Application Note Template

Matthew Rainey
Rowan University

October 21, 2018

1 Design Overview

This application note demonstrates the operation of an RGB node using an ATMega328p-pu microcontroller. The nodes primary function is to receive a packet from another node or a computer, set the RGB LED PWM values, calculate the new packet, and transmit it to the next node in series. This application note explains what this node does, how it operates, and how to interface with this node.

1.1 Design Features

- standard 9600b operation
- sleep mode utilization for lowest power consumption possible
- low i/o clock rate and usage of interrupts to minimize power consumption
- maximum of 16 nodes can be utilized

1.2 Featured Applications

A few featured applications of this RGB node are:

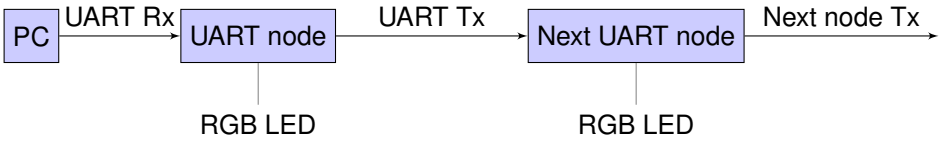
- Christmas lights
- Display lights
- Pranks on coworkers

1.3 Design Resources

This project is open-source and available on github.
The project can be found here: Project link.

1.4 Block Diagram

Here is a basic block diagram of the node setup



1.5 Board Image

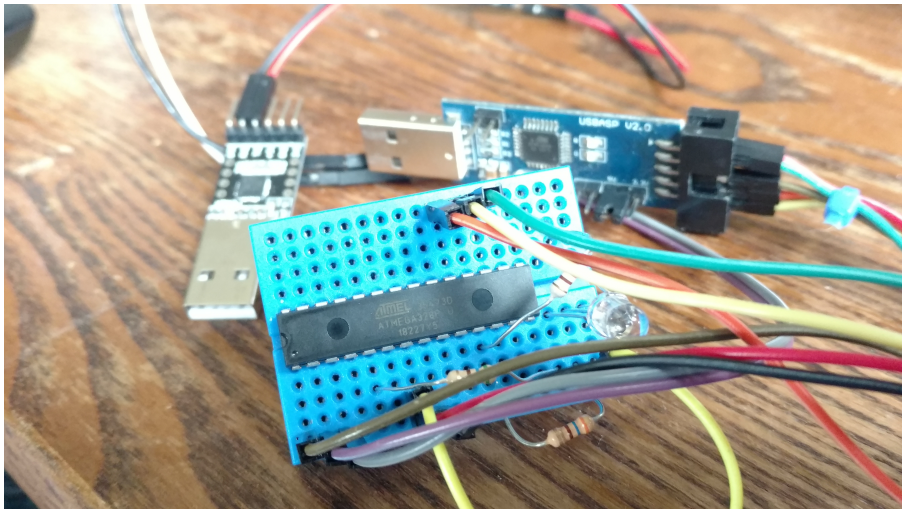


Figure 1: Photo of breadboard, programmer and USB - UART converter

Below is a photo of the latest implementation of the RGB node. In later versions, a protoboard design may be used.

