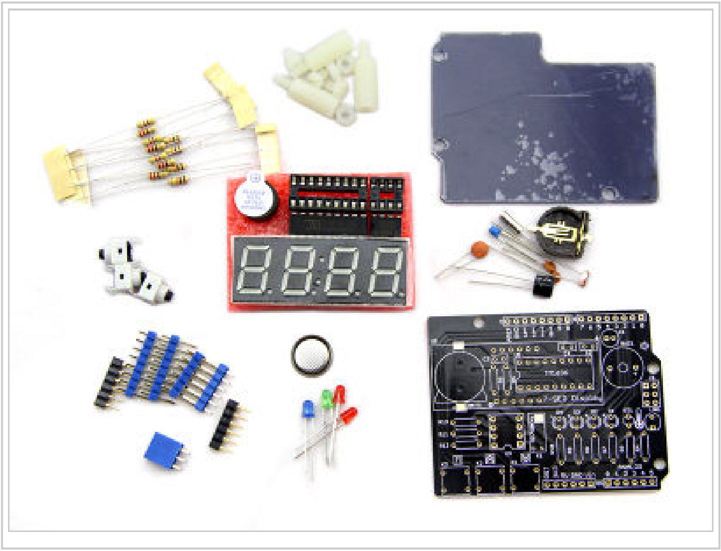
Tick Tock Shield Kit

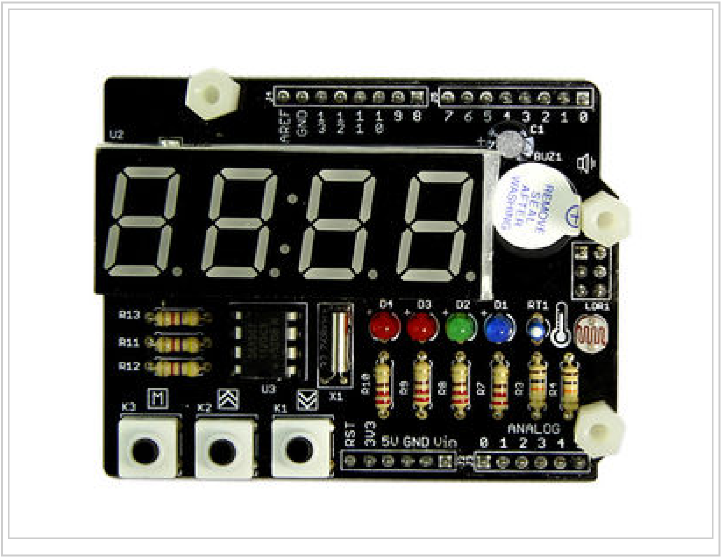
From Wiki 来自痴汉的爱

[中文 (http://www.seeedstudio.com/wiki/index.php?title=Tick_Tock_Shield_Kit_初学者套件&uselang=zh)]

