Some documentation for RGB LED strip controller

This LED has RGB LEDs on them which are not individually addressable like LED strips that use a WS2812. Instead all LEDs are connected together and are activated by pulling the Cathode of each LED to ground. The LED strip is setup using segments of 20 cm containing 6 RGB LEDs including current limiting resistors and operates at 24 Volt.

For this controller I am using a Remote Control that uses the NEC Remote Control format. This Remote Control has buttons on it that represent a color as shown in the picture. I do not always want to use the RGB LED strip with a Remote Control so when powering it up, the controller generates a random color pattern. As soon as a Remote Control is used, it uses the predefined colors of the Remote Control. Note that I did not implement features like Flash, Strobe, Smooth, etc. There is sufficient space left in the Flash memory of the PIC Microcontroller to do so.

The controller is using a PIC12F617 Microcontroller programmed with the JAL programming language.

Supplies

You need to have the following components for this project:

- A breadboard
- 1 * PIC Microcontroller 12F617
- 1 * 5V Voltage Regulator 78L05
- 1 * Infra Red Receiver TSOP4838
- 3 * Transistors BC547
- Capacitors: 2 * 100 nF, 1 * 470 nF
- Resistors: 3 * 4k7
- 1 * LED strip
- 1 * Power Jack

The hardware design

See the schematic diagram. The heart is formed by an 8-pin PIC12F617 Microcontroller but any other PIC Microcontroller can be used as long as it has at least 1.5 kByte of Flash and at least 90 bytes of RAM. An external power source of at most 24 Volt is needed where the 78L05 creates the 5 Volt supply Voltage for the PIC. The LEDs in the strip are activated by pulling the Cathode of each LED to ground. This is done using transistors. In this design the BC547 is used which can handle up to 100 mA which was sufficient for the length of the RGB strip I was using but when the strip becomes longer you may need transistors that can handle more current.

For my project I needed a length of 100 cm, using 5 segments. When all LEDs where turned on to White, a total of 94 mA was drawn from the 24 Volt supply Voltage. When switching the RGB LEDs on individually, the current is about 37 mA in all cases. When the device is switched off, about 6 mA of current is drawn.

Note that in this schematic diagram I mentioned the SMLVN6RGB as RGB LED but I do not know which LEDs where exactly on the LED strip. I just used this type of LED since they seem to match with the RGB LEDs on the LED strip.

Building the Electronics

You can build the circuit on a breadboard, which I did. Since the power supply can be from any source I used a power jack as input. The LED strip is connected by soldering a male header pin on the strip itself and connecting it to a female header on the breadboard.

The Software

The software is written for a PIC12F617 using the JAL programming language but as mentioned earlier you can also use any other PIC as long as it has sufficient Flash and RAM. The software uses 1308 bytes of Flash and 97 bytes of RAM. This PIC can handle 2 kByte of Flash and 128 bytes of RAM so there is still space to add some extra features to it.

In order to control the brightness of the LEDs, Pulse Width Modulation (PWM) is used. Since three independent PWM signals need to be generated, this generation is done in software. When only one PWM signal was needed, you can the PWM hardware of this Microcontroller.

When looking at the JAL source code (see attached listing), you find the following set-up:

- PIC Microcontroller configuration (Fuses)
- Pin definitions
- Include files
  - Device file of the PIC12F617
  - NEC Remote Control decoder library
- Constants and variables
- Regular functions and procedures
- Interrupt procedure
- Main program (forever loop)

In order to generate the PWM signal, Timer 2 of the PIC Microcontroller is used. This is done on an interrupt basis running at a frequency of 8772 Hz. This high frequency is needed to make sure that the refresh rates of the LED is high enough. With this frequency the refresh rate is around 70 Hz which is invisible for the eye.

For decoding the Infra Red message from the NEC Remote Control, I am using a JAL library. The decoding is done on an interrupt basis, using Timer 1 of the PIC Microcontroller.

The main program consists of a forever loop and has two modes:

1. When powered up, it creates a random fade-in fade-out pattern, which is a variant of the Illuminated Cubes project I posted several years ago on Instructables. https://www.instructables.com/Illuminated-Cubes/

2. When the Remote Control is used, the LEDs are set to the predefined colors as given on the Remote control.