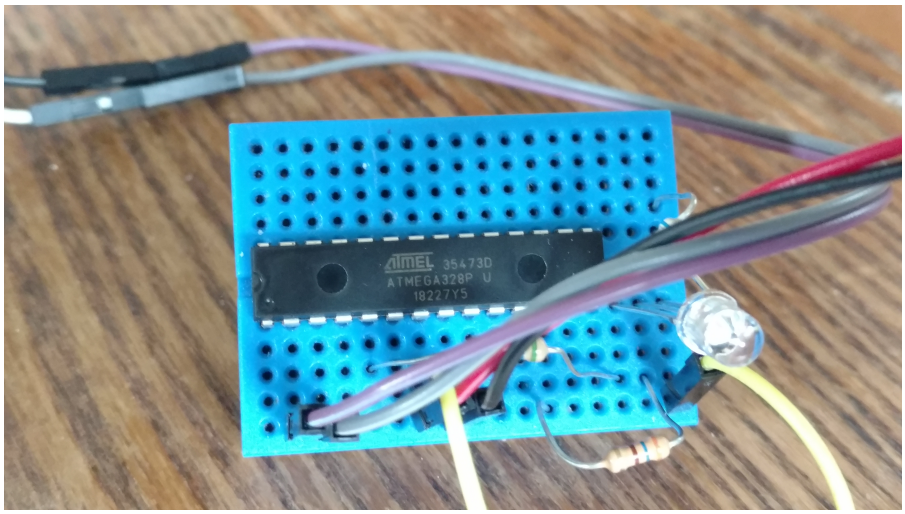


Figure 2: Photo of breadboard and USB - UART converter

2 Key System Specifications

This UART node operates off a 1 MHz oscillator sourced from the internal 8 MHz RC oscillator divided by 8. The UART transceiver uses 8 bits, no parity, and 1 stop bit.

Microcontroller	AtMEGA328p
Baud Rate	9600 baud 8N1
voltage	3.3v
Current consumption @ 3.3v	170 uA
Rx - Tx delay	40 μs
Max node design	16

3 System Description

This project was created for demonstrating basic UART usage and timer PWM usage on an ATMEGA328p-pu microcontroller.

3.1 Detailed Block Diagram

Below is a more detailed image of the basic structure of the microcontroller and what it's interfacing with.

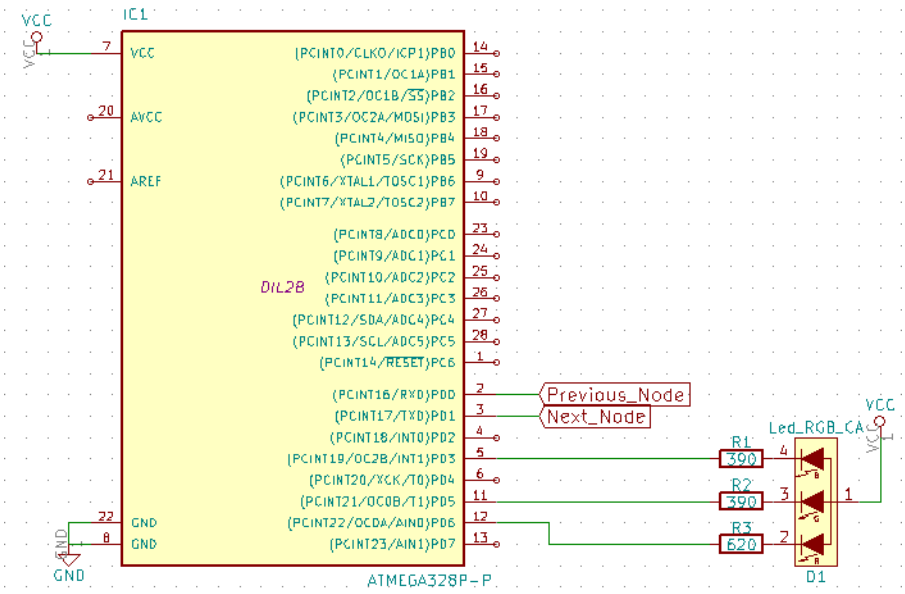


Figure 3: KiCad Schematic

3.2 Highlighted Devices

- ATMEGA328p-pu - Primary microcontroller for the RGB node.
- RGB LED - RGB LED used for the node

3.3 Device/IC 1

This device is a low power 8-bit microcontroller offered by Atmel with 2 8-bit timers and 1 16-bit timer. In this project, both 8-bit timers were used in fast PWM mode, and the USI is used as a UART transceiver. For timer 0, both PWM outputs were used. OC0A sinks the red LED, OC0B sinks the green LED. OC0B was used instead of OC0A because OC0A also functions as MOSI from the SPI programmer. In timer 2, only one output was used. OC2B sinks the blue LED. 430 bytes of the 32 KB flash were used. 52 bytes of the ram were used, 50 of these being the stored array for up to 16 nodes. The absolute maximum current consumption, sourcing or sinking, is 20 mA with a maximum of 200 mA for the entire package.

3.4 Device/IC 2

This device is a through-hole four pin T-1 3/4 5mm LED common-anode package. The red's forward voltage is around 2v, the green's forward voltage is around 3v, and the blue's forward volage is around 3v. The absolute maximum forward current of any

LED is 20 mA. For this project, the LED's maximum current was around 5 mA, which is well below the absolute maximum ratings of both devices.