Introduction

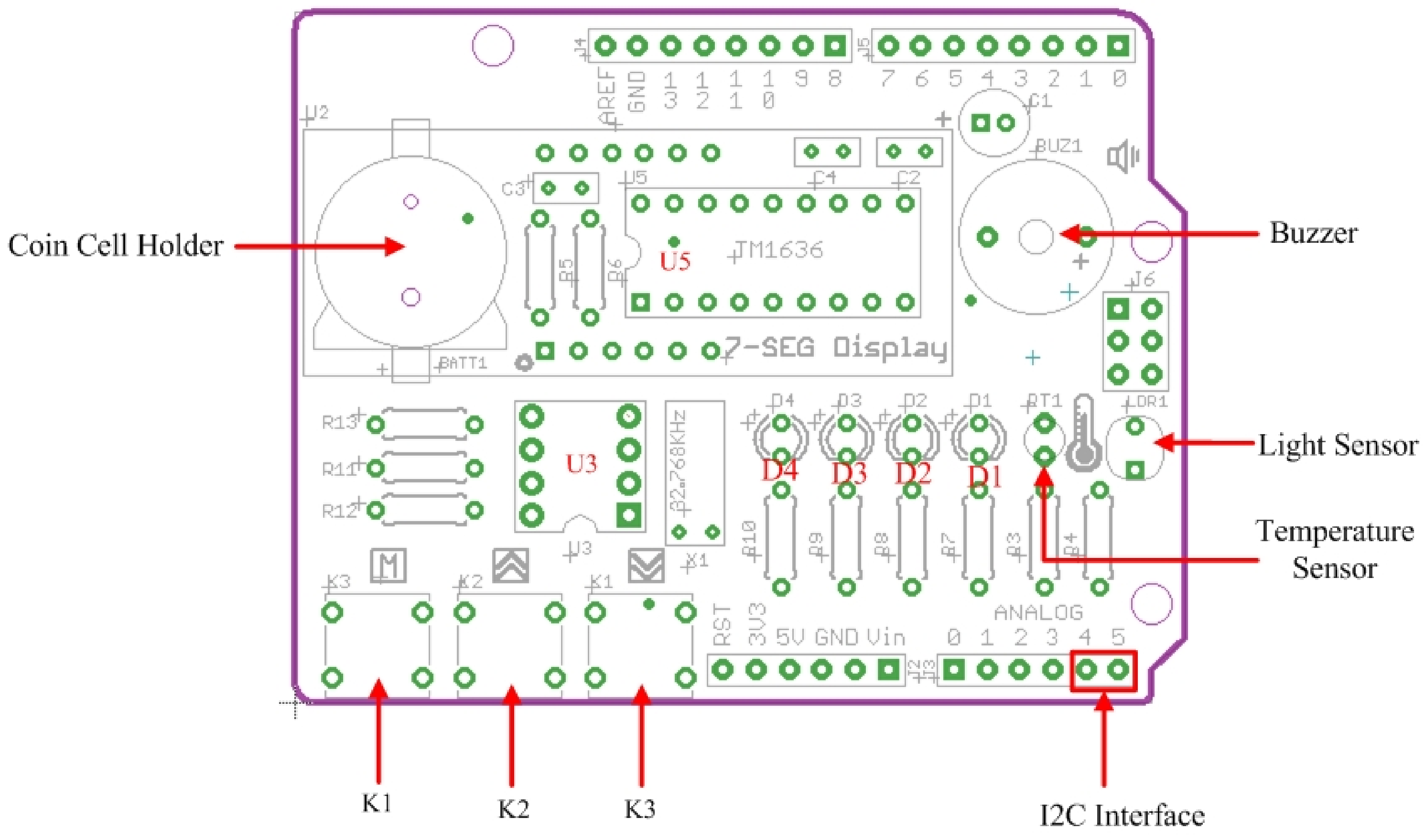


Tick Tock Shield is a perfect start for beginners of Arduino world. You can learn and practice basic soldering tricks and programming principles when tinkering with this kit. Electronic could be fun even if you have bare technical knowledge cause we get your back by preparing a detailed soldering guide and a fully supported library of programming examples from easy to difficult. Hope you have fun and gain some knowledge with this kit!



Tick Tock Shield contains most common resources for a basic electronic project, like buttons, sensors, buzzer and display. After you're done with the soldering task, it turns out to be a geek style alarm clock which can auto adjust the brightness of display and keep in synch with real world time.
Model: SLD90400P (http://www.seeedstudio.com/depot/tick-tock-shield-p-1371.html?cPath=6_7)

Specification



Hardware Resources

- Coin Cell Holder: provides power to RTC IC when external power is off;
- Buzzer: create audio effect;
- Light Sensor: detect ambient light intensity;
- Temperature Sensor: detect ambient temperature;
- K1...K3: temporary buttons;
- D1...D4: LEDs in blue, green, red and red;
- U3: DS1307, Real Time Clock IC;
- U5: TM1636, 7-seg display driver IC.

Pins Used On Arduino

- D2: control LED1;
- D3: control LED2;
- D4: control LED3;
- D5: control LED4;
- D6: control buzzer;
- D7: TM1636 SCLK pin;

- D8: TM1636 DIO pin;
- D9: control K1;
- D10: control K2;
- D11: control K3;
- A0(D14): poll readings from temperature sensor;
- A1(D15): poll readings from light sensor;
- A4(D18): DS1307 SDA pin
- A5(D19): DS1307 SCK pin

