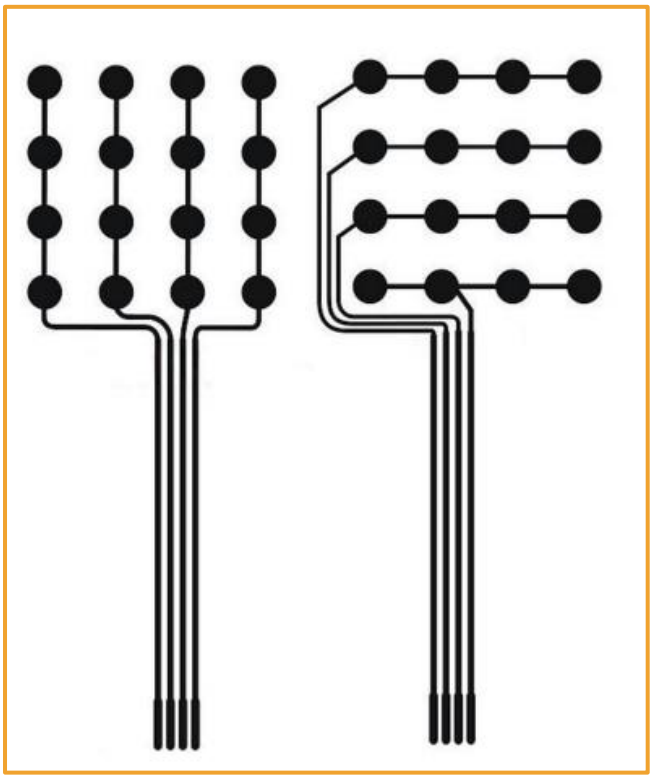
Введение

Матричная клавиатура, использованная в этом эксперименте, содержит в общей сложности 16 клавиш, 4 строки и 4 столбца. Каждый исходный 4 ключа соединены вместе, чтобы сформировать линию, как и каждый столбец, так что в общей сложности 8 строк, то есть 4 строки и 4 столбца.

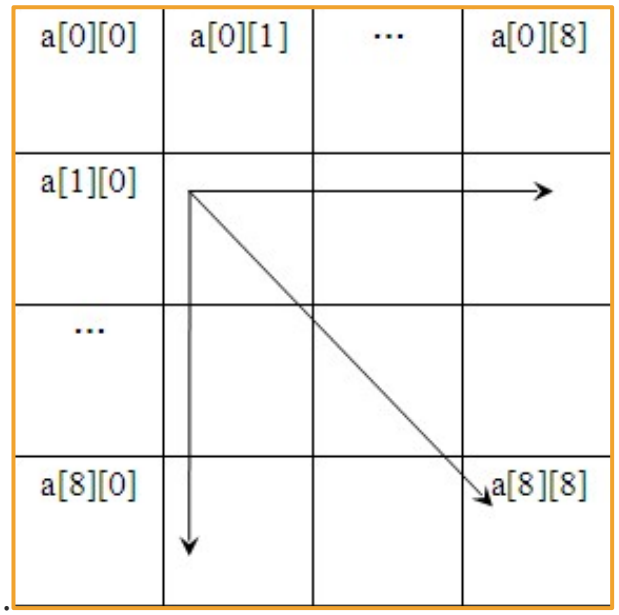
Принципиальная схема матричной клавиатуры



- Рабочий принцип

Сначала мы отправляем низкий уровень столбцам, а оставшиеся столбцы остаются высокими, а затем перебираем каждую строку. Если низкий уровень не обнаружен, это означает, что ни одна клавиша этого столбца не была нажата, а затем мы продолжаем отправлять низкие уровни поочередно оставшимся столбцам и сканировать их; если низкие уровни постоянно обнаруживаются, в строке должна быть клавиша, и Столбец низкого уровня - это столбец, в котором нажата клавиша. После определения строки и столбца подтвердите ключ. Arduino сканирует и обнаруживает достаточно быстро, так что вам не нужно беспокоиться о том, что вы пропустили нажатие клавиш.
В программе мы определяем и используем двумерный массив символов. Массивы предназначены для нумерации простых типов данных: a [0], a [1], a [2], a [3] .... Однако