4 SYSTEM DESIGN THEORY

This design required analyzing what peripherals the microcontroller had to be used, and what was the most efficient way to use these.
This microcontroller has a dedicated USART transceiver module and timers that operate in fast PWM mode, which in this case, acts like a DAC. Combining these two with some logic can create a node. In the current implementation, up to 16 nodes may be used. This can easily be modified to accommodate up to 680 nodes, even though the protocol limits this to 255 nodes. If greater than 255 nodes are used, the protocol needs to be reexamined and modified to have two bytes for packet length instead of one.
Since the RGB LED is common anode, there is an issue. When AVR's PWM value is set to 255 (100% duty cycle), there is a 1-clock cycle pulse of 0. This is an issue for common anode LEDs because it appears as a slight glowing light. To fix this, the timer uses fast PWM with inverted output, and the OCRnm timer compare values are subtracted from 255 and set.

4.1 Design Requirement 1

The node receives a packet from a PC or another node. This packet contains a byte for the amount of bytes in the packet, followed by the LED duty cycles needed for this node, followed by the rest of the PWM values for other nodes, and finally a stop bit. The best way to implement this in this microcontroller is to accomplish as much as possible with the hardware, and to accomplish the rest with software, using sleep modes as much as possible.

5 Getting Started/How to use the device

5.1 Connecting power

This is as easy as it sounds. Connect the following pins to Vcc and ground

- VCC - Pin 7
- GND - Pin 8,22

Please note that the input voltage needs to be somewhere between 1.8 and 5.5v. When connecting to other nodes, these nodes need to be within 0.5 volts of this node.

5.2 Connecting other nodes

This is just as easy as connecting power. The only thing you need to watch for is TXD of the previous node needs to be connected to RXD of this node, and TXD of this node needs to be connected to RXD of the next node.

- TXD - Pin 3 (PD1)
- RXD - Pin 2 (PD0)

After this, the RGB LED needs to be connected.

5.3 Connecting the LED

After this, connect the RGB LED to the following pins:

- R - Pin 12 (PD6)
- G - Pin 11 (PD5)
- B - Pin 5 (PD3)
- GND - Pin 8

The node should be all ready to go

6 Getting Started Software/Firmware

6.1 Programming the device

Before this tutorial, you need to have avr-gcc, avrdude installed.
First, you connect your favorite ISP programmer in SPI.

- MISO - Pin 18 (PB4)
- MOSI - Pin 17 (PB3)
- RST - Pin 1 (PC6)
- SCK - Pin 19 (PB5)
- VCC - Pin 7
- GND - Pin 8,22

Following this, you need to make sure the device is connected. Open your favorite terminal and run

```
$ avrdude -c usbasp -p m328p
```

The fuses should be E:FF, H:D9, L:E2. If not, refer to here for writing fuses. following this, in the ”atmega328p” folder, fun the following command:

```
$ make install
```

This will program the device and verify the chip to make sure that it's programmed correctly. The node should be completely setup and ready for use. If an issue is encountered and a solution cannot be found, please search AVRfreaks. If an issue is still not found, feel free to make a post on that website, as it is one of the largest microcontroller communities on the planet.

7 Test Setup

For setup, please refer to the prevous section "Getting Started/How to use the device"

7.1 Test Data

It was discovered using the "echo" command, data can be piped directly to the USB/serial converter, making scripting a breeze. A script was written that set the LED red, green, blue three times at varying intensity.
A component of the script can be found below:

```
$stime=0.01;

$echo -n -e "\x03 \xFF \x00 \x00 \x0D" >> /dev/ttyUSB0;
$sleep $stime;
$echo -n -e "\x03 \x00 \xFF \x00 \x0D" >> /dev/ttyUSB0;
$sleep $stime;
$echo -n -e "\x03 \x00 \x00 \xFF \x0D" >> /dev/ttyUSB0;
$sleep $stime;
```

8 Design Files

8.1 Schematics

a schematic can be found in figure 3.

8.2 Bill of Materials

This bill of materials is priced in quantities of 1000. If surface mount components are used, price can be significantly reduced.

ITEM	QTY	PRICE/EA
ATMEGA328p-pu	1	$1.45
RGB LED	1	$0.09
1/4 Carbon Film Res 620Ω	1	$0.01
1/4 Carbon Film Res 390Ω	2	$0.01
35mm x 47mm breadboard	1	$0.14
TOTAL		$1.71