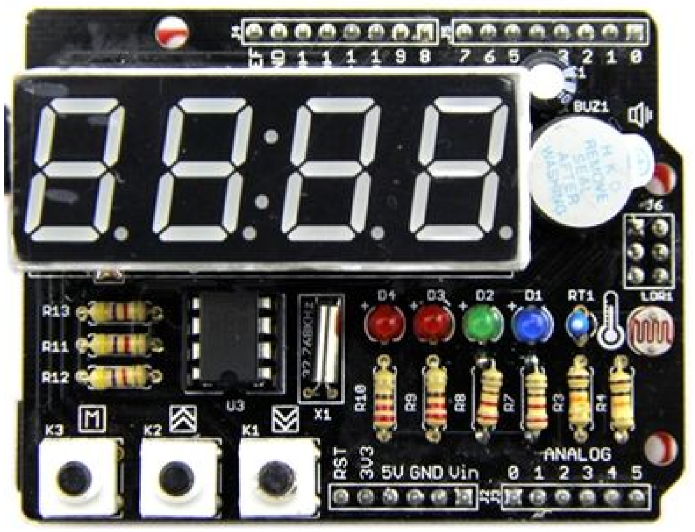
Demonstration

You are supposed to finish the soldering of your Tick Tock Shield before moving on to following demos. If you didn't, click here (http://www.seeedstudio.com/document/pdf/Tick%20Tock%20Shield%20Soldering%20Guide.pdf) to download the soldering guide. Here we start to present you 7 demos from easy to difficult. They involve usage of all hardware resources on Tick Tock Shield:

- Demo 1: Sweep The LEDs
- Demo 2: Control LEDs By Keys
- Demo 3: Change The Pitch Of Buzzer By Keys
- Demo 4: Display Temperature
- Demo 5: Make A Light Sensor Controlled LED
- Demo 6: Display Numerical Characters
- Demo 7: Real Time Clock

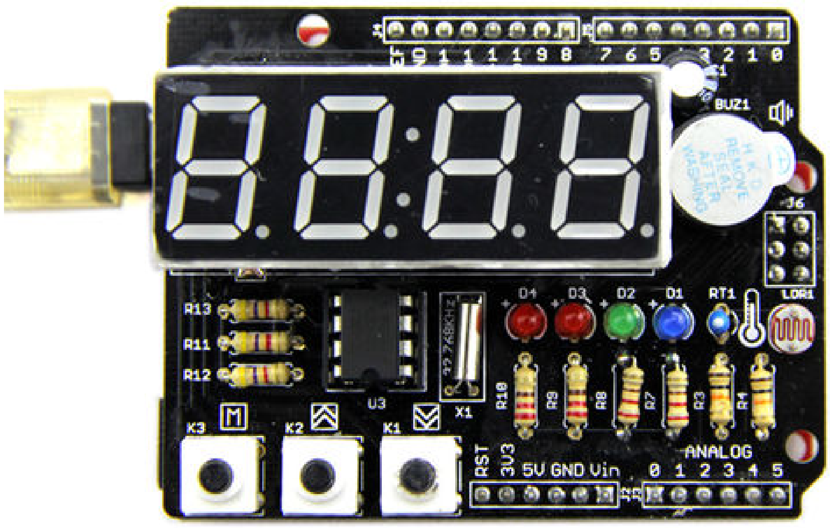
Each demo caontains detailed comments in the .ino file. To figure out how every line affects the turnout, you have to study the sketch and comments carefully.

OK, let's check out what the Tick Tock Shield is capable of.



Preparation: Install the Hardware and Software

1. Plug Tick Tock Shield onto Arduino board. Connect Arduino to PC via USB cable as show below.



2. Download the file： Tick Tock Shield Library (http://www.seeedstudio.com/wiki/File:Tick_Tock_Shield_libraries.zip)
3. Unzip and put them in the libraries file of Arduino IDE by the path: ..\arduino-1.0.1\libraries.

Note:

1) Libraries MsTimer2 and Timerone are packed in Tick Tock Shield Library we prvide above. We download them from Arduino website. If you already have them in your libraries file, them no need to add them once again.
2) Through all 7 demos in Tick Tock Shield library, we define:

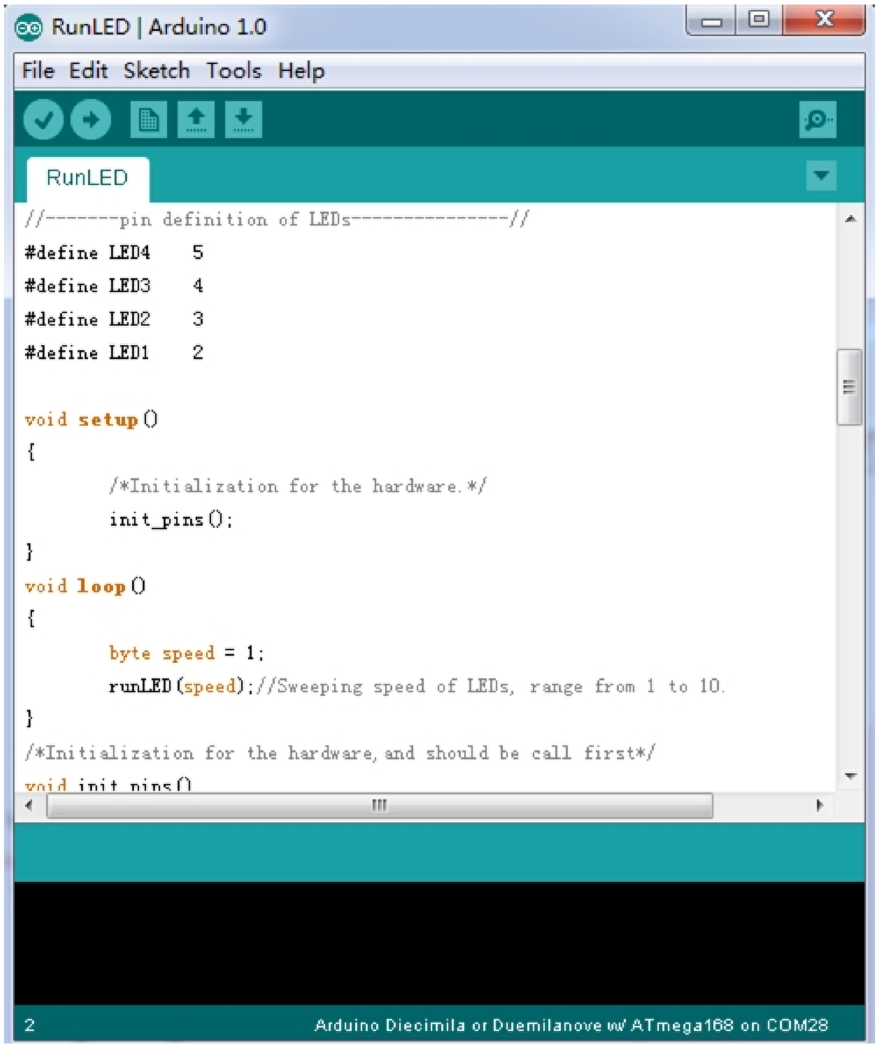
K3	- menu key
K2	- increase key
K1	- decrease key
D4	- LED_CLOCK_ADJUST
D3	- LED_ALARM_ADJUST
D2	- LED_ALARM_ENABLE
D1	- LED_BRIGHT_ADJUST

Getting Started: Fun With Tick Tock Shield

From easy to difficult, we prepared a series of demos for you to explore what your Tick Tock Shield is capable of. Follow me to have fun with it.

Demo 1: Sweep The LEDs

1. This demo only involves the usage of the most basic actuator - LEDs.
2. Restart the Arduino IDE. Open the example "RunLED" via the path: File --> Examples --> TickTockShield--> RunLED.



```
//-------pin definition of LEDs---------------//
#define LED4    5
#define LED3    4
#define LED2    3
#define LED1    2

void setup()
{
        /*Initialization for the hardware.*/
        init_pins();
}
void loop()
{
        byte speed = 1;
        runLED(speed);//Sweeping speed of LEDs, range from 1 to 10.
}
/*Initialization for the hardware, and should be call first*/
void init_pins()
```

```
2                    Arduino Diecimila or Duemilanove w/ ATmega168 on COM28
```

3. This demo can sweep 4 LEDs with a settable speed. You can change the sweep speed by changing the parameter of the function "runLED(speed)". Find further illutration of the funtion in its comment.
4. Click the upload button to upload the program to the Arduino.
5. You can see four LED lights turn on and off from left to right at a given speed after the program is uploaded.

Demo 2: Control LEDs By Keys

This demo show you how to control LEDs with the most basic input - buttons.

1. Open the example "ControlLED" in the same way as open the “RunLED” above.
2. This demo can turn on or off four LEDs by keys. We set two flags to store the status of this test, TEST_START and TEST_END. Every time the “menu” key gets pressed, the status toggles between TEST_START and TEST_END. In the status of TEST_START, "increase" key can turn on one more LED from right to left every time it gets pressed. The "decrease" key has an inverse effect as the "increase" key.

Demo 3: Change The Pitch Of Buzzer By Keys

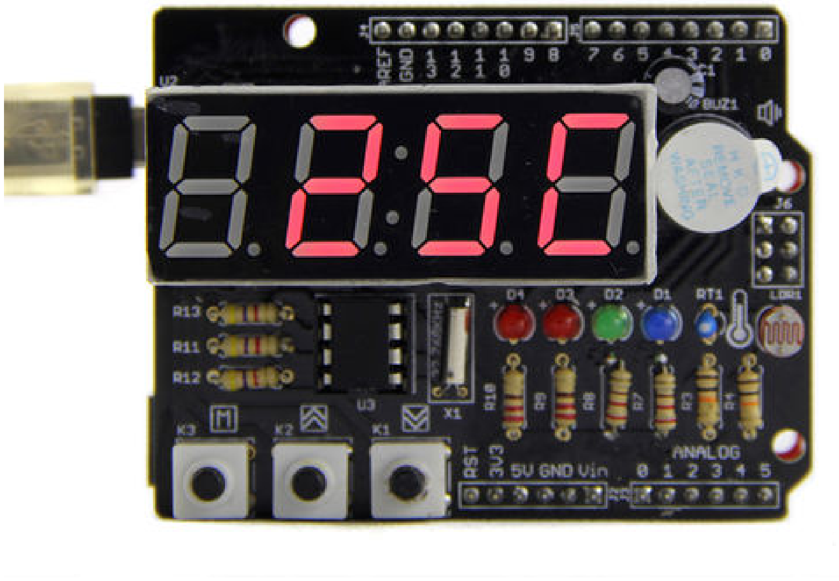
This demo involves how to make a sound with buzzer, and because you have already known how to use buttons, in this demo, we use buttons to increase or decrease the pitch of buzzer.

1. Open the example "changeThePitch".
2. Every time the “increase” key gets pressed, the pitch of the buzzer will rise up. And when the “decrease” key gets pressed, the pitch will fall down.

Demo 4: Display Temperature

Get started with sensors. First we have here is the temperature sensor. Try to read its value and display it on the 7 segment display.

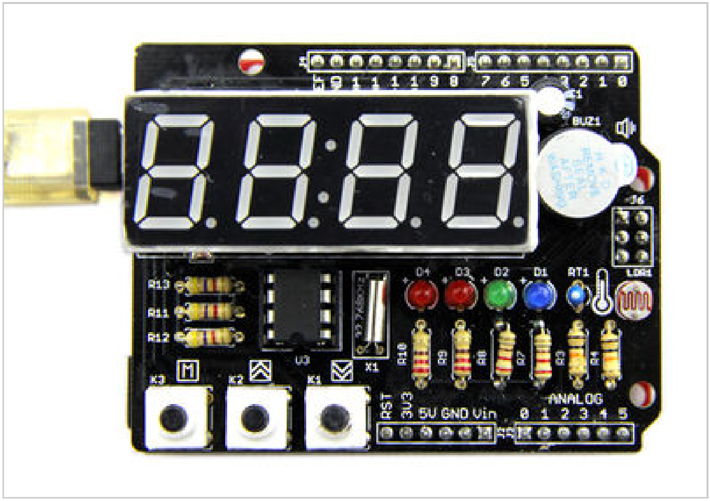
1. Open the example "MeasureTemprature".
2. The 7-segment Display will display current temperature read from temperature sensor as shown below.
3. If you find in the picture below, the contrast is not strong enough, try a lower view angle. This principle also apply to following demos which contain usage of 7-segment display.