в некоторых случаях этот тип данных не удобен. Например, мы хотим определить массив для записи результата 9 9 таблицы умножения. Лучший способ записать результаты таблицы умножения 99 - это не определить массив длиной 81, а определить массив из 9 строк и 9 столбцов. Поэтому ссылочный массив имеет два нижних индекса, а именно строки и столбцы. Такой массив называется двумерным массивом и т. Д., Трехмерным массивом и т. Д



Очевидно, что двумерные массивы настолько удобны для записи данных ключевых символов. Например, нам нужно знать данные во второй строке и третьем столбце клавиатуры, если мы ссылаемся на hexaKeys [1] [2] в определенном двумерном массиве hexaKeys. В начале определения двумерного массива мы можем присвоить ему значения. В качестве примера рассмотрим приведенную выше клавиатуру 4 x 4:

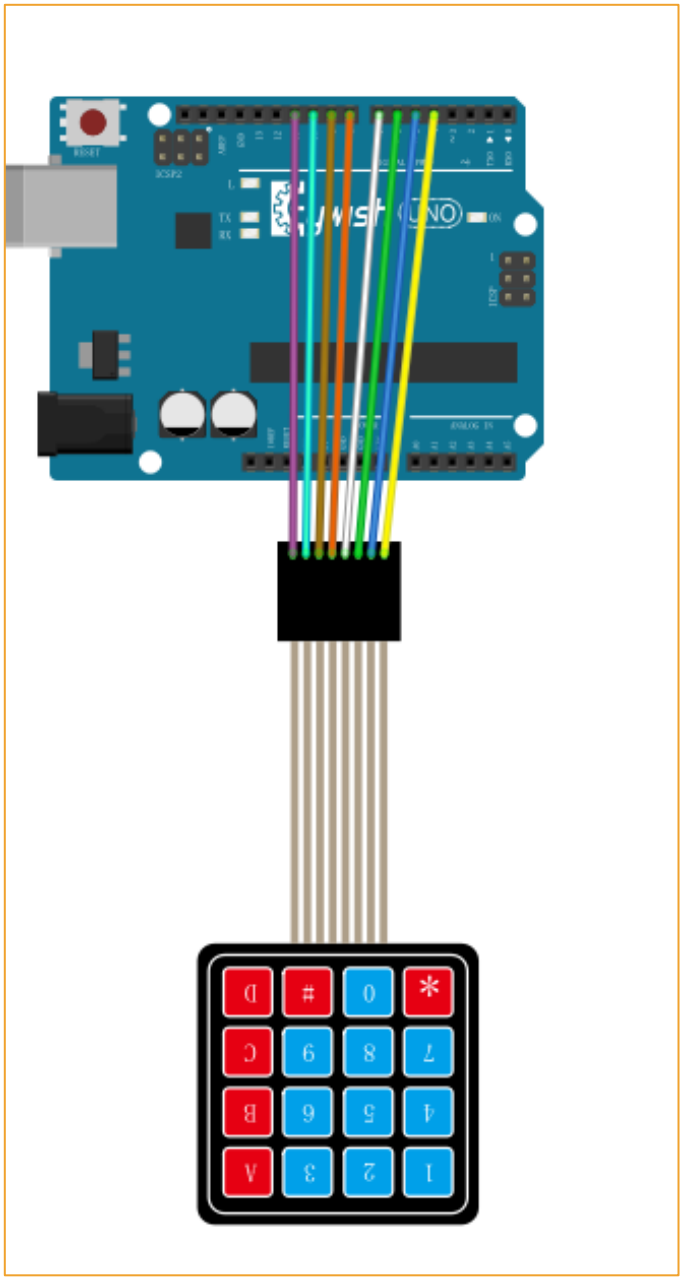
```
char hexaKeys[ROWS][COLS] = {
{'1','2','3','A'},
{'4','5','6','B'},
{'7','8','9','C'},
{'*','0','#','D'}
};
```

Компоненты

- Материнская плата Keywish Arduino UNO R3
- Макетная плата
- USB-кабель для передачи данных
- Матричная клавиатура * 1
- Несколько перемычек

