VLSI Design Automation Assignment2

Placement and Routing

Vinay Kumar Burugu M08813508

Hareesh Alamalakala M08974088

Placement algorithm: Force Directed Placement Algorithm

Data Structures used:

1. A net information vector which stores net number, source terminal, target terminal and source and target numbers.
2. A connectivity matrix to store each cell connections with other cells through nets.
3. A cell weight vector to store the connectivity size of each cell.
4. A cell placement vector to store the location of each cell in placement matrix.
5. A placement matrix to manipulate the cells and place them at the best possible locations using the force directed algorithm.

Force directed placement algorithm:

Initially, all the cells are placed randomly (all the cells are placed at consecutive positions in the placement matrix). Cells are sorted based on the connectivity of the cells. A cell with highest connectivity is chosen as the seed cell for the algorithm. Its target location is calculated based on the zero force equation (net force on the cell is equated to zero and corresponding new x and y co-ordinates are found).

The seed cell location is updated as empty. Now we have four cases based on the target location:

Case 1: Target location is empty

The seed cell is placed at target location and the target location is blocked. The seed cell is updated as locked in the cell weight vector. The algorithm continues choosing the next highest weighted cell as the seed.

Case 2: Target location is occupied

The seed cell is moved to the target location and it is blocked. The cell present at the target location is chosen as the next seed and proceeds.

Case 3: Target location is blocked.

Since the location is already blocked, the cell cannot placed at this location. So, a nearest target location is calculated by traversing in all four directions possible. If any of the locations is blocked, abort count is increased by one.

Here there are two sub cases:

Case i: New target location is empty

Same as Case 1. Cell is moved to target location and blocked. Next highest weighted element is chosen as the seed.

Case ii: New target location is occupied

Same as case 2. Cell is moved to target location and blocked. Cell present at target location is chosen as seed cell and algorithm continues.

If the abort count reaches abort limit all the locked cells are unlocked and same process is repeated. The algorithm is run until iteration limit is reached.

Routing Algorithm: Lee Algorithm

Data Structures used:

1. A struct to store lee numbers, blocks, via, net numbers, net surrounding information in both the layers.
2. A cell co-ordinate vector to store cell terminal locations.
3. A queue to run breadth first search.

Lee Algorithm:

Wave Propagation:

Lee algorithm tries to propagate a wave from source to target by assigning a number equivalent to Manhattan distance at each block. This wave propagation ends when the target point is reached. Lee propagation happens in two layers. Whenever there is a blockage in one layer, it checks in the other layer and assigns lee number if there is no blockage.

Complexity: O(L^2) L: Manhattan distance between source and target.

Back trace:

Back trace starts at target and tries to go in the same direction until it can and then take turns to reach destination whenever it cannot go in the same direction. At every stage it tries to go to the block which has a lee number less than its previous value. Whenever there is a layer change (m1 to m2 or m2 to m1), a via is inserted at that location.

Complexity: O(L)

Wave Clearing:

RESULTS FOR THE BENCHMARKS: Wave clearing is similar to wave propagation. Wave clearing starts at source and clears all the lee numbers until target is reached. Complexity: O(L^2).

Different approaches tried and challenges faced:

Since lee algorithm tries to stay in the same direction until it cannot go further, wires were drawn until it reaches any of the terminals and then take a turn. Because of this, net from the terminal at which other net took bend cannot be routed. To fix this problem, all the blocks just above the terminal are assigned only to that particular net which comes out of it. Other nets cannot use these blocks. A separate variable is used to store this information.

Initially, a single lee numbering scheme was used to find paths. All horizontal lines are considered as one layer and vertical lines are considered as second layer. So, whenever there is a bend in the path, a via is inserted.

Second approach was to try and route all the nets with metal1. If a net cannot be routed by metal 1 its back trace is cleared. All the nets which cannot be routed by metal 1 are now routed by metal2. 100% routability is not guaranteed with this approach. If two nets with metal 1 and metal2 follow the same path at certain location, no other net can cross these wires with either metal 1 or metal 2. So, this method is discarded.

Finally, a net is routed with metal1 until it can route and a via is inserted and its routing is continued with metal2 until there are blockages for metal 1. This process is continued until the target is reached. Two metal layer lee numbering and back trace are used to achieve this.

Placement and routing interaction:

Both placement and routing algorithms have a common data structure to store net information obtained from the bench mark file. Cell connectivity matrix is formulated while storing this net information in vector form. Finally, after running force directed algorithm, placement matrix is converted to chip grid matrix which gives sufficient spacing between each cell of size 6 by 6.

After performing lee algorithm, chip grid matrix is converted to magic file. Each layer type is stored as a variable in chipUnitBlock data structure. Based on the values of the variables at each grid location m1, m2, via locations are routed.

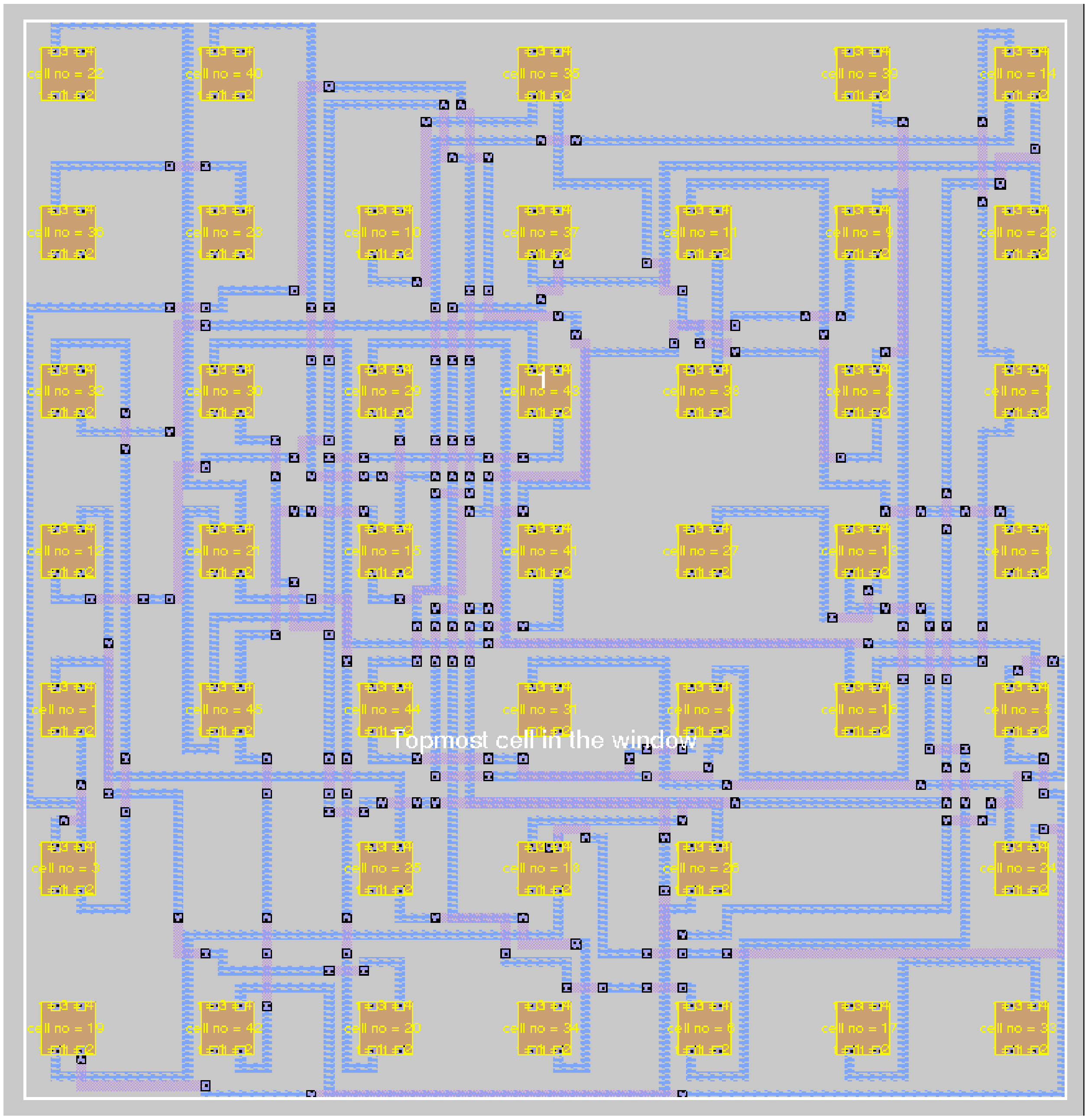
How to improve the algorithm:

In the current algorithm, net tries to route in metal1 whenever possible. It routes in metal2 only when it cannot route in metal 1. Lee numbering is done in the fashion mentioned. So, this increases the no. of vias enormously. To reduce the vias, the algorithm can be modified such that the net is routed in the same layer until it can be routed and switch to the other layer only when it cannot be routed in the current layer

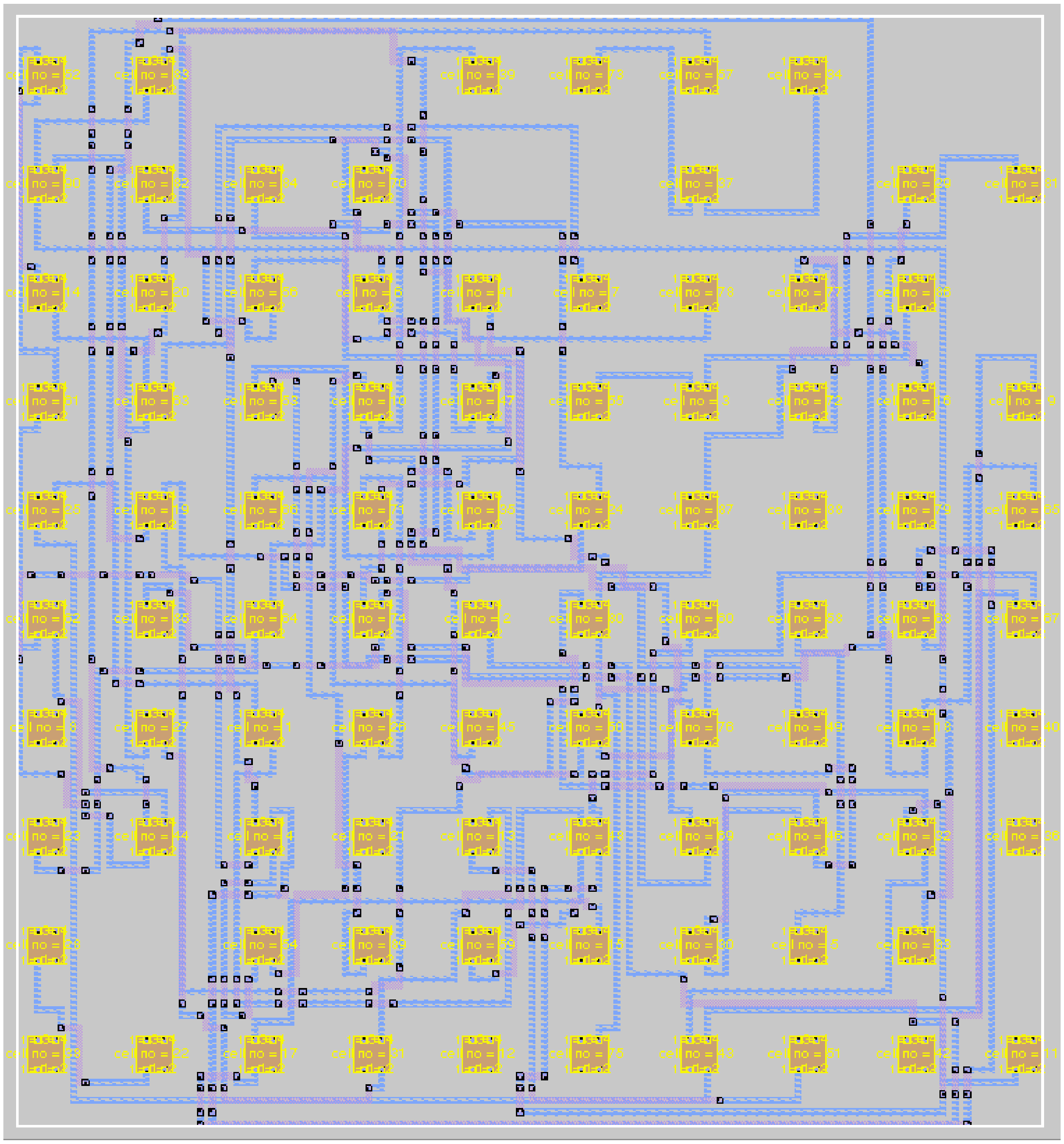
RESULTS FOR THE BENCHMARKS:

Benchmark number	Bounding box dimensions	Bounding box area	Wire Length Before placment	Wire Length After placement	Wire Length After routing	No of nets routed	Number of vias	Memory Used (KB)	Execution Time
B1	118x112	13216	226	199	3963	41	238	4500	26msec
B2	170x184	31280	709	444	8104	71	465	7672	66 msec
B3	245x254	62230	2069	1338	15034	112	869	13480	2.07 sec
B4	336x357	119952	6253	3555	33059	118	1521	23512	7.67 sec
B5	947x954	903438	18303	13604	210458	437	10821	142840	1.45min
B6	1141x1134	1293834	29764	20278	300606	519	13863	198540	2.39min
B7	830x854	708820	19992	16742	213150	340	8983	105084	1.19min
B9	512x519	265728	9640	6217	45192	318	2734	52024	7.49msec
B10	440x445	195800	3987	2704	14804	173	1000	39088	1.79 msec

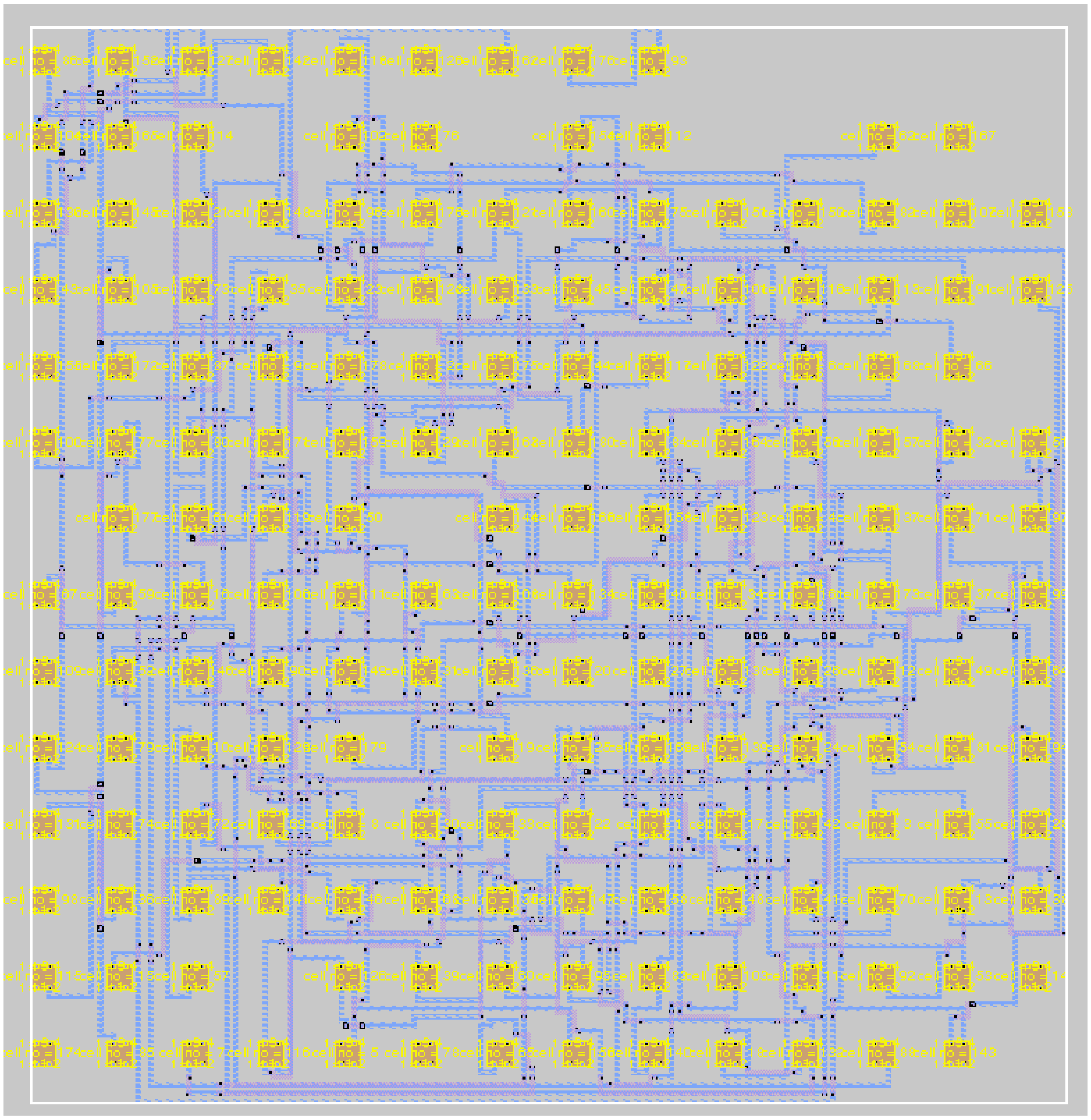
B1:



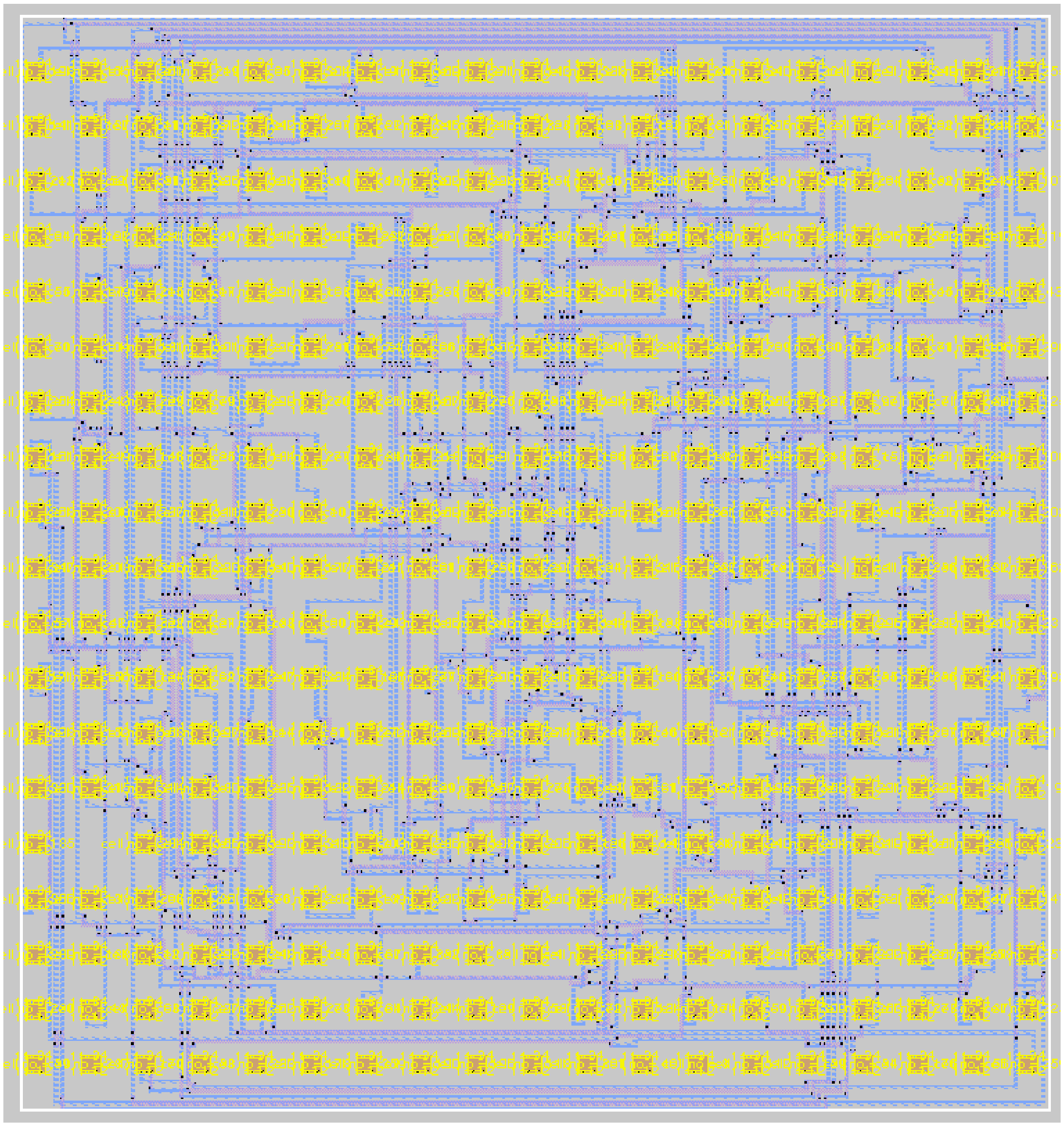
B2:



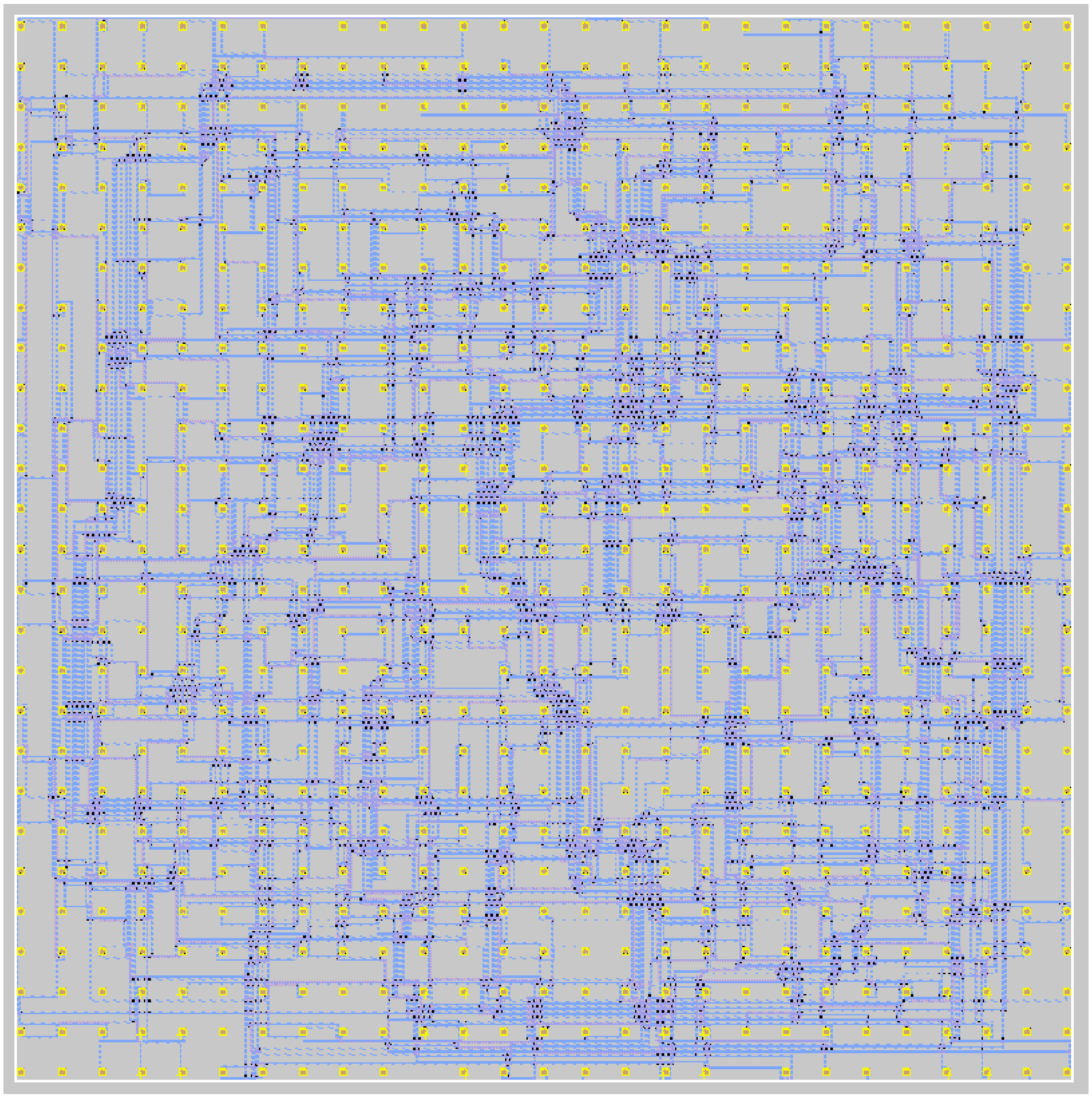
B3:



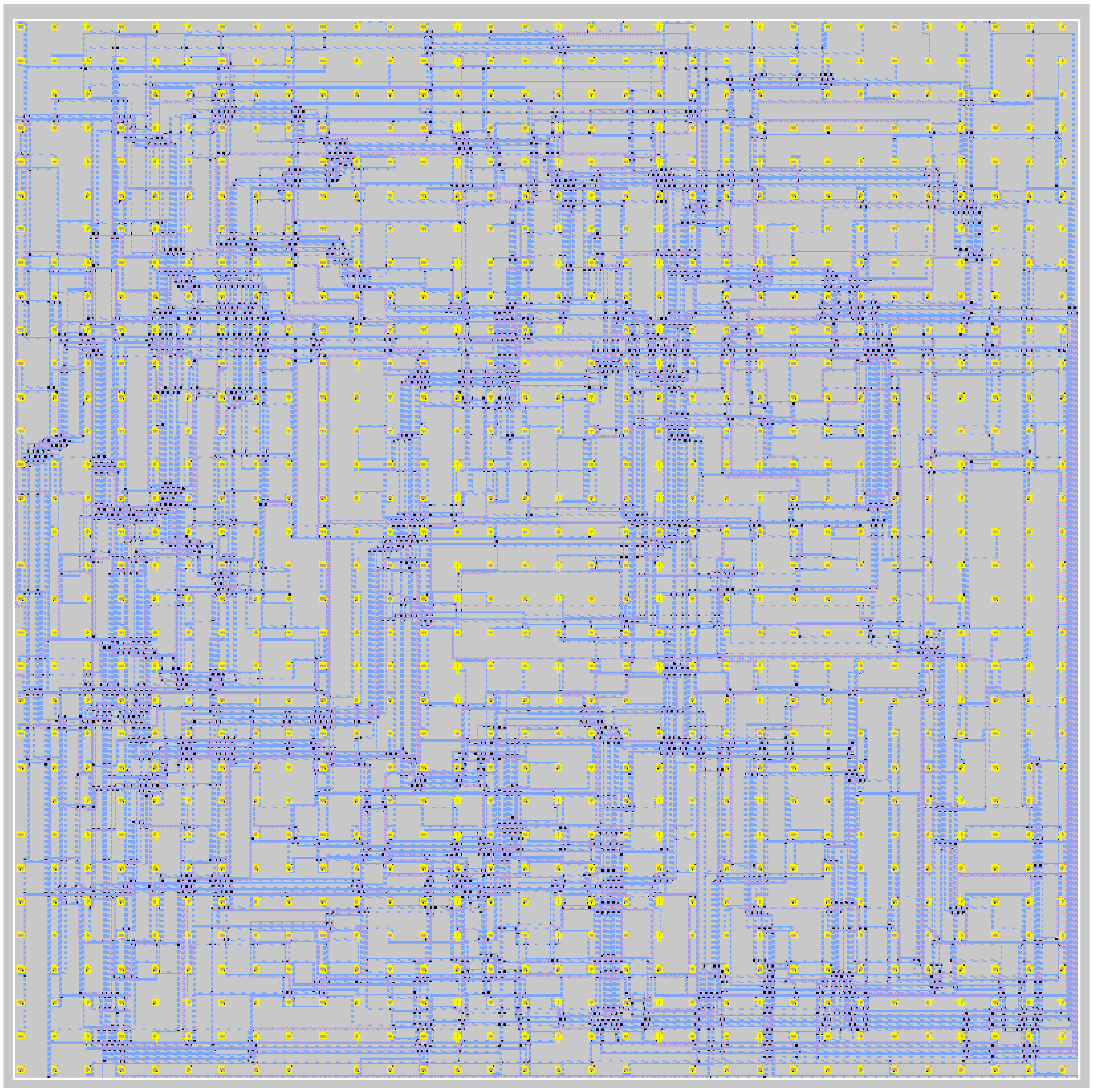
B4:



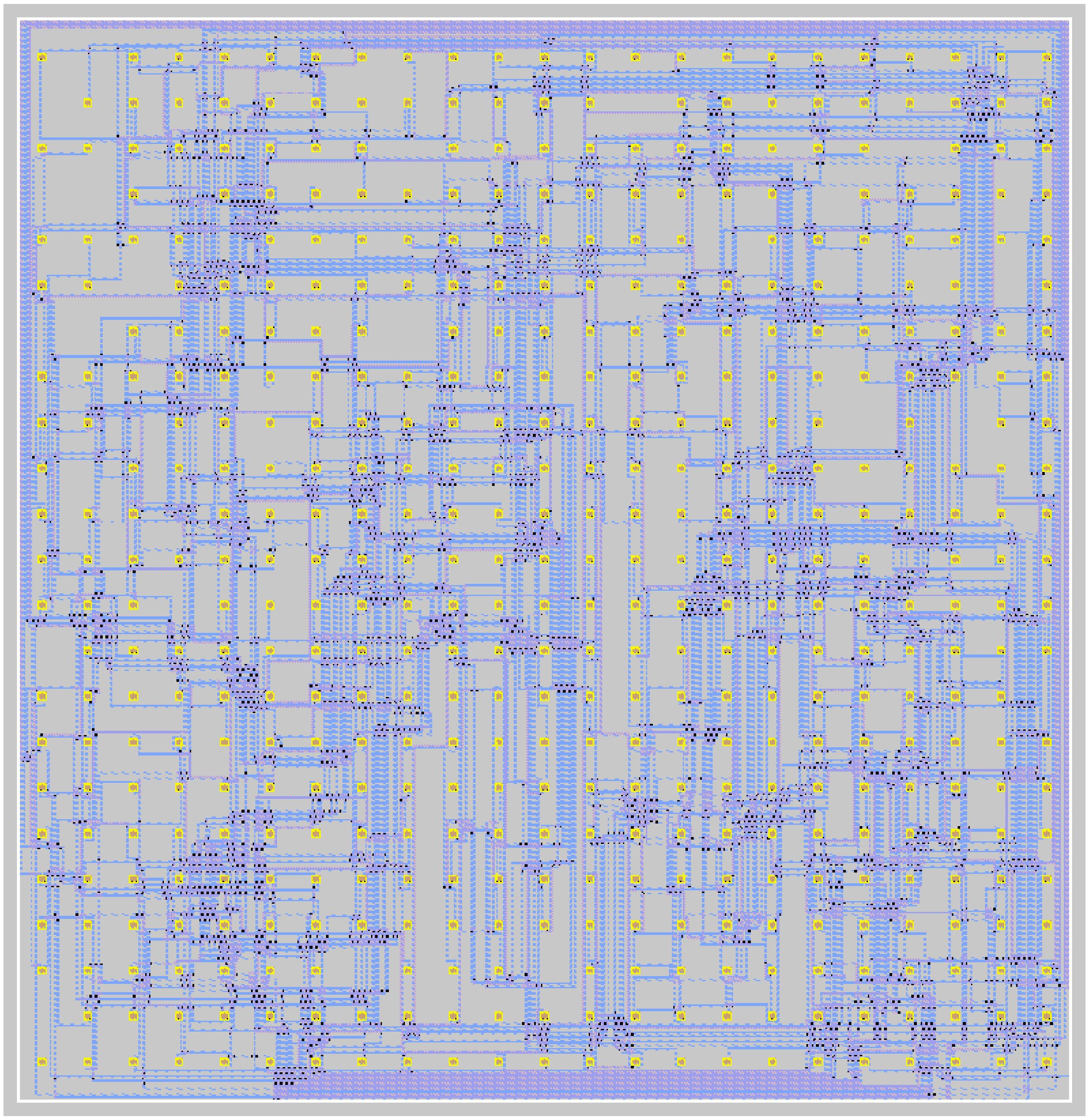
B5:



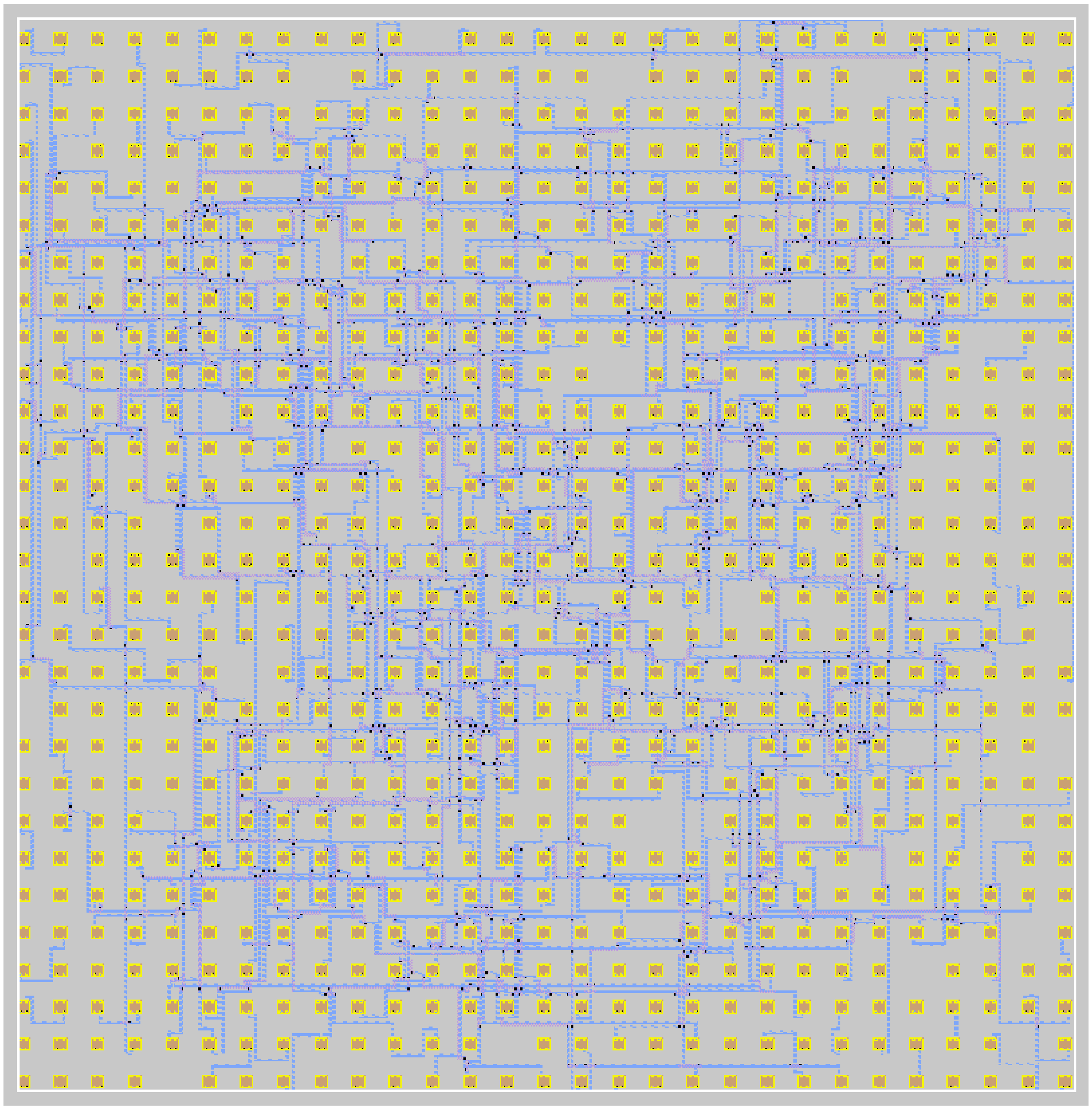
B6:



B7:



B9:



B10: