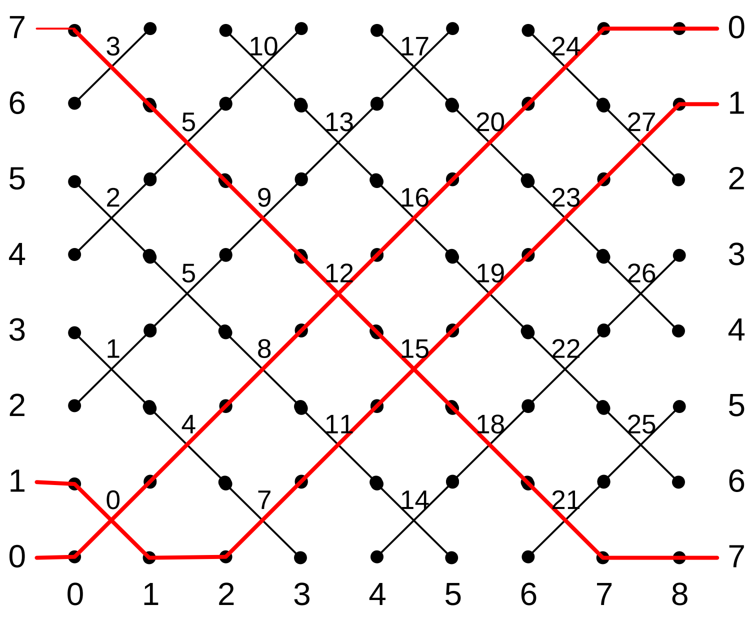
Two-port reorder cell:

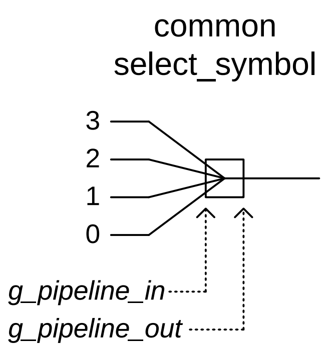


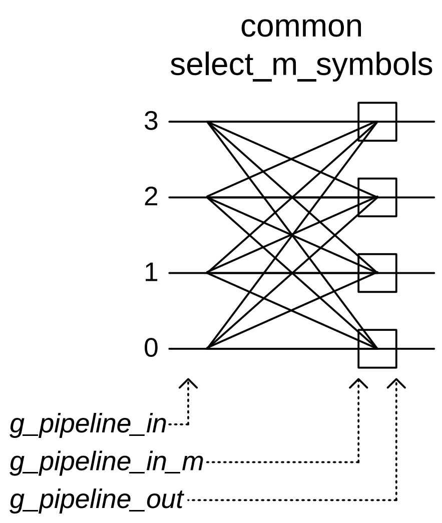


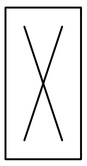
N = 8 stages
K = 28 reorder2 cells

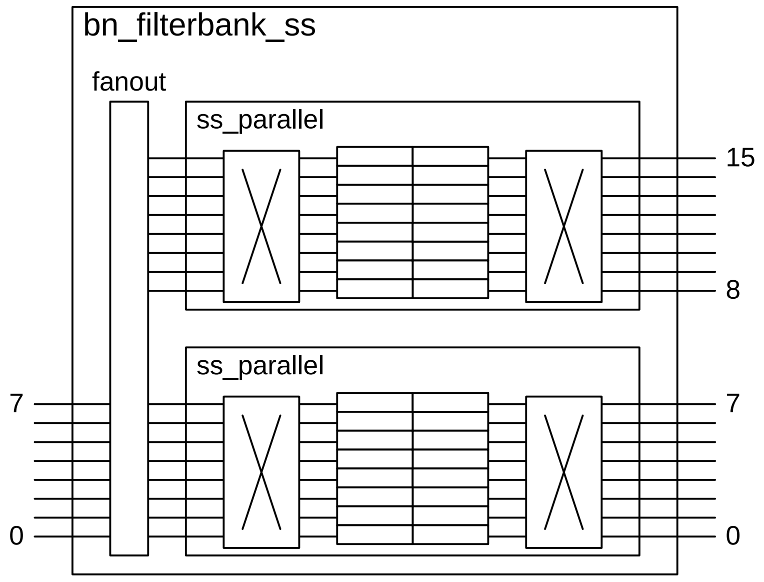
common_reorder_symbol





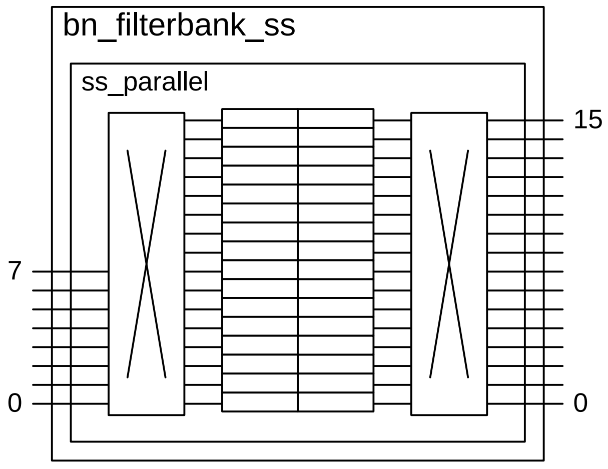



= common_select_m_symbols
or
= common_reorder_symbol

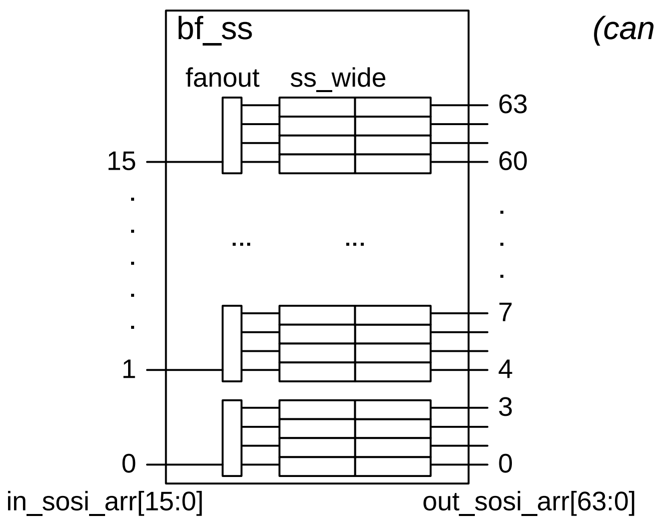


Input 8 streams:
- output of 2 WPFB
- 2 real SP per WPFB
- wideband factor P=4

Output 16 streams:
- distribute subbands to 16 FN

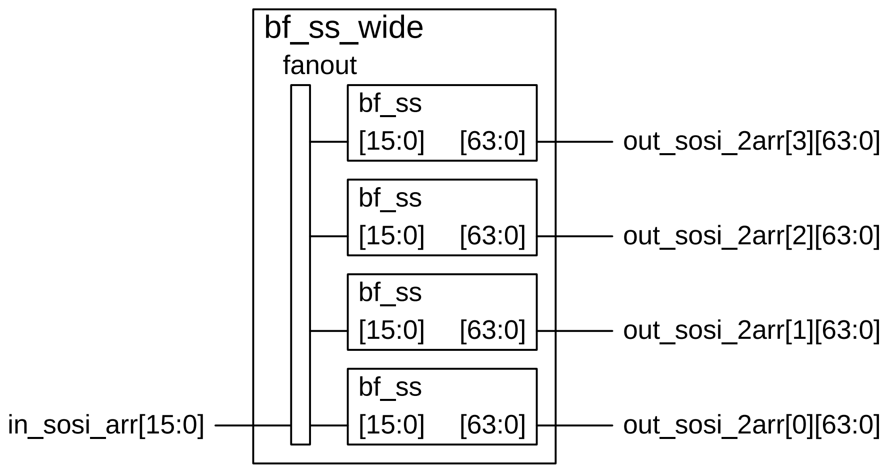


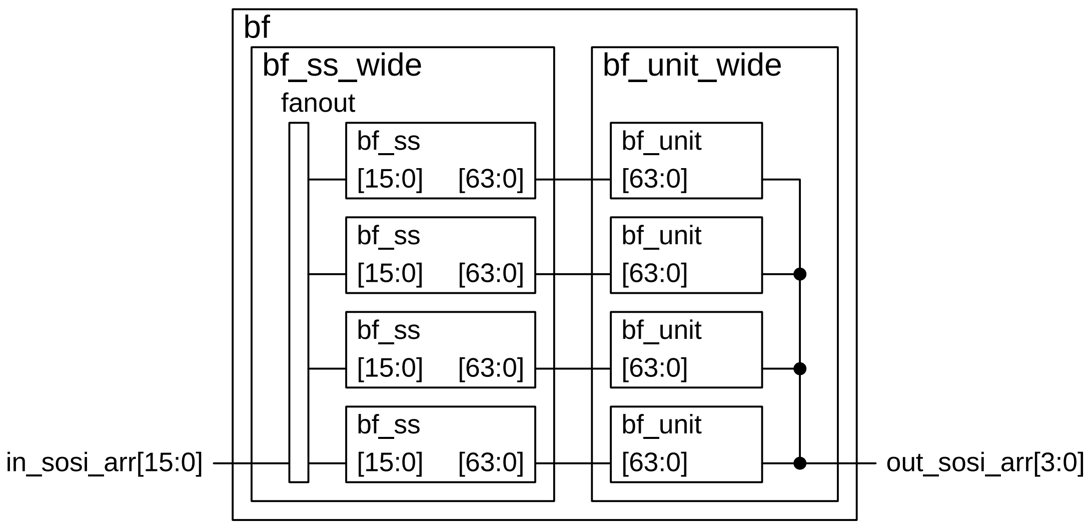
(can replace BF switch + offset FIFOs)



- from 16 BN
- 24 subbands per stream

- 64 SP
- Select form 24 subbands per stream





PFB[1:0] output for P=4 wideband factor and 4 signal paths A,B,C,D:

```
7  C384,D384, C385,D385, ... C511,d511,
6  C256,D256, C257,D257, ... C383,d383,
5  C128,D128, C129,D129, ... C255,d255,
