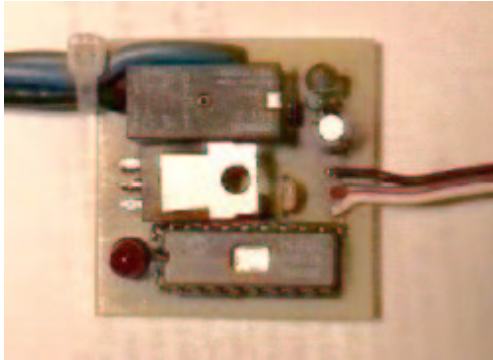


SPD-1 PWM Speed Controller

Andy Shaw



Probably everyone that plays around with both radio control models and microcontrollers has a go at this one! Having tried various commercial speed controllers I decided to have a go myself. My original design made use of a MOSFET H Bridge to avoid having a reversing relay (I really hated the way you had lots of chatter from some speed controller's reversing relay). However this seemed to be rather over the top for the type of scale models that I build. Instead I decided to build a controller with a reversing relay but avoiding all of the chatter. So the basic specification got to look like this...

- No Relay chatter.
- Soft Start.
- Safe switch on (the props on Submarines can be very sharp!).
- 5-10A current handling.
- High frequency PWM.
- Failsafe on signal loss.
- LED status indicator (I like flashing lights!).

Hardware

As you can see from the photograph the hardware is minimal. A MOSFET to provide the PWM switching, a small DPDT relay for reverse operation, a driver transistor for the relay, the all important LED and a few smoothing capacitors. The chip I used was the PIC 16C61 (basically a 16C71 without the A/D).

Radio Control Input

When developing a previous project I created a set of Macros that provide a simple timer based scheduling system. This in a slightly updated form was the base for this project. Again I make use of TMR0 to provide the underlying timing of the system. Making use of my preferred setup so that a standard radio control pulse (See Figure 3.) is represented by a value of -64..+64. Unlike the earlier work this project does not perform most of the capture of the R/C pulse at interrupt time. Doing things this way caused many problems with jitter in the PWM output. Instead the interrupt routine simply captures the value in TMR0 at the start and end of the pulse and schedules a non-interrupt task to actually perform the processing. As with my previous project the R/C input routines provide detection of loss of signal and trigger a failsafe mode.

PWM Generation

I wanted as high a frequency PWM as possible without using hardware or fast clock rates. I use a standard 4MHz part, so I decided upon a 2KHz (approximately) PWM frequency (see figures 5 and 6). At this frequency I found I could generate a pulse train that had a step granularity of 8uS which gives 64 distinct steps. The smallest pulse I could generate was 10uS in length (See figure 4.). The code actually performs all of the setup work outside of the main PWM loop. This loop is the highest priority task and when called assumes that it is partway through the longer part of the cycle. It synchronizes with TMR0 and completes this longer phase. It then makes use of a software loop to generate the shorter part of the cycle. When this is complete it sets things up for the longer cycle and allows other tasks to run.

All of the other tasks run during the longer part of the cycle. These tasks include: R/C input helper task, PWM Move generation, PWM Parameter generation, R/C timeout and the status indicator task.

Of these tasks perhaps the most interesting is the PWM move generation. This tasks job is to take the R/C input and to change the PWM output to match it. It is this task that provides the soft start function by providing a smooth ramp up of the PWM output rather than a sudden jump. This task also controls the reversing relay. Providing two switching points rather than one, thus introducing a

degree of hysteresis prevents relay chatter. A nice additional touch is the use of a timer so that the relay is always turned off when the R/C stick is at neutral for any length of time. The final function that is provided by this task is to ensure that the relay is only switched when the motor is at rest. This allows a lower rated (and smaller) relay to be used. A control input moving from full forward to full reverse results in output from this task that takes the PWM to zero, a small pause (to allow the

motor to stop), the relay is switched, then a smooth ramp up to full power. The actual translation of R/C input to PWM output is done via a look up table. This allows the power curve to be soft. In the current implementation the curve provides a slower ramp in speed at the low to middle part of the curve. This allows finer control at the more commonly used speeds.

Specification

PWM Frequency	2KHz
PWM Resolution	8us
Minimum Pulse Width	10us
RC Input	1-2ms pulse every 20ms
Current	4A Continuous, 10A Peak
Operating Voltage	4.8V-6V
Drive Voltage	4.8V-24V

Special Features

Soft Start
 Failsafe On Lost Signal
 LED Status Display
 Safe Start

Status Indicator

Flash	Status
1	Normal Operation
2	Safe Start Waiting For Neutral
3	Reverse Relay operating
4	Full Power
6	Failsafe Signal Lost

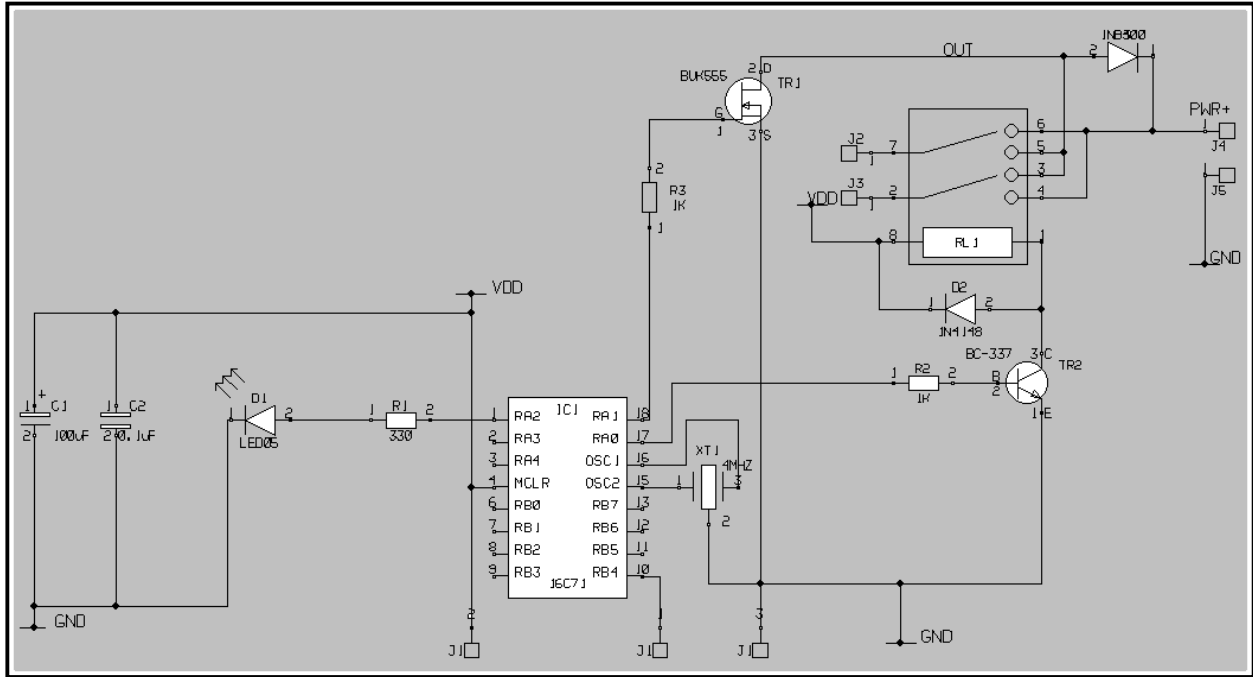
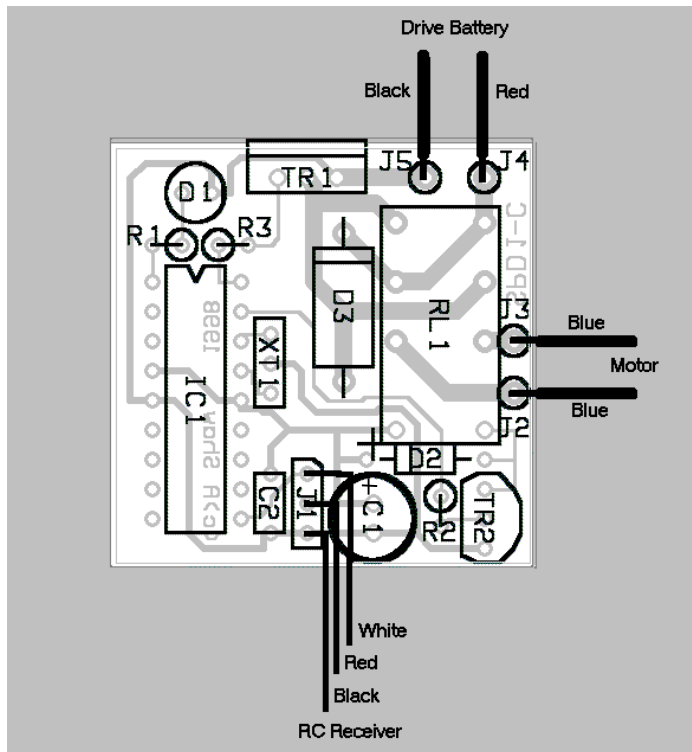


Figure 1. Full circuit diagram for the SPD1 PWM Speed Controller.



Components

IC1	PIC16C71 SPD RC
TR1	BUK555
TR2	BC-337
D1	Red 5mm LED
D2	1N4148
D3	1N5400
XT1	4MHZ Resonator
R1	330 0.6W
R2	1K 0.6W
R3	1K 0.6W
C1	100uF 16V
C2	0.1uF
RL1	G5V-2 5V Relay

Figure 2. Component layout and connections

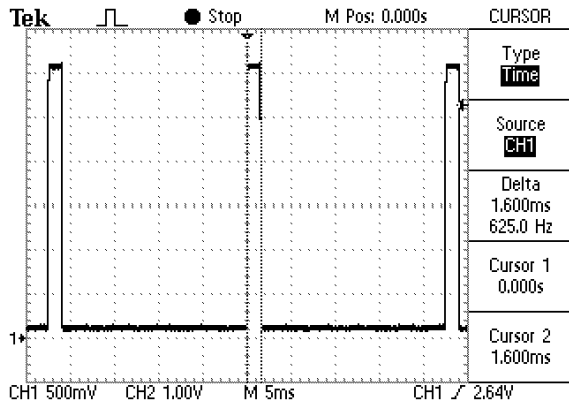


Figure 3. R/C pulse at neutral

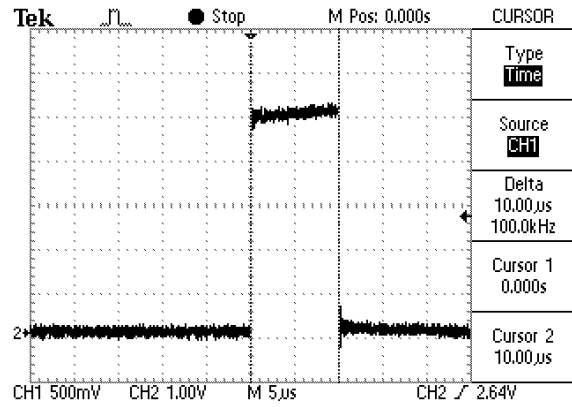


Figure 4. Minimum PWM pulse width

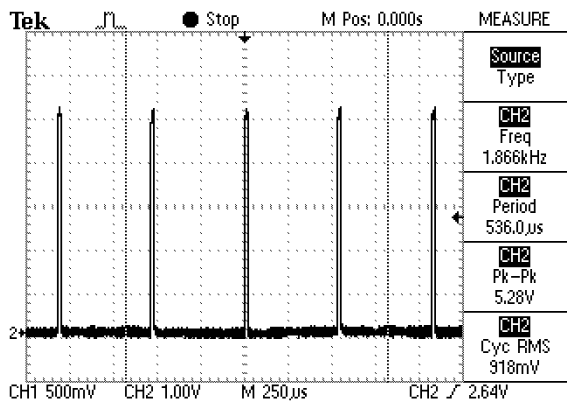


Figure 5. PWM pulses at minimum power

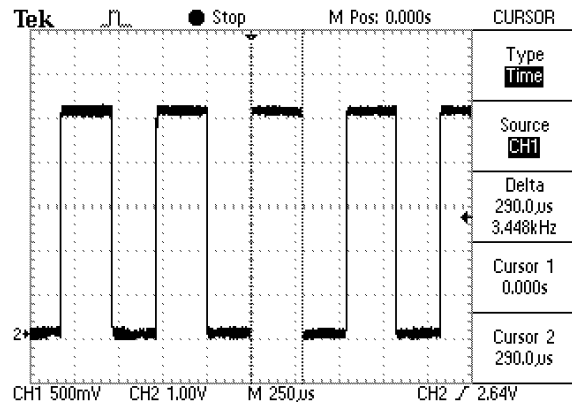


Figure 6. PWM pulses at half power