```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48 /* GOL LED Matrix Clock - Alex McAlpine - www.BrainLubeOnline.com ********************************************************************
49  * 10-03-2013 ********************************************************************************************************************
50  * Every second, the time display disintegrates following the rules of John Conway's Game of Life using the pixels of the display. **
51  * Clock iterates by the second. ************************************************************************************************
52  * Every minute the clock resets the image to the time and restarts the disintegration process. ********************************
53  * There are three modes of operation (Ideally) : *******************************************************************************
54  * ******************************** 2D-Array Loopback ****************************************************************************
55  * ********************************** Ex. Element [0][0] of array is directly effected by the last element[*][*] of the ***********
56  * ********************************* GOL matrix array. ****************************************************************************
57  * ******************************** Direct Finite Display Array *****************************************************************
58  * ********************************** The limits of the display are the limits of the GOL matrix array. ***************************
59  * ******************************** Arbitrary Finite Array **********************************************************************
60  * ********************************** The limits of the game matrix are user defined. ******************************************
61  * The rules may be found here: http://en.wikipedia.org/wiki/Conway%27s_Game_of_Life ***************************************************
62  * http://www.bitstorm.org/gameoflife/ is a GREAT site I used for character development to view the disintegrations. ****************
63  * I Would like to 86 the FORs for recusion, and use pointers for the glyphs; also use boolean arrays or bitshifting where possible.*
64  * I might add a fogged acrylic block to the LED Array to defuse the light. ***********************************************************
65  * Dislikes: ********************************************************************************************************************
66  * ***************** I have the same problem with varying brightnesses of different combinations of LED pixels. This is becuase I *****
67  * ***************** have my current limiting resistors on each of the row pins, therefore if I light more than one LED on that row ***
68  * ***************** (for example, multiple columns) the LEDs must share the limited current, a common problem. ***********************
69  * ***************** The solution would be to add a decrementing dwell time that counts the number of elements currently "on" *********
70  * ************************************************************************************************************************************/
71
72 //Timing of the disintegrations is not prefect, but the time will reset to the correct minute when iterations are complete.
73 //This means the clock will display time +-1 minute of the calculated time at this point.
74
75 //Preprocessor Directives
76 #include "mbed.h"
77
78 DigitalOut RowOut[10]= {p5,p6,p7,p8,p9,p10,p11,p12,p13,p14};
79 DigitalOut ColOut[16]= {p15,p16,p17,p18,p19,p20,p21,p22,p30,p29,p28,p27,p26,p25,p24,p23};
80 Serial pc(USBTX, USBRX); // TX, RX - Serial Debug
81
82 // Global Variables
83 short int GridArray[12][18];            //Later initialized to 0; //Grid for LED Matrix
84 short int rowindex, colindex;           //2-D loop index variables
85 unsigned short Glyphs[10][5][4];       //3-D Array for Number Glyphs
86 char day[16];                           //Char time register
87 int m = 0;                              //Minute Register
88 int h = 0;                              //Hour Register
89 unsigned short GridCellNext[10][16];   //Register for Next Grid Itteration
90
91
92 const short int high = 1;
93 const short int low = 0;
94 const float dwell = .004;               //For Persistance of Vision (POV) Timing
```

```
90
91  void parser(void);
92  void ArrayInit(void);
93  void CharSetter(int h, int m);
94  void GOLprocessor(void);
95
96
97
98
99  int main()
100 {
101 struct tm t;            //Time Function Allocations      //Set the time by the variables below
102     t.tm_sec = 00;          // 0-59
103     t.tm_min = 36;          // 0-59
104     t.tm_hour = 1;          // 0-23
105     t.tm_mday = 17;          // 1-31
106     t.tm_mon = 10;          // 0-11
107     t.tm_year = 113;        // year since 1900
108
109     time_t seconds = mktime(&t); // convert to timestamp
110     set_time(mktime(&t));        //SET RTC Time
111
112     ArrayInit();
113
114     while(1)
115
116     {
117         time_t seconds = time(NULL);
118         strftime(day,16,"%M", localtime(&seconds));     //get time into 'day' char
119         m = atoi(day);                                  //convert to int (minute)
120         strftime(day,16,"%H", localtime(&seconds));     //get time into 'day' char
121         h = atoi(day);                                  //convert to int (hour)
122         CharSetter(h,m);                                 //Arranges Glyphs to show the time
123         parser();
124         parser();
125         GOLprocessor();
126
127
128
129
130
131
132     }//end big main() while
133 }//end main()
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148 void GOLprocessor()
149 {
150     for(int i = 0; i < 59; i++) {
151
152         for(rowindex = 0; rowindex < 10; rowindex++) { //Reset GridCellNext
153             for(colindex = 0; colindex < 16; colindex++) {
154                 GridCellNext[rowindex][colindex] = 0;
155             }
156         }
157
158         for(rowindex = 1; rowindex < 11; rowindex++) {  //Charge GridCellNext with judgement Math for GOL Tests
159             for(colindex = 1; colindex < 17; colindex++) {
160                 GridCellNext[rowindex - 1][colindex - 1] = GridArray[rowindex -1][colindex -1]+ GridArray[rowindex -1][colindex]+ GridArray[rowindex -1][colindex +1]+ Gr
161             }
162         }
163
164         for(rowindex = 0; rowindex < 10; rowindex++) { //Conduct GOL Rule Tests and record onto GridArray
165             for(colindex = 0; colindex < 16; colindex++) {
166                 if(GridCellNext[rowindex][colindex] == 3) {
167                     GridArray[rowindex +1][colindex+1] = 1;
168                 }
169                 if(GridCellNext[rowindex][colindex] > 3) {
170                     GridArray[rowindex +1][colindex+1] = 0;
171                 }
172
173                 if(GridCellNext[rowindex][colindex] < 2 ) {
174                     GridArray[rowindex +1][colindex+1] = 0;
175                 }
176                 if(GridCellNext[rowindex][colindex] == 2 || GridCellNext[rowindex][colindex] == 3 ) {
177                     if(GridArray[rowindex +1][colindex+1] != 0) {   //If there is no life, then there will remain no life.
178                         GridArray[rowindex +1][colindex +1] = 1;
179                     }
180                 }
181
182             }
183         }
184
185         parser();
186     }//END for
```

```
    }//END for

}//END GOLprocessor()




void CharSetter(int h,int m)
{
    pc.printf("\033[1;48HTime: %d:%d",h,m);
    pc.printf("\033[2;1H");

    if(h >=10) {
        h = h -10;
        for(rowindex = 0; rowindex < 5; rowindex++) {
            for(colindex = 0; colindex < 4; colindex++) {
                GridArray[rowindex+1][colindex+1]= Glyphs[1][rowindex][colindex];
                //pc.printf("%d",Glyphs[1][rowindex][colindex]);
            }//end for

            for(colindex = 0; colindex < 4; colindex++) {
                GridArray[rowindex+1][colindex+5]= Glyphs[h][rowindex][colindex];
                //pc.printf("%d",Glyphs[h][rowindex][colindex]);
            }//end for

            if(m >= 10) {
                short int decm = m / 10;
                for(colindex = 0; colindex < 4; colindex++) {
                    GridArray[rowindex+1][colindex+9]= Glyphs[decm][rowindex][colindex];
                    //pc.printf("%d",Glyphs[decm][rowindex][colindex]);
                }//end for

                short int smallm = m - (decm*10);
                for(colindex = 0; colindex < 4; colindex++) {
                    GridArray[rowindex+1][colindex+13]= Glyphs[smallm][rowindex][colindex];
                    //pc.printf("%d",Glyphs[smallm][rowindex][colindex]);
                }//end for
            }//end if
            else {
                for(colindex = 0; colindex < 4; colindex++) {
                    GridArray[rowindex+1][colindex+9]= Glyphs[0][rowindex][colindex];
                    //pc.printf("%d",Glyphs[0][rowindex][colindex]);
                }//end for

                for(colindex = 0; colindex < 4; colindex++) {
                    GridArray[rowindex+1][colindex+13]= Glyphs[m][rowindex][colindex];

                    //pc.printf("%d",Glyphs[m][rowindex][colindex]);
                }//end for
            }//end else
            //pc.printf("\n\r");
        }//end for
    }//end first big if
    else {
        for(rowindex = 0; rowindex < 5; rowindex++) { //Reset GridCellNext
            for(colindex = 0; colindex < 4; colindex++) {
                GridArray[rowindex+1][colindex+1]= Glyphs[0][rowindex][colindex];