4  C0,  D0,   C1,  D1,   ... C127,d127,

3  A384,B384, A385,B385, ... A511,B511,
2  A256,B256, A257,B257, ... A383,B383,
1  A128,B128, A129,B129, ... A255,B255,
0  A0,  B0,   A1,  B1,   ... A127,B127
```

Subband reorder by bn_filterbank_ss with ss_parallel for P=1 and 4 signal paths (SP). SP A,B on one stream and SP C,D on the other stream:

```
    4  C0,  D0,   C1,  D1,   ... C127,d127,
    0  A0,  B0,   A1,  B1,   ... A127,B127
```

input reorder:

```
  sel   0,   0,    3,   3,   ...

    4  C0,  D0,   A1,  B1,   ...
    0  A0,  B0,   C1,  D1,   ...
```

ss_serial:

```
  addr  2,   3,    0,   1,   ...
  addr  0,   1,    2,   3,   ...

    4  A1,  B1,   C0,  D0,   ...
    0  A0,  B0,   C1,  D1,   ...
```

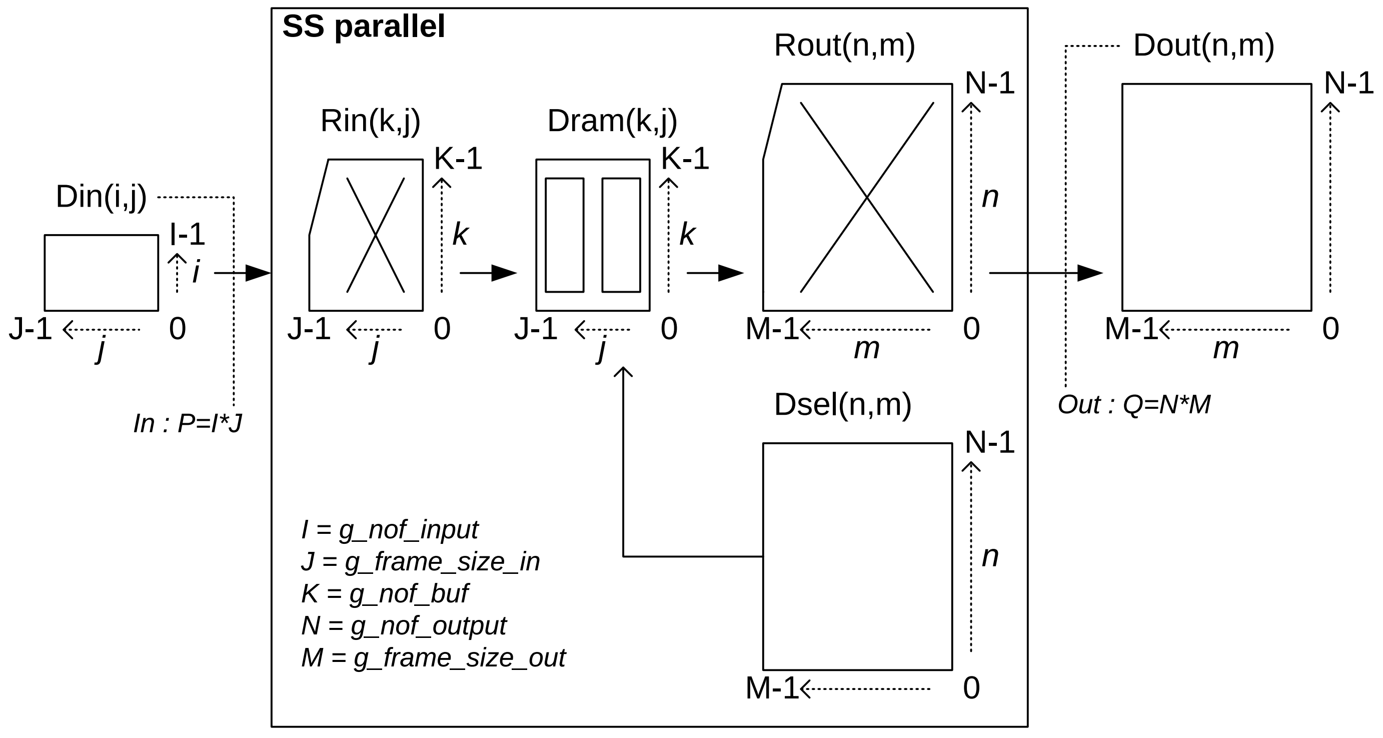
output reorder:

```
  sel   0,   0,    3,   3,   ...

    4  A1,  B1,   C1,  D1,   ... A127, B127,  C127, D127
    0  A0,  B0,   C0,  D0,   ... A126, B126,  C126, D126
```

The subbands from all 4 SP A,B,C,D are now available per stream.

```
4: 1 3 5 7  1 3 4 6  4 6 5 7
0: 0 2 4 6  0 2 5 7  0 2 1 3
```



<u>Introduction</u>
The input Din has P=IJ values. The output Dout has Q=NM values. Dout can contain any selection from Din, so in total there are P^Q possible selections for Dout. If Q > P then there are duplicates, if Q < P then some input values are not selected.

<u>Problem definition</u>
Let input Din(i,j) contain value p = i + j*I to uniquely identify each input value (0:P-1). With output values Dout selected from Din by user determine the settings for input reorder Rin, serial select Dsel and output reorder Rout.

<u>Parallel dimension: parameters I, K, N</u>
There are I parallel input streams and N parallel output streams. The dual page memory Dram(k,j) has K parallel buffers. Typically I = N, but not necessarily. For bn_filterbank_ss the basic reordering is:

i	Din	Dram	Dout
1:	1 3 5 7	1 3 4 6	4 6 5 7
0:	0 2 4 6	0 2 5 7	0 2 1 3

Typically K = N, but it can be sufficient to use K < N. It can even be necessary to use K > N. This happens when more than N input values in series need to be output in parallel, e.g. as for:

i	Din	Dram	Dout
2:		1 2 5	
1:	1 3 5	1 2 5	2 5 5
0:	0 2 4	0 3 4	0 0 2

<u>Serial dimension: parameters J, M</u>
The input Din arrives in blocks of J data. Each data block gets stored in the dual page RAM buffers and gets output as a block of M data after a page swap. If M > J then there are duplicates, if M < J then some input values are not selected for output. The output block rate is equal to the input block rate. If M > J then the input must have sufficient gap time between blocks or the output needs to be clocked at a sufficiently higher clock rate.

<u>Clock domains</u>
The MM control occurs from an independent clock domain. The SS parallel data flow operates in a single ST clock domain. Alternatively the input ST clock domain and output ST clock domain could be separated in ss_wide, to support dual clock domain operation.

Draft algorithm for automatically finding the selection settings for SS parallel

- Initialize Rin, Rout, Dsel, Dram with -1 to mark that they are unused or still free. Dependent on Dout they may remain unused.
- Start search for each Dout. This is better than starting with Din, because this makes it easier to handle duplicate data and missing data.
- For m in M

```
  For n in N
     If locate Dout(n,m) in Dram → (row, col)
        Rout(n,m) = row
        Dsel(n,m) = col
     Else
        locate Dout(n,m) in Din → (row, col)
        If find free cell in Dram(0:K-1, col) → k
           Rin(k, col) = row
           Dram(k, col) = Dout(n,m)
           Dsel(k, m) = col
           Rout(n, m) = k
        Else
           exit “Cannot make selection, need to increase K”
```

Remarks:

- If locate finds Dout value in Dram, then it also needs to also check that Rout and Dsel are free, because they may be occupied already and then K needs to be > N.