LED matrix 5X7 Eye Blink (Refresh Screen Mode)

www.hanez.org
http://hanez.org/arduino-ta07-11-5x7-dot-matrix-display.html#Code

CONNECTION: From LED Matrix to Arduino Pins.
1 to 1, 2 to 2, 3 to 3, 4 to 4, 5 to 5, 6 to 6, (one side)
7 to 7, 8 to 8, 9 to 9, 10 to 10, 11 to 11, 12 to 12 (other side)

HARDWARE REQUIRED

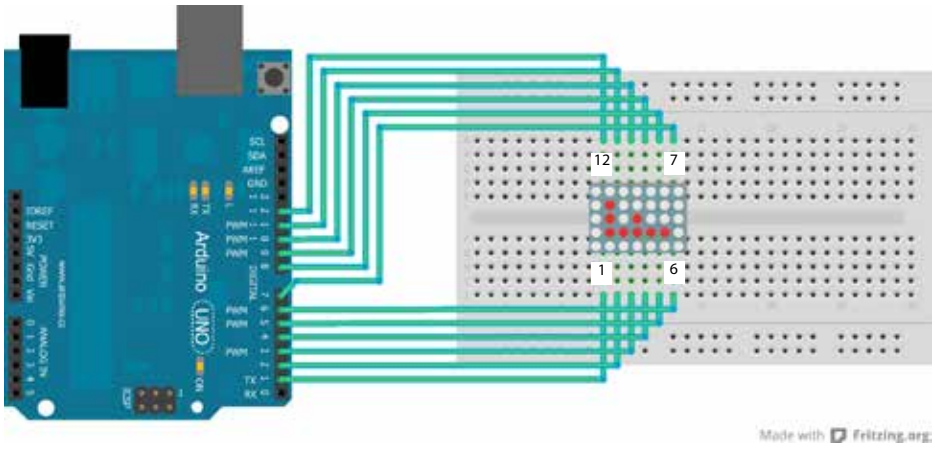
Arduino Board
5X7 LED Matrix (Kingbright TA07-11EWA) Common Anode or Cathode

CIRCUIT

This projects uses the FrequensyTimer2 Library.

IMPORT NOTICE
LED displays are often packaged as matrices of LEDs arranged in rows of common anodes and columns of common cathodes, or the reverse.
FOR COMMON ANODE LED MATRIX SET THE Current varible equal to 1
FOR COMMON CATHODE LED MATRIX SET THE Current varible equal to 0

IMAGE



CODE

```
// FROM 5X7 LED MATRIX GOES TO ARDUINO PINS
// 1>2, 2>9, 3>3, 4>11, 5>12, 6>13, 7>7, 8>8, 9>4, 10>10, 11>6, 12>5

#include <FrequencyTimer2.h>

#define A { \
{1, 0, 0, 0, 0}, \
{1, 1, 1, 1, 0}, \
{1, 0, 0, 0, 1}, \
{1, 0, 0, 0, 1}, \
{1, 0, 0, 0, 1}, \
{1, 1, 1, 1, 0}, \
{1, 0, 0, 0, 0} \
}
```

CODE

```
* Source: http://www.arduino.cc/playground/
Main/DirectDriveLEDMatrix
*
* Modifications for 5x7 LED Matrix element
* by Stefan Wolfrum in July 2012.
* ----------------------------------------
*
* Show messages on an 5x7 led matrix,
* scrolling from right to left.
*
* Uses FrequencyTimer2 library to
* constantly run an interrupt routine
* at a specified frequency. This
* refreshes the display without the
* main loop having to do anything.
*
*/
```

```
#define B { \
{0, 1, 0, 0, 0}, \
{1, 1, 1, 1, 0}, \
{1, 0, 0, 0, 1}, \
{1, 0, 0, 0, 1}, \
{1, 0, 0, 0, 1}, \
{1, 1, 1, 1, 0}, \
{0, 1, 0, 0, 0} \
}

#define C { \
{0, 0, 1, 0, 0}, \
{0, 1, 1, 1, 0}, \
{1, 1, 0, 0, 1}, \
{1, 1, 0, 0, 1}, \
{1, 1, 0, 0, 1}, \
{0, 1, 1, 1, 0}, \
{0, 0, 1, 0, 0} \
}

#define D { \
{0, 0, 1, 0, 0}, \
{0, 1, 1, 1, 0}, \
{0, 1, 0, 0, 1}, \
{0, 1, 0, 0, 1}, \
{0, 1, 0, 0, 1}, \
{0, 1, 1, 1, 0}, \
{0, 0, 1, 0, 0} \
}

#define E { \
{0, 0, 1, 0, 0}, \
{0, 0, 1, 1, 0}, \
{0, 0, 1, 0, 1}, \
{0, 0, 1, 0, 1}, \
{0, 0, 1, 0, 1}, \
{0, 0, 1, 1, 0}, \
{0, 0, 1, 0, 0} \
}

#define F { \
{0, 0, 1, 0, 0}, \
{0, 0, 1, 1, 0}, \
{0, 0, 0, 1, 1}, \
{0, 0, 0, 1, 1}, \
{0, 0, 0, 1, 1}, \
{0, 0, 1, 1, 0}, \
{0, 0, 1, 0, 0} \
}

#define G { \
{0, 0, 1, 0, 0}, \
{0, 0, 0, 1, 0}, \
{0, 0, 0, 0, 1}, \
{0, 0, 0, 0, 1}, \
{0, 0, 0, 0, 1}, \
{0, 0, 0, 1, 0}, \
{0, 0, 1, 0, 0} \
}

#define small_O { \
{0, 0, 1, 0, 0}, \
{0, 0, 1, 0, 0}, \
{0, 0, 1, 1, 1}, \
{0, 0, 1, 1, 1}, \
{0, 0, 1, 1, 1}, \
{0, 0, 1, 0, 0}, \
{0, 0, 1, 0, 0} \
}
```

CODE

```
#define small_W { \
{0, 0, 1, 0, 0}, \
{0, 0, 1, 1, 0}, \
{0, 0, 0, 1, 1}, \
{0, 0, 0, 1, 1}, \
{0, 0, 0, 1, 1}, \
{0, 0, 1, 1, 0}, \
{0, 0, 1, 0, 0} \
}

#define small_R { \
{0, 0, 0, 1, 0}, \
{0, 0, 0, 1, 0}, \
{0, 0, 0, 1, 1}, \
{0, 0, 0, 1, 1}, \
{0, 0, 0, 1, 1}, \
{0, 0, 0, 1, 0}, \
{0, 0, 0, 1, 0} \
}

#define small_D { \
{0, 0, 0, 0, 1}, \
{0, 0, 0, 0, 1}, \
{0, 1, 1, 0, 1}, \
{1, 0, 0, 1, 1}, \
{1, 0, 0, 0, 1}, \
{1, 0, 0, 0, 1}, \
{0, 1, 1, 1, 1} \
}

#define COLS 5
#define ROWS 7
#define PINS 13

// For LED Matrix Common Anode choose Current=0;
// For LED Matrix Common Cathode choose Current=1;
int Current=1;

// pin[xx] on led matrix connected to nn on Arduino (-1 is dummy to make array start at pos 1)
int pins[PINS]= {-1, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12};
// col[xx] of leds = pin yy on led matrix
int cols[COLS] = {pins[1], pins[3], pins[10], pins[7], pins[8]};

// row[xx] of leds = pin yy on led matri
int rows[ROWS] = {pins[12], pins[11], pins[2], pins[9], pins[4], pins[5],
pins[6]};

const int numPatterns = 24;
byte patterns[numPatterns][7][5] = {
  A, B, C, D, E, F,
  G, F, E, D, C, B,
  A, A, A, A, A, A,
  A, A, A, A, A, A
};

int pattern = 0;
boolean normal;

void setup()
{
  //  Switch the current flow in Rows and Columns
    if (Current==1) {
     normal=1;
    } else {
   normal=0;
}
  // sets the pins as output
  for (int i = 1; i <= 12; i++) {
    pinMode(pins[i], OUTPUT);
  }
```

CODE

```
  // set up cols and rows
  for (int i = 1; i <= colNum; i++) {
    digitalWrite(cols[i - 1], normal);
  }

  for (int i = 1; i <= rowNum; i++) {
    digitalWrite(rows[i - 1], normal);
  }

  clearLeds();

  // Turn off toggling of pin 11
  FrequencyTimer2::disable();
  // Set refresh rate (interrupt timeout period)
  FrequencyTimer2::setPeriod(2000);
  // Set interrupt routine to be called
  FrequencyTimer2::setOnOverflow(display);

  setPattern(pattern);
}

void loop()
{
  pattern = ++pattern % numPatterns;
  setPattern(pattern);
  delay(50);
}

void clearLeds()
{
  // Clear display array
  for (int i = 0; i < colNum; i++) {
    for (int j = 0; j < rowNum; j++) {
      leds[i][j] = 0;
    }
  }
}

void setPattern(int pattern)
{
  for (int i = 0; i < colNum; i++) {
    for (int j = 0; j < rowNum; j++) {
      leds[i][j] = patterns[pattern][j][i];
    }
  }
}


// Interrupt routine
void display()
{
  // Turn whole previous column off:
  digitalWrite(cols[col], normal);
  col++;
  if (col == colNum) {
    col = 0;
  }

  for (int row = 0; row < rowNum; row++) {
    if (leds[col][row] == 1) {
      digitalWrite(rows[row], normal);
    }
    else {
      digitalWrite(rows[row], !normal); // reverse the normal
    }
  }
  // Turn whole column on at once (for equal lighting times):
  digitalWrite(cols[col], !normal); // reverse the normal
}
```