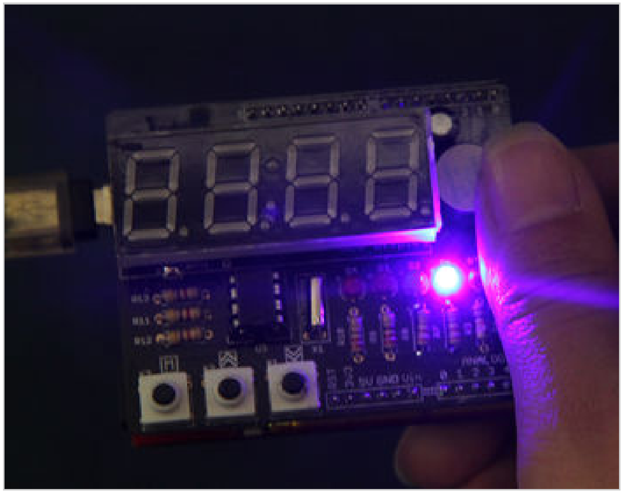


Demo 5: Make A Light Sensor Controlled LED

Is it convenient if the brightness of display can auto adjust itself according to the ambient light? This demo shows you how to do this by using a light sensor.

1. Open the example "SensorControlBrightness" in the same way.
2. This example can change the brightness of BRIGHT_ADJUST Indicator according to the ambient light intensity. The darker the environment is, the lighter the LED turns. Picture on the right is the turnout.

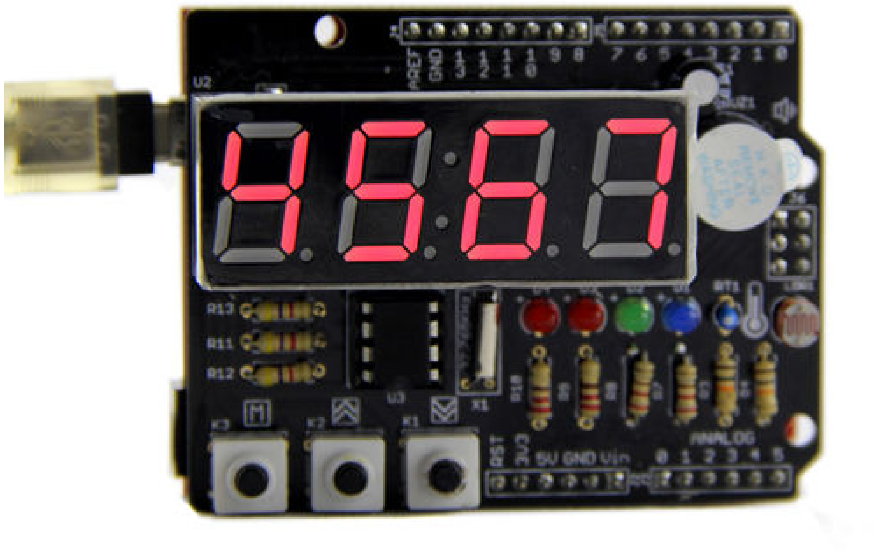

Connect Tick Shield.jpg


Control Light.jpg

Demo 6: Display Numerical Characters

This demo shows you how to control the content of 7 segment display.

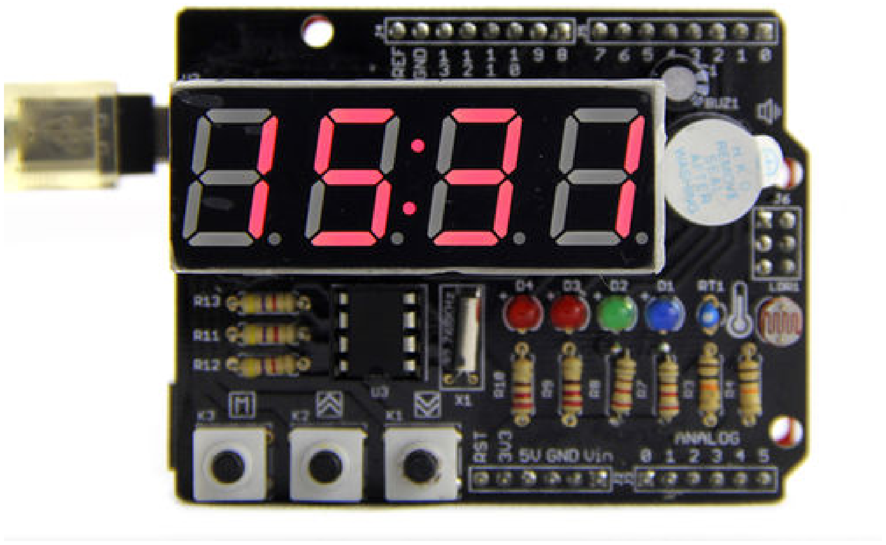
1. Open the example “CharacterFlow”.
2. This example can run numeric characters 0, 1, 2, 3, 4, 5, 6, 7, 8 ,9, A, b, C, d, E, F from right to left.



Demo 7: Real Time Clock

This is a final demo that involves all hardware resources we mentioned above and performs all functions of a real life alarm clock, like time display, making a buzz to get you out of your bed and so on. What's more, because a light sensor and a temperature sensor are added, Tick Tock Shield does more than a normal alarm clock. It can sense the ambient light and auto-adjust the brightness of 7-segment display. Also it can show you the current temperature at a given time interval.

1. Open the example "RealTimeClock" and upload the example into Arduino.
2. The 7-segment Display will alternately display current time and temperature.



3. Method to adjust the time and set the alarm clock:

   1) Because there is a RTC chip on board, so you don't need to reset the time every time you turn it on, of course, the condition is the coin battery for RTC chip is alive. But if this is your first time to start up the Tick Tock Shield, time setting is required.

   2) Three buttons are used to adjust the time. They are "menu"(K3), "increase"(K2) and "decrease"(K1). Press "memu" to cycle between the next setting modes:

      a) set the hour part of normal time display
      b) set the minute part of normal time display

c) set the hour part of alarm clock
d) set the minute part of alarm clock
e) set the default brightness of 7 segment display
f) turn on or off the alarm function
g) quit time setting modes

3) If you want to quit before processing all 7 steps above. There are two interruption ways. One, press K3 no matter on which step for 3s. In this way,Tick Tock Shield will confirm all settings you have made and quit. Two, leave the Tick Tock Shield alone for 5s. In this way. No setting will be stored and you quit time setting mode also.

4) K2 is assigned to set the status of alarm clock. You can know the status of alarm clock via ALARM_ENABLE Indicator.

Resources

Tick Shield Library (http://www.seeedstudio.com/wiki/File:Tick_Shield_libraries.zip)
Tick Shield eagle file (http://www.seeedstudio.com/wiki/File:Tick_Shield_eagle_file.zip)
Tick_Shield_Schematic (http://www.seeedstudio.com/wiki/File:Tick_Shield_Schematic.pdf)
Tm1636 datasheet (http://www.seeedstudio.com/wiki/File:Tm1636.pdf)
DS1307N datasheet (http://www.seeedstudio.com/wiki/File:DS1307N.pdf)

Reference

EEPROM Class

Class Function: Read and write EEPROM of AVR chip. The EEPROM size of ATmega328P Chip is 1K Bytes for Seeeduino V3.0. The contents of the EEPROM you have wrote will not be lost when power off. It is used for saving alarm time and alarm enable flag in the RealTimeClock Demo, You don't need to reset the alarm clock.

Function Description:
1. EEPROM.read(int address);
This function is used to read data from a specified address of EEPROM.

- address: the address of the targetted unit.

Example：

```
temp_data[i] = EEPROM.read(i);
```

2. EEPROM.write(int address, uint8_t value);
The function is used to write data to a specified address of EEPROM.

- address: the address of the tergetted unit
- uint8_t value: data to be written to the targetted unit.

Example:

```
if(temp_data[i] != mark[i])
    {
      EEPROM.write(0,mark[0]);
      EEPROM.write(1,mark[1]);
      EEPROM.write(2,mark[2]);
      EEPROM.write(3,mark[3]);
      EEPROM.write(4,mark[4]);
      return true;
    }
```

TM1636 class

Class Function: this class contains all funtions to control four 7-segment displays.
Function Description:
1. tm1636.point(boolean PointFlag);
The function is used to turn on or off the clock point (:). This function will come into effect every time the display content changes.

- PointFlag: can be 0(off) or 1(on).

Example:

```
if(flag_clockpoint)
      {
        tm1636.point(POINT_ON);
      }
      else tm1636.point(POINT_OFF);
```

2. Tm1636.display(int8_t DispData[]);
Create a character flow with the content of DispData[].

- DispData[]: an array in int8_t type, including 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 0, A, b, C, d, E and F.

Example:

```
tm1636.display(disp);
```

TickTockShield Class

Class Function: operate all components o
Function Description:
1. ticktockshield.init();
The Initialization of Tick Tock Shield.
2. ticktockshield.runLED(byte speed, byte d

This function sweep 4 LEDs in the direction

- speed: the the LED flow rate from 1(
- direction: LED flow direction, can be

Example:

Contents

```
/*Run the 4 LEDs from left to right*/
ticktockshield.runLED(1,LEFT_TO_RIGHT);
```

3. ticktockshield. turnOnLED();
Turn on all 4 LEDs.
4. ticktockshield. turnOffLED();
Turn off all 4 LEDs.
5. setLed(unsigned char led_status, int pinLED);
The Function is used to turn on or off an individual LED.

- led_status: can be LED_ON or LED_OFF.
- pinLED: can be one of LED_CLOCK_ADJUST，LED_ALARM_ADJUST，LED_BRIGHT_ADJUST，LED_ALARM_ENABLE.

↑TOP

Example:

```
void TickTockShield::turnOffLED()
{
        setLed(LOW,LED_CLOCK_ADJUST);
        setLed(LOW,LED_ALARM_ADJUST);
        setLed(LOW,LED_BRIGHT_ADJUST);
        setLed(LOW,LED_ALARM_ENABLE);
}
```

6. ticktockshield.alarming();
Let the buzzer buzz.
Example:

```
/*It will sound alarm for a minute untill the "MENU" key is pressed*/
    if(ticktockshield.isAlarmEnable())
    {
      ticktockshield.alarming();
    }
```

7. ticktockshield.turnOffAlarm();
Turn off the alarm when the buzzer is buzzing.
8. ticktockshield.setAlarm(uint8_t hour,uint8_t minute,uint8_t flag_enabl);
Set the alarm time.

- hour: hour
- minute: minute
- flag_enabl: flag that restores the status of alarm, can be 0(unenabled) or 1(enabled)

Example:

```
ticktockshield.setAlarm(12,0);//Yes,the alarm clock is initialized to 12:00 and the data in the EEPROM.
```

9. ticktockshield.getAlarm();
Read the preset alarm value from EEPROM and store them into given variables, including the enable flag of the alarm.
Example:

```
if(isFirstLoad())//if it is the first time to load the firmware?
        {
                ticktockshield.setAlarm(12,0);
        }
        else ticktockshield.getAlarm();//No,read the alarm clock stored in EEPROM
```

10. ticktockshield.getTemperature();
Get the reading from temperature sensor.
Example:

```
/*Read the ambient temperature and display on the digital tube.*/
        ticktockshield.displayTemperature(ticktockshield.getTemperature());
```

11. ticktockshield.displayTemperature(int8_t temperature);
Display temperature value(negative value supported) on the 7 segment display. The character C represents celsius degrees.
Example:

```
 ticktockshield.displayTemperature(ticktockshield.getTemperature());
```

12. ticktockshield.scanKey();
Find out which key gets pressed. Return the pin number of the key pressed. And return "-1" if no keys is pressed.

```
if((flag_scan_again)&&(KEY_MENU == ticktockshield.scanKey()))
      {
        ticktockshield.writeToAdjustArea();
        ticktockshield.processKey();
        system_states = SYSTEM_ADJUSTING;
      }
```

13. ticktockshield.processKey();
Process the command you enter through keys. If it's a normal press. This function will make a buzz corresponding to the press. If you press the "menu" key longer than 3s, then this function will make Tick Tock Shield enter time setting mode. If no other key gets pressed after the "menu" key, then this function will make Tick Tock Shield quit the time setting mode.

14. ticktockshield.processSystemStatus();
Execute different tasks according to the system status when it's called. The system status can be "adjust clock time" 、 "alarm time", "adjust 7 segment display brightness" and "enable alarm".

15. ticktockshield.writeToAdjustArea();
Deliver the latest time information from normal time display mode to time setting mode so that they can be used under time setting mode.
16. ticktockshield.writeToNormalArea();
Deliver the time information set in time setting mode to normal time display mode.
17. ticktockshield.writeTime();
Write time information into RTC chip.
18. ticktockshield.getTime();
Read the current time information from RTC.
19. ticktockshield.displayTime();
Display time on 7 segment Display.

Example:

```
 if(ticktockshield.isAlarmEnable())
      {
        tm1636.point(POINT_ON);
        ticktockshield.displayTime();
        system_states = SYSTEM_ALARMING;
        return;
      }
```

20. ticktockshield.display(int8_t DispData []);
Display alpha-numeric information 7 segment Display.

Retrieved from "http://www.seeedstudio.com/wiki/index.php?title=Tick_Tock_Shield_Kit&oldid=40173"
