XYTable[] array generator

Intro

This Arduino sketch will create a lookup table for LED projects instead of writing and using mapping code. It uses LEDMatrix definitions (ex: HORIZONTAL_ZIGZAG_MATRIX) to define the LED mapping.

The LED mapping apps I have found all have shortcoming on the size or layout of the matrix. Especially for blocks or cells within the matrix like the popular 8x8 blocks. This sketch includes:

Up to 32k LEDs
Small to very large matrices – laid out in any direction with or w/o zigzag
Matrix can be made of blocks (cells) of any size that of any size– laid out in any direction with or w/o zigzag in the block and block layout within the matrix.
Produces a report on the Serial Terminal of the specified configuration and the resulting mapping array.
Simple cut and paste into your header file.
Arduino code in small single purpose functions that are easy to modify

LEDMatrix reference:
https://jorgen-vikinggod.github.io/LEDMatrix has an excellent tutorial on mapping definitions.

OUTPUT: will be on the serial terminal so you can review the results and make changes.
TO SAVE: copy and paste the terminal output array to a file.

Parameter List Guide

========== for width and height ==========

Do you have an LED array to start with or an IRREGULAR layout? You can #include it in step 1.
If you have an irregular array, you should 1st create the array with missing LEDs here:
FastLED XY Map Generator: *https:macetech.github.io/FastLED-XY-Map-Generator*

NOTE: make the array simple, left to right and top to bottom, no zigzag.
We will make all these changes here.

WHEN DESIGNING BLOCKS IN A MATRIX, COMBINATIONS CAN GIVE SURPRISING RESULTS.
IF THE RESULT IS NOT WHAT YOU EXPECTS, REVIEW EACH STEP'S SETTINGS AGAIN!

STEPS

1. array name must be*: "const uint16_t PROGMEM XYTable[][x] = {}"* or:
   "const uint16_t XYTable[][x] = {}" The [x] length is required.
   PROGMEM is optional. Set PROG_MEM below as true or false to match

```
#define HAVE_ARRAY false
#if HAVE_ARRAY
    #include "XYTable_LookUp.h"          edit your file name here
#endif
```

2. Set the overall Panel size in number of LEDs (POSITIVE VALUES ONLY). Previous LEDMatrix versions use a negative value for reserved (right to left) and (bottom to top). Use step 4 below to do this.

   #define MATRIX_WIDTH 32 NOTE: Previous LEDMatrix versions use a negative values
   #define MATRIX_HEIGHT 32 NOTE: Previous LEDMatrix use a negative value values

3. How are the LEDs in the total matrix organized? (Regardless of and blocks or cells making up the matrix).

   >>>>> If using BLOCKs/cell to make up your matrix, this is n/a. See steps 6 and 7

   MatrixType_t matrix_type = HORIZONTAL_MATRIX;
   Options: HORIZONTAL_MATRIX, VERTICAL_MATRIX, HORIZONTAL_ZIGZAG_MATRIX, VERTICAL_ZIGZAG_MATRIX

4. What direction does the FIRST row go?
   What direction does the FIRST column go?

   >>>> These flip THE ENTIRE MATRIX even LED order inside of BLOCKS if present
   >>>> TO change the order of BLOCKS see the blocks see step 7.

   MatrixOrder_horizDir horizDir = LEFT_2_RIGHT; Options: LEFT_2_RIGHT, RIGHT_2_LEFT
   MatrixOrder_vertDir vertDir = TOP_DOWN; Options: TOP_DOWN, BOTTOM_UP

================ BLOCKS options ================

5. LEDs make a BLOCK (cell), BLOCKS make up a MATRIX (panel). if you have one long LED string in your display set HAS_BLOCK false and ignore these BLOCK values

   #define HAS_BLOCKS true Is this matrix made up of block/cells of LEDs?
   #define MATRIX_TILE_WIDTH 8 width of each matrix BLOCK/CELL (not total display)
   #define MATRIX_TILE_HEIGHT 8 height of each matrix BLOCK/CELL

   #define MATRIX_TILE_H 4 the number of tiles arranged horizontally
   #define MATRIX_TILE_V 4 the number of tiles arranged vertically

6. How are the LEDs organized inside the block/cell?

   BlockType_t blockOrg = HORIZONTAL_BLOCKS;
   Options: HORIZONTAL_BLOCKS, VERTICAL_BLOCKS, HORIZONTAL_ZIGZAG_BLOCKS, VERTICAL_ZIGZAG_BLOCKS

7. a) How are the block/cells organized in the matrix?

   BlockType_t blocksInMatrix = HORIZONTAL_BLOCKS;

Options: HORIZONTAL_BLOCKS, VERTICAL_BLOCKS, HORIZONTAL_ZIGZAG_BLOCKS, VERTICAL_ZIGZAG_BLOCKS

b) These 2 flip the order of the BLOCKS in the matrix. The LED order inside the blocks stay the same. To flip everything including the LEDs inside the blocks see step 4.

MatrixOrder_horizDir H_blockDir = LEFT_2_RIGHT; Options: LEFT_2_RIGHT, RIGHT_2_LEFT
MatrixOrder_vertDir V_blockDir = TOP_DOWN; Options: TOP_DOWN, BOTTOM_UP

8. DEBUGGING: Add h and v hash marks between blocks for easier viewing you can delete in an editor after copying.

#define TABLE_DIVIDERS true Options: true, false

========== End BLOCKS ==========

9. DONE - compile and run