                //pc.printf("%d",Glyphs[0][rowindex][colindex]);
            }//end for

            for(colindex = 0; colindex < 4; colindex++) {
                GridArray[rowindex+1][colindex+5]= Glyphs[h][rowindex][colindex];
                //pc.printf("%d",Glyphs[h][rowindex][colindex]);
            }//end for

            if(m >= 10) {
                short int decm = m / 10;

                for(colindex = 0; colindex < 4; colindex++) {
                    GridArray[rowindex+1][colindex+9]= Glyphs[decm][rowindex][colindex];
                    //pc.printf("%d",Glyphs[decm][rowindex][colindex]);
                }//end for

                short int smallm = m - (decm *10);
                for(colindex = 0; colindex < 4; colindex++) {
                    GridArray[rowindex+1][colindex+13]= Glyphs[smallm][rowindex][colindex];
                    //pc.printf("%d",Glyphs[smallm][rowindex][colindex]);
                }//end for
            }//end if
            else {
                for(colindex = 0; colindex < 4; colindex++) {
                    GridArray[rowindex+1][colindex+9]= Glyphs[0][rowindex][colindex];
                    //pc.printf("%d",Glyphs[0][rowindex][colindex]);
                }//end for

                for(colindex = 0; colindex < 4; colindex++) {
                    GridArray[rowindex+1][colindex+13]= Glyphs[m][rowindex][colindex];
                    //pc.printf("%d",Glyphs[m][rowindex][colindex]);
                }//end for
            }//end else
            //pc.printf("\n\r");
        }//end for
    }//end big else

//pc.printf("\033[2;1H");

}//END CharSetter
```

```
273
274
275 void ArrayInit()
276 {
277     pc.baud(115200);         //Debug Port
278     pc.printf("\033[1;1H - GOL Matrix Clock - www.BrainLubeOnline.com\n\rInitializing\r");
279
280     //'0' Character
281     Glyphs[0][0][0] = 1;
282     Glyphs[0][1][0] = 1;
283     Glyphs[0][2][0] = 1;
284     Glyphs[0][3][0] = 1;
285     Glyphs[0][4][0] = 1;
286     Glyphs[0][0][1] = 1;
287     Glyphs[0][1][1] = 0;
288     Glyphs[0][2][1] = 0;
289     Glyphs[0][3][1] = 0;
290     Glyphs[0][4][1] = 1;
291     Glyphs[0][0][2] = 1;
292     Glyphs[0][1][2] = 1;
293     Glyphs[0][2][2] = 1;
294     Glyphs[0][3][2] = 1;
295     Glyphs[0][4][2] = 1;
296     //Space after Char
297     Glyphs[0][0][3] = 0;
298     Glyphs[0][1][3] = 0;
299     Glyphs[0][2][3] = 0;
300     Glyphs[0][3][3] = 0;
301     Glyphs[0][4][3] = 0;
302
303     //'1' Character
304     Glyphs[1][0][0] = 0;
305     Glyphs[1][1][0] = 1;
306     Glyphs[1][2][0] = 0;
307     Glyphs[1][3][0] = 0;
308     Glyphs[1][4][0] = 1;
309     Glyphs[1][0][1] = 1;
310     Glyphs[1][1][1] = 1;
311     Glyphs[1][2][1] = 1;
312     Glyphs[1][3][1] = 1;
313     Glyphs[1][4][1] = 1;
314     Glyphs[1][0][2] = 0;
315     Glyphs[1][1][2] = 0;
316     Glyphs[1][2][2] = 0;
317     Glyphs[1][3][2] = 0;
318     Glyphs[1][4][2] = 1;
319     //Space after Char
320     Glyphs[1][0][3] = 0;
321     Glyphs[1][1][3] = 0;
322     Glyphs[1][2][3] = 0;
323     Glyphs[1][3][3] = 0;
324     Glyphs[1][4][3] = 0;
325
326     //'2' Character
327     Glyphs[2][0][0] = 1;
328     Glyphs[2][1][0] = 0;
329     Glyphs[2][2][0] = 1;
330     Glyphs[2][3][0] = 1;
331     Glyphs[2][4][0] = 1;
332     Glyphs[2][0][1] = 1;
333     Glyphs[2][1][1] = 0;
334     Glyphs[2][2][1] = 1;
335     Glyphs[2][3][1] = 0;
336     Glyphs[2][4][1] = 1;
337     Glyphs[2][0][2] = 1;
338     Glyphs[2][1][2] = 1;
339     Glyphs[2][2][2] = 1;
340     Glyphs[2][3][2] = 0;
341     Glyphs[2][4][2] = 1;
342     //Space after Char
343     Glyphs[2][0][3] = 0;
344     Glyphs[2][1][3] = 0;
345     Glyphs[2][2][3] = 0;
346     Glyphs[2][3][3] = 0;
347     Glyphs[2][4][3] = 0;
348
349     //'3' Character
350
351     Glyphs[3][0][0] = 1;
352     Glyphs[3][1][0] = 0;
353     Glyphs[3][2][0] = 0;
354     Glyphs[3][3][0] = 0;
355     Glyphs[3][4][0] = 1;
356     Glyphs[3][0][1] = 1;
357     Glyphs[3][1][1] = 0;
358     Glyphs[3][2][1] = 1;
359     Glyphs[3][3][1] = 0;
360     Glyphs[3][4][1] = 1;
361     Glyphs[3][0][2] = 1;
362     Glyphs[3][1][2] = 1;
363     Glyphs[3][2][2] = 1;
    Glyphs[3][3][2] = 1;
    Glyphs[3][4][2] = 1;
    //Space after Char
    Glyphs[3][0][3] = 0;
    Glyphs[3][1][3] = 0;
    Glyphs[3][2][3] = 0;
    Glyphs[3][3][3] = 0;
    Glyphs[3][4][3] = 0;

    //'4' Character
    Glyphs[4][0][0] = 1;
    Glyphs[4][1][0] = 1;
    Glyphs[4][2][0] = 1;
    Glyphs[4][3][0] = 0;
```

```
364     Glyphs[4][3][0] = 0;
365     Glyphs[4][4][0] = 0;
366     Glyphs[4][0][1] = 0;
367     Glyphs[4][1][1] = 0;
368     Glyphs[4][2][1] = 1;
369     Glyphs[4][3][1] = 0;
370     Glyphs[4][4][1] = 0;
371     Glyphs[4][0][2] = 1;
372     Glyphs[4][1][2] = 1;
373     Glyphs[4][2][2] = 1;
374     Glyphs[4][3][2] = 1;
375     Glyphs[4][4][2] = 1;
376     //Space after Char
377     Glyphs[4][0][3] = 0;
378     Glyphs[4][1][3] = 0;
379     Glyphs[4][2][3] = 0;
380     Glyphs[4][3][3] = 0;
381     Glyphs[4][4][3] = 0;
382
383     //'5' Character
384     Glyphs[5][0][0] = 1;
385     Glyphs[5][1][0] = 1;
386     Glyphs[5][2][0] = 1;
387     Glyphs[5][3][0] = 0;
388     Glyphs[5][4][0] = 1;
389     Glyphs[5][0][1] = 1;
390     Glyphs[5][1][1] = 0;
391     Glyphs[5][2][1] = 1;
392     Glyphs[5][3][1] = 0;
393     Glyphs[5][4][1] = 1;
394     Glyphs[5][0][2] = 1;
395     Glyphs[5][1][2] = 0;
396     Glyphs[5][2][2] = 1;
397     Glyphs[5][3][2] = 1;
398     Glyphs[5][4][2] = 1;
399     //Space after Char
400     Glyphs[5][0][3] = 0;
401     Glyphs[5][1][3] = 0;
402     Glyphs[5][2][3] = 0;
403
404     Glyphs[5][3][3] = 0;
405     Glyphs[5][4][3] = 0;
406
407     //'6' Character
408     Glyphs[6][0][0] = 1;
409     Glyphs[6][1][0] = 1;
410     Glyphs[6][2][0] = 1;
411     Glyphs[6][3][0] = 1;
412     Glyphs[6][4][0] = 1;
413     Glyphs[6][0][1] = 1;
414     Glyphs[6][1][1] = 0;
415     Glyphs[6][2][1] = 1;
416     Glyphs[6][3][1] = 0;
417     Glyphs[6][4][1] = 1;
418     Glyphs[6][0][2] = 1;
419     Glyphs[6][1][2] = 0;
420     Glyphs[6][2][2] = 1;
421     Glyphs[6][3][2] = 1;
422     Glyphs[6][4][2] = 1;
423     //Space after Char
424     Glyphs[6][0][3] = 0;
425     Glyphs[6][1][3] = 0;
426     Glyphs[6][2][3] = 0;
427     Glyphs[6][3][3] = 0;
428     Glyphs[6][4][3] = 0;
429
430     //'7' Character
431     Glyphs[7][0][0] = 1;
432     Glyphs[7][1][0] = 0;
433     Glyphs[7][2][0] = 0;
434     Glyphs[7][3][0] = 0;
435     Glyphs[7][4][0] = 0;
436     Glyphs[7][0][1] = 1;
437     Glyphs[7][1][1] = 0;
438     Glyphs[7][2][1] = 0;
439     Glyphs[7][3][1] = 0;
440     Glyphs[7][4][1] = 0;
441     Glyphs[7][0][2] = 1;
442     Glyphs[7][1][2] = 1;
443     Glyphs[7][2][2] = 1;
444     Glyphs[7][3][2] = 1;
445     Glyphs[7][4][2] = 1;
446     //Space after Char
447     Glyphs[7][0][3] = 0;
448     Glyphs[7][1][3] = 0;
449     Glyphs[7][2][3] = 0;
450     Glyphs[7][3][3] = 0;
451     Glyphs[7][4][3] = 0;
452
453     //'8' Character
454     Glyphs[8][0][0] = 1;
455     Glyphs[8][1][0] = 1;
456     Glyphs[8][2][0] = 1;
457     Glyphs[8][3][0] = 1;
458     Glyphs[8][4][0] = 1;
459     Glyphs[8][0][1] = 1;
460     Glyphs[8][1][1] = 0;
461     Glyphs[8][2][1] = 1;
462     Glyphs[8][3][1] = 0;
463     Glyphs[8][4][1] = 1;
464     Glyphs[8][0][2] = 1;
465     Glyphs[8][1][2] = 1;
466     Glyphs[8][2][2] = 1;
467     Glyphs[8][3][2] = 1;
468     Glyphs[8][4][2] = 1;
469
```

```
455     //Space after Char
456     Glyphs[8][0][3] = 0;
457     Glyphs[8][1][3] = 0;
458     Glyphs[8][2][3] = 0;
459     Glyphs[8][3][3] = 0;
460     Glyphs[8][4][3] = 0;
461
462     //'9' Character
463     Glyphs[9][0][0] = 1;
464     Glyphs[9][1][0] = 1;
465     Glyphs[9][2][0] = 1;
466     Glyphs[9][3][0] = 0;
467     Glyphs[9][4][0] = 0;
468     Glyphs[9][0][1] = 1;
469     Glyphs[9][1][1] = 0;
470     Glyphs[9][2][1] = 1;
471     Glyphs[9][3][1] = 0;
472     Glyphs[9][4][1] = 0;
473     Glyphs[9][0][2] = 1;
474     Glyphs[9][1][2] = 1;
475     Glyphs[9][2][2] = 1;
476     Glyphs[9][3][2] = 1;
477     Glyphs[9][4][2] = 1;
478     //Space after Char
479     Glyphs[9][0][3] = 0;
480     Glyphs[9][1][3] = 0;
481     Glyphs[9][2][3] = 0;
482     Glyphs[9][3][3] = 0;
483     Glyphs[9][4][3] = 0;
484
485     /* This is a special stylized font set for square-cell led arrays:
486
487         //'0' Character
488         Glyphs[0][0][0] = 0;
489         Glyphs[0][1][0] = 1;
490         Glyphs[0][2][0] = 1;
491         Glyphs[0][3][0] = 1;
492         Glyphs[0][4][0] = 1;
493         Glyphs[0][0][1] = 1;
494         Glyphs[0][1][1] = 0;
495         Glyphs[0][2][1] = 0;
496         Glyphs[0][3][1] = 0;
497         Glyphs[0][4][1] = 1;
498         Glyphs[0][0][2] = 1;
499         Glyphs[0][1][2] = 1;
500         Glyphs[0][2][2] = 1;
501         Glyphs[0][3][2] = 1;
502         Glyphs[0][4][2] = 0;
503         //Space after Char
504         Glyphs[0][0][3] = 0;
505         Glyphs[0][1][3] = 0;
506         Glyphs[0][2][3] = 0;
507         Glyphs[0][3][3] = 0;
508         Glyphs[0][4][3] = 0;
509
510         //'1' Character
511         Glyphs[1][0][0] = 0;
512         Glyphs[1][1][0] = 1;
513         Glyphs[1][2][0] = 0;
514         Glyphs[1][3][0] = 0;
515         Glyphs[1][4][0] = 1;
516         Glyphs[1][0][1] = 1;
517         Glyphs[1][1][1] = 1;
518         Glyphs[1][2][1] = 1;
519         Glyphs[1][3][1] = 1;
520
521         Glyphs[1][4][1] = 1;
522         Glyphs[1][0][2] = 0;
523         Glyphs[1][1][2] = 0;
524         Glyphs[1][2][2] = 0;
525         Glyphs[1][3][2] = 0;
526         Glyphs[1][4][2] = 1;
527         //Space after Char
528         Glyphs[1][0][3] = 0;
529         Glyphs[1][1][3] = 0;
530         Glyphs[1][2][3] = 0;
531         Glyphs[1][3][3] = 0;
532         Glyphs[1][4][3] = 0;
533
534         //'2' Character
535         Glyphs[2][0][0] = 1;
536         Glyphs[2][1][0] = 0;
537         Glyphs[2][2][0] = 0;
538         Glyphs[2][3][0] = 1;
539         Glyphs[2][4][0] = 1;
540         Glyphs[2][0][1] = 1;
541         Glyphs[2][1][1] = 0;
542         Glyphs[2][2][1] = 1;
543         Glyphs[2][3][1] = 0;
544         Glyphs[2][4][1] = 1;
545         Glyphs[2][0][2] = 0;
546         Glyphs[2][1][2] = 1;
547         Glyphs[2][2][2] = 1;
548         Glyphs[2][3][2] = 0;
549         Glyphs[2][4][2] = 1;
550         //Space after Char
551         Glyphs[2][0][3] = 0;
552         Glyphs[2][1][3] = 0;
553         Glyphs[2][2][3] = 0;
554         Glyphs[2][3][3] = 0;
555         Glyphs[2][4][3] = 0;
556
557         //'3' Character
558         Glyphs[3][0][0] = 1;
559         Glyphs[3][1][0] = 0;
560         Glyphs[3][2][0] = 0;
```

```
        Glyphs[3][3][0] = 0;
        Glyphs[3][4][0] = 1;
        Glyphs[3][0][1] = 1;
        Glyphs[3][1][1] = 0;
        Glyphs[3][2][1] = 1;
        Glyphs[3][3][1] = 0;
        Glyphs[3][4][1] = 1;
        Glyphs[3][0][2] = 0;
        Glyphs[3][1][2] = 1;
        Glyphs[3][2][2] = 1;
        Glyphs[3][3][2] = 1;
        Glyphs[3][4][2] = 1;
        //Space after Char
        Glyphs[3][0][3] = 0;
        Glyphs[3][1][3] = 0;
        Glyphs[3][2][3] = 0;
        Glyphs[3][3][3] = 0;
        Glyphs[3][4][3] = 0;

        //'4' Character
        Glyphs[4][0][0] = 1;
        Glyphs[4][1][0] = 1;
        Glyphs[4][2][0] = 1;
        Glyphs[4][3][0] = 0;
        Glyphs[4][4][0] = 0;

        Glyphs[4][0][1] = 0;
        Glyphs[4][1][1] = 0;
        Glyphs[4][2][1] = 1;
        Glyphs[4][3][1] = 0;
        Glyphs[4][4][1] = 0;
        Glyphs[4][0][2] = 0;
        Glyphs[4][1][2] = 1;
        Glyphs[4][2][2] = 1;
        Glyphs[4][3][2] = 1;
        Glyphs[4][4][2] = 1;
        //Space after Char
        Glyphs[4][0][3] = 0;
        Glyphs[4][1][3] = 0;
        Glyphs[4][2][3] = 0;
        Glyphs[4][3][3] = 0;
        Glyphs[4][4][3] = 0;

        //'5' Character
        Glyphs[5][0][0] = 1;
        Glyphs[5][1][0] = 1;
        Glyphs[5][2][0] = 1;
        Glyphs[5][3][0] = 0;
        Glyphs[5][4][0] = 1;
        Glyphs[5][0][1] = 1;
        Glyphs[5][1][1] = 0;
        Glyphs[5][2][1] = 1;
        Glyphs[5][3][1] = 0;
        Glyphs[5][4][1] = 1;
        Glyphs[5][0][2] = 1;
        Glyphs[5][1][2] = 0;
        Glyphs[5][2][2] = 1;
        Glyphs[5][3][2] = 1;
        Glyphs[5][4][2] = 0;
        //Space after Char
        Glyphs[5][0][3] = 0;
        Glyphs[5][1][3] = 0;
        Glyphs[5][2][3] = 0;
        Glyphs[5][3][3] = 0;
        Glyphs[5][4][3] = 0;

        //'6' Character
        Glyphs[6][0][0] = 0;
        Glyphs[6][1][0] = 1;
        Glyphs[6][2][0] = 1;
        Glyphs[6][3][0] = 1;
        Glyphs[6][4][0] = 1;
        Glyphs[6][0][1] = 1;
        Glyphs[6][1][1] = 0;
        Glyphs[6][2][1] = 1;
        Glyphs[6][3][1] = 0;
        Glyphs[6][4][1] = 1;
        Glyphs[6][0][2] = 1;
        Glyphs[6][1][2] = 0;
        Glyphs[6][2][2] = 1;
        Glyphs[6][3][2] = 1;
        Glyphs[6][4][2] = 1;
        //Space after Char
        Glyphs[6][0][3] = 0;
        Glyphs[6][1][3] = 0;
        Glyphs[6][2][3] = 0;
        Glyphs[6][3][3] = 0;
        Glyphs[6][4][3] = 0;

        //'7' Character
        Glyphs[7][0][0] = 1;

        Glyphs[7][1][0] = 0;
        Glyphs[7][2][0] = 0;
        Glyphs[7][3][0] = 0;
        Glyphs[7][4][0] = 1;
        Glyphs[7][0][1] = 1;
        Glyphs[7][1][1] = 0;
        Glyphs[7][2][1] = 0;
        Glyphs[7][3][1] = 1;
        Glyphs[7][4][1] = 0;
        Glyphs[7][0][2] = 1;
        Glyphs[7][1][2] = 1;
        Glyphs[7][2][2] = 1;
        Glyphs[7][3][2] = 0;
        Glyphs[7][4][2] = 0;
```

```
        Glyphs[7][4][2] = 0;
        //Space after Char
        Glyphs[7][0][3] = 0;
        Glyphs[7][1][3] = 0;
        Glyphs[7][2][3] = 0;
        Glyphs[7][3][3] = 0;
        Glyphs[7][4][3] = 0;

        //'8' Character
        Glyphs[8][0][0] = 1;
        Glyphs[8][1][0] = 1;
        Glyphs[8][2][0] = 1;
        Glyphs[8][3][0] = 1;
        Glyphs[8][4][0] = 0;
        Glyphs[8][0][1] = 1;
        Glyphs[8][1][1] = 0;
        Glyphs[8][2][1] = 1;
        Glyphs[8][3][1] = 0;
        Glyphs[8][4][1] = 1;
        Glyphs[8][0][2] = 0;
        Glyphs[8][1][2] = 1;
        Glyphs[8][2][2] = 1;
        Glyphs[8][3][2] = 1;
        Glyphs[8][4][2] = 1;
        //Space after Char
        Glyphs[8][0][3] = 0;
        Glyphs[8][1][3] = 0;
        Glyphs[8][2][3] = 0;
        Glyphs[8][3][3] = 0;
        Glyphs[8][4][3] = 0;

        //'9' Character
        Glyphs[9][0][0] = 1;
        Glyphs[9][1][0] = 1;
        Glyphs[9][2][0] = 1;
        Glyphs[9][3][0] = 0;
        Glyphs[9][4][0] = 0;
        Glyphs[9][0][1] = 1;
        Glyphs[9][1][1] = 0;
        Glyphs[9][2][1] = 1;
        Glyphs[9][3][1] = 0;
        Glyphs[9][4][1] = 1;
        Glyphs[9][0][2] = 0;
        Glyphs[9][1][2] = 1;
        Glyphs[9][2][2] = 1;
        Glyphs[9][3][2] = 1;
        Glyphs[9][4][2] = 0;
        //Space after Char
        Glyphs[9][0][3] = 0;
        Glyphs[9][1][3] = 0;
        Glyphs[9][2][3] = 0;
        Glyphs[9][3][3] = 0;

        Glyphs[9][4][3] = 0;
    */



    for(int index = 0; index < 10; index++) { //Multiplexed Row Pins Initialized to LOW
        RowOut[index] = low;

    }

    for(int index = 0; index < 16; index++) { //Multiplexed Column Pins Initialized to LOW
        ColOut[index] = high;

    }

    for(rowindex = 0; rowindex < 12; rowindex++) { //Initialize Grid Array
        for(colindex = 0; colindex < 18; colindex++) {
            GridArray[rowindex][colindex] = 0;
        }
    }
}//END ArrayInit()



void parser(void)
{
    for(int i = 0; i < 21; i++) {
        for(rowindex = 1; rowindex < 11; rowindex++) { //Parse Matrix for Display Pins
            RowOut[rowindex - 1]= high;              //Multiplex Itteration for 1-Pin Row to HIGH
            for(colindex = 1; colindex < 17; colindex++) {
                ColOut[colindex - 1]= !GridArray[rowindex][colindex];       //Invert results
            }
            wait(dwell);                      //Dwell to be adjusted for Human POV
            RowOut[rowindex - 1]= low;
        }
    }//END big for
}//End Parser()
```

729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763