Проводка

arduino Uno	Матричная клавиатура
4	1
5	2
6	3
7	4
8	5
9	6
10	7
11	8

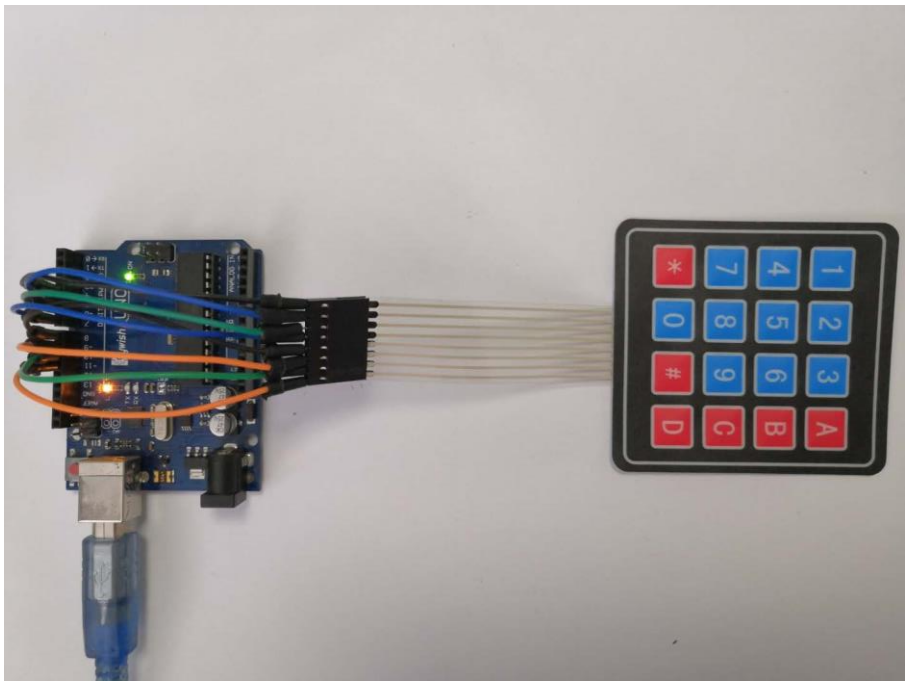

D
#
0
*
C
9
8
7
B
6
5
4
A
3
2
1

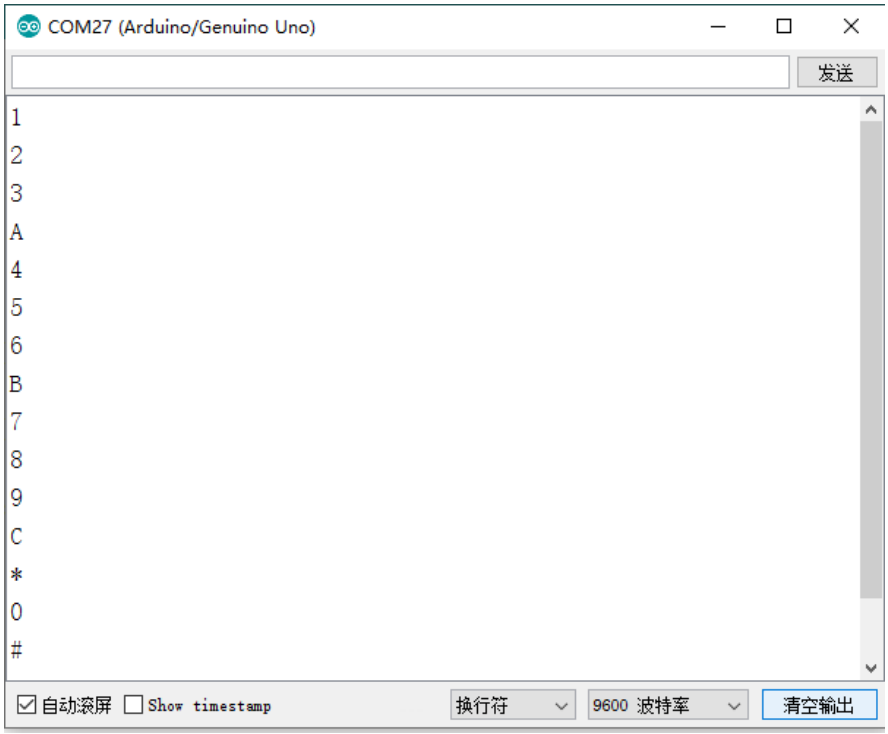
Программа

```
#include "Keypad.h"
#define   ROW_1   4
#define   ROW_2   5
#define   ROW_3   6
#define   ROW_4   7
#define   COL_1   8
#define   COL_2   9
#define   COL_3   10
#define   COL_4   11
const byte ROWS = 4; //四行
const byte COLS = 4; //四列
char hexaKeys[ROWS][COLS] = {
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D'}
};
//连接到键盘的行引出线
byte rowPins[ROWS] = {ROW_1, ROW_2, ROW_3, ROW_4};
//连接到键盘的列引脚
byte colPins[COLS] = {COL_1, COL_2, COL_3, COL_4};
Keypad customKeypad = Keypad( makeKeymap(hexaKeys), rowPins, colPins, ROWS, COLS);
void setup()
{
   int i;
   for(i=0; i< ROWS; i++)
   {
       pinMode(rowPins[i], OUTPUT);  //设置输出模式
       pinMode(colPins[i], OUTPUT);
   }
   Serial.begin(9600); //设置串口波特率为 9600
}

void loop()
{
    char customKey = customKeypad.getKey();
    if (customKey)
    {
       Serial.println(customKey);
    }
}
```

Результаты эксперимента

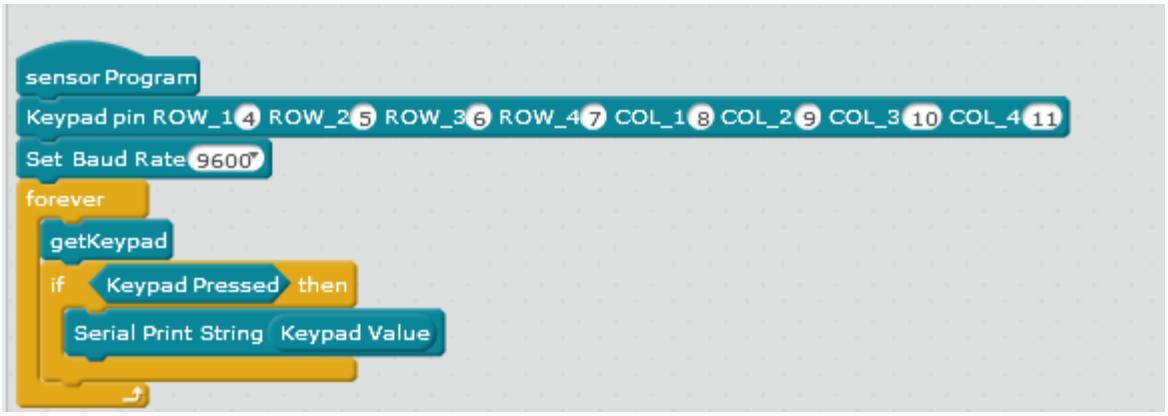




Результатом этого эксперимента является то, что цифры или буквы, которые мы нажимали через кнопки на матричной клавиатуре, будут отображаться на последовательном мониторе.

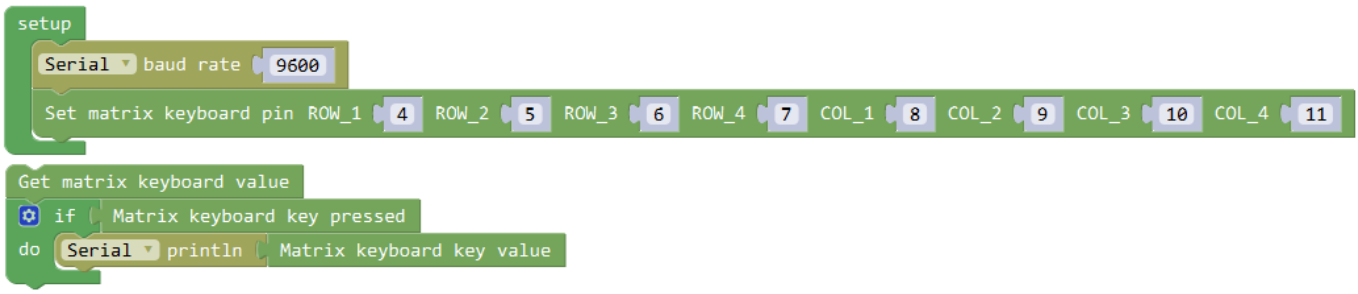
Программа графического программирования mBlock

Программа матричной клавиатуры, написанная mBlock, показана ниже:



Программа графического программирования Mixly

Программа матричной клавиатуры, написанная Mixly , показана ниже:



Программа графического программирования MagicBlock

Программа матричной клавиатуры, написаннаяMagicBlock , показана ниже: