

Electronic Receiver Switch Schematic Diagram

signal pulse then FF2 will shift in a 0, otherwise it will shift in a 1. This 'comparison' happens every 50ms and affects the output of FF2. DATA OUT(2), and its complement both connect to a switch that determines which one connects to the power MOSFET. By flipping the switch, we simply invert the ON/OFF operation of the peripheral device (much like a servo reverse).

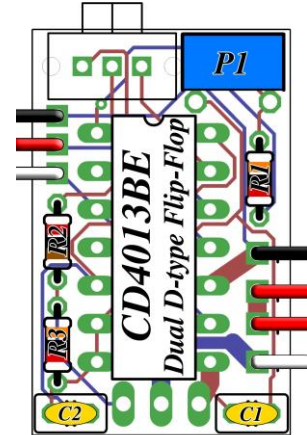
After the SPDT switch, we have an IRLZ44N power MOSFET which acts like a big switch to turn the LOAD ON and OFF. MOSFET's do not require current to turn on, just a charge present at the gate. For this reason R3 is placed between the gate and ground to guarantee that the MOSFET turns OFF when the receiver is turned off (the gate charge will sink into ground). MOSFET's also have fairly high gate-capacitance, and we take advantage of this with R2, which makes the gate charge accumulate slowly when the MOSFET is to be turned on - this is like a "soft-start" feature (but it will still turn on faster than you can see).

Finally, C2 connects across the receiver power lines simply to help keep the supply voltage on our circuit board stable - standard procedure for many circuits.

#### Assembly instructions:

Use the printed circuit board (pcb) layout diagram to the right as well as the images below to place and solder your components. The solder in this kit is a "no-clean" solder which means that the flux in it is designed to have no effect on circuit operation and be aesthetically satisfactory. It is recommended that you solder the CD4013BE chip first (make sure you get the orientation right - the chip has a notch at one end), then move on to the resistors and capacitors (these have no orientation). Before soldering the MOSFET in place, bend the legs 90 degrees as shown below. Clip off the component leads as you go (a flush cutting tool makes neat work of this).

Finally, solder the wires in place. Frequently when soldering wires to a pcb it is easier to first tin (soak with solder) the wires so that the individual copper strands don't get snagged as you insert the wire (if even one strand bends to the side instead of going through the hole it could touch something else and cause a short on your board). The best way to do this is to strip off far more insulation than necessary (perhaps 1/2"), twist the strands so they hold together very tightly with the smallest total diameter possible, then tin the entire stripped portion with solder. When twisting the strands you'll never get the very ends to hold tightly, and tinning often leaves a thick lump of solder at the end that you can't completely remove with the soldering iron. This is why you should expose far more copper than needed, focus on tinning the portion nearest the insulation very tightly and cleanly, then clip off the part you don't want. If you are having trouble getting the wire to fit, you can always neatly shear off a portion of strands prior to tinning. One quick note on the power wires: if you look at the board you will find that the red power wires don't actually have to be soldered to the board, since all that is achieved is that they are connected together. If you prefer, you can just connect the positive end of your load battery directly to your load, and you will have something that looks more like the battery, load, MOSFET circle on the far right side of the schematic. The power wires could be inserted from either side of the pcb and routed forward or backward per your preference (photos below show a couple of different arrangements).



Thanks for purchasing this Hansen Hobbies Receiver Switch Kit! Please make sure you have everything:

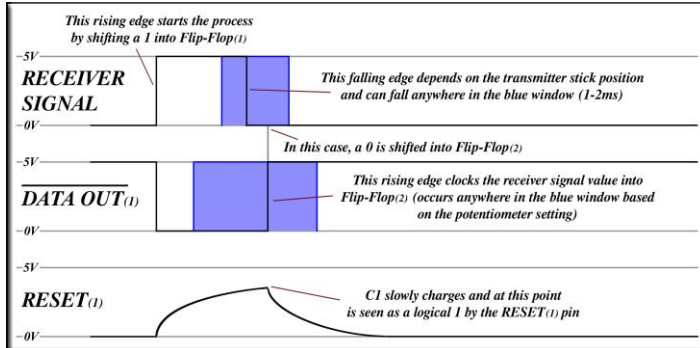
- .69x1.13" PCB (1)
- CD4013BE chip (1)
- IRLZ44N MOSFET (1)
- SPDT switch (1)
- 100KΩ pot (1)
- 20KΩ resistor (2)
- 1KΩ resistor (1)
- 33nF capacitor (2)
- 8" receiver lead (1)
- .75" clear heat shrink
- solder 18"

#### Circuit explanation:

The CD4013BE contains two D-type flip-flops. If you have no idea what a flip-flop is, then you'll benefit from a little online research. Here's a quick review: a flip-flop has the ability to 'lock in' a logic value (0 or 1) and make that value present at the DATA OUT pin. The locked-in value is only allowed to change when the CLOCK pin rises from low to high (a rising edge). At that time the logic value present at the DATA IN pin is shifted in and the DATA OUT pin outputs that value. Note: "DATA OUT" with a line over it is simply the inverted DATA OUT - always. There are also SET and RESET pins that let us force the DATA OUT value 1 or 0 respectively whenever we want, simply by pulling one or the other high; the ones we aren't using are connected to ground.

Radio control receivers use Pulse Width Modulation (PWM) to communicate with a servo. The pulse occurs every 50ms and is variable between 1ms and 2ms long as shown in the graph below - the width of the pulse tells the servo what position to be at. Our goal is to decode this signal into an ON or OFF state so we can turn peripheral devices ON/OFF during flight. The best approach is to create a signal in phase (rising edges occur at the same time) with the receiver signal, then compare this new signal width with the receiver signal width and turn the device ON if the receiver signal is longer, or OFF if it is shorter.

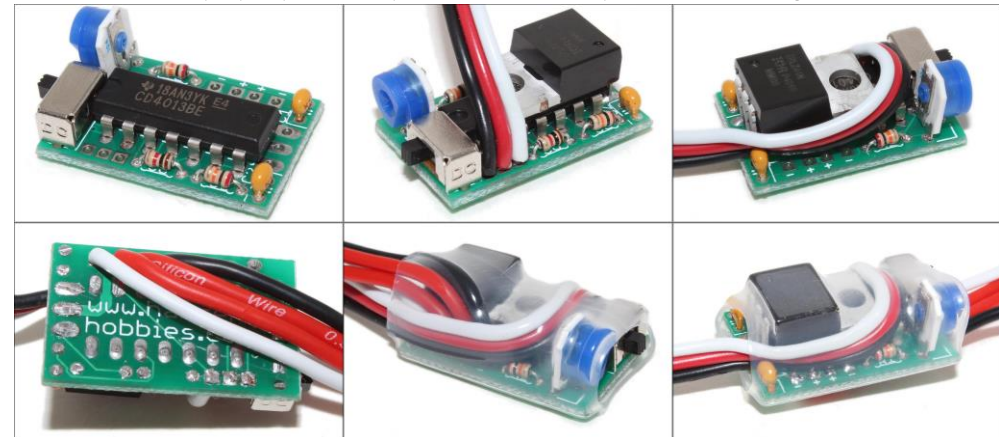
We create the comparison signal by connecting the FLIP-FLOP(1) (FF1) CLOCK(1) to the receiver signal, and the DATA IN(1) to Vcc (positive). Whenever the receiver signal rises, a 1 will be shifted into FF1 and DATA OUT(1) will output a 1. When this happens C1 will begin to charge through R1 and P1 (this takes a certain



amount of time) - when the voltage across C1 gets high enough to be seen as a logical 1 on the RESET(1) pin, then FF1 will reset and DATA OUT(1) will output a 0. This happens at every rising edge of the receiver signal and is shown in the graphs to the left.

We now have a PWM'd signal occurring at DATA OUT(1) and the width of its pulse is determined by the

resistor-capacitor combination on RESET(1). By adjusting P1 we can change the length of the pulse from about 0.5ms to 2.5ms, which will let us make it longer or shorter than any valid receiver signal. Now we'll use FF2 to compare the two signals. Notice that the receiver signal is connected to DATA IN(2) and the inverted output from FF1 is connected to CLOCK(2). This gets a bit confusing, but the graphs above should help a lot. As you can see, the inverted DATA OUT(1) is normally high, and pulses low for a short period (as determined by P1). When it rises back up, FF2 shifts in the receiver signal value. If the DATA OUT(1) pulse is longer than the receiver



After all the components are soldered in place (and leads clipped flush) check over your circuit to make sure there aren't any bridged connections or misplaced components. When you're satisfied that everything is working correctly, you can protect the circuit with the supplied heat shrink tubing. We recommend you route the receiver lead wires as shown - the wires will be clamped down among the pcb, components, and heat shrink to prevent strain on the solder joints.

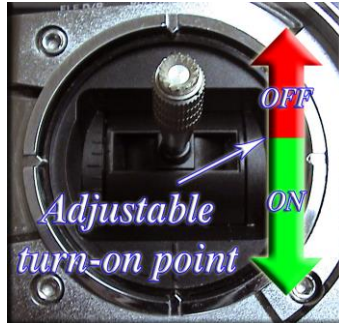
**Operating Instructions:**

This device is designed to allow **ON/OFF** control of onboard devices like lights, glow plug drivers, motors, smoke systems, rocket boosters, solenoids, relays, and whatever else you can think up. You can safely power a device with a DC supply ranging up to **55V** at up to **5A**. The voltage from the receiver must can be between **3** and **18V**.

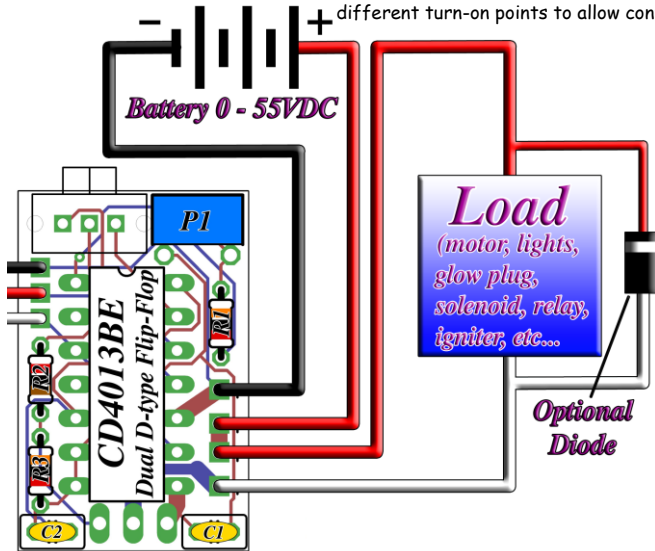
**Hook up:** Setting up your **ERS** is very easy - you'll need something that you can visibly turn **ON/OFF** so you can adjust the **turn-on point** - a small light bulb or LED will work fine for this. First plug the **ERS** into the receiver channel that you want to use to activate your device. If you intend to operate the **ERS** with a toggle switch on your transmitter we recommend you do the initial setup/testing using an adjustable channel so that you can center the turn-on point using the potentiometer. Next connect your power source (any battery - can be your receiver battery) to the red and black wires, and the device to be powered to the red and white wires as shown (it doesn't matter which red wire goes where, as they are internally connected). If your device needs to have special orientation make sure its positive terminal connects to a red wire, and the negative terminal to the white wire. Also, if your device is a motor, relay, solenoid, electromagnet, etc... a diode (included) should be connected across its terminals as shown below. This will protect the **ERS** from reverse high-voltage spikes associated with inductive devices.

**Adjusting:**

Move the transmitter stick to the position where you want the device to turn **ON**, then slowly turn the potentiometer on the **ERS** until you see your device turn **ON/OFF**. Your device will always be **ON** either above or below this point, as shown to the right. Flipping the small slide switch on the ERS will cause the **ON** and **OFF** regions to switch (like a servo reverse). If you plan to operate the **ERS** with a toggle switch on your transmitter first set the potentiometer to switch at mid-stick using an adjustable channel, then plug it into your toggled channel. The toggled channel should have all trims set to zero and configured for maximum throw.



**Notes:** Multiple **ERS** can be connected to one receiver channel (using a y-harness or similar) and set for different turn-on points to allow control of multiple devices.



You can use this device to power electric motors, however we don't recommend it for use in place of a speed control to power flight motors because it doesn't have the safety features that most Electronic Speed Controls have built-in.

**Troubleshooting:**

If you finish your **ERS Kit** and find that it is not working properly, you may find this section helpful.

If you plug your **ERS** into your receiver and any part of the circuit gets warm or hot, immediately remove power and check your work for shorts.

Before concluding that the circuit isn't working and continuing to the suggestions below, plug a servo into the receiver channel that you're using and make sure it moves through its full range (if it doesn't then something is wrong with your receiver or transmitter settings). Also set the potentiometer to the center of its range. If you still don't observe any **ON/OFF** operation, continue on below.

The best way to troubleshoot a circuit is with an oscilloscope. If you have and know how to operate one, then you probably don't need advice on how to troubleshoot this circuit, or you can follow along below as I describe debugging with a multimeter.

If you don't have a scope, then next best tool is a multimeter that lets you test voltage and continuity. Oscilloscopes are designed to measure fast moving signals whereas multimeters (excluding fancy ones) are designed to only measure steady state levels. However, we can still get a lot of information from a multimeter reading even with a high speed digital circuit. Below, when I mention measuring a voltage, it is always with reference to ground (negative), so the negative/ground probe should be connected to one of the many ground points (receiver **black wire**, pins **6, 7, 8, 10** of the **CD4013BE**, and several other points that can be seen in the schematic).

The first thing to measure is the input voltage on the **CD4013BE** chip (look at the schematic - pins **7 & 14**). It should be around **5V** (equal to your receiver voltage). Next measure the voltage of the receiver signal (pin **3** or **9**). As described above, this is a pulsed signal, but your multimeter will probably still see something. For a **~1.5ms** pulse occurring every **50ms** I expect a voltage meter to read about **1.5/50 = 3%** of the supply power (about **0.15V** if you receiver operates on **5V**). You may even see your multimeter changing value erratically since it is connected to a pulsed voltage. If we see something like this then we know the circuit is probably getting the signal. If you see a steady **0V** or **5V** then there's a chance that this signal line is shorted out and you need to follow it around the board to find out where. You can also switch your meter to do a continuity test to see if two things are connected where they shouldn't be. If everything above looks good then at least you know the circuit is getting power and signal.

It's often faster/easier to work backwards in the circuit. Maybe everything is working fine but the **MOSFET** is broken so our device never switches **ON/OFF**. The gate of the **MOSFET** which connects to **R2 & R3** should be either **0V** or **5V** and should change when the transmitter stick moves past the turn-on point. If you don't get activity here take a step back before the SPDT switch and look at pins **12 & 13**. If you see activity there then you know the problem is somewhere between there and the gate of the **MOSFET**.

Metering the node with pin **2 & 11** will tell you if the first stage of the flip flop is working. You should see something that looks like a servo signal here (around **0.15V** as above) and you should be able to make it go up and down by turning the potentiometer. In fact, you should be able to feed the signal from this node into a servo and control its position by turning the potentiometer.

If all of the above hasn't helped you sniff out the problem, then a last ditch effort at debugging is to start performing continuity tests everywhere. These tests should be done with power disconnected. Set your meter to test continuity (every meter should be able to test continuity) and start checking all the connections as they're shown in the schematic. Just as important, check to make sure things that should not be connected aren't. You can also set your meter to measure resistance (and possibly capacitance) and check that the parts are reading the correct values.

Even for a circuit this small, the above suggestions are just a start. Look at the schematic, think about what each node of the circuit should be doing, and devise a test to see if it's acting the way it's supposed to. In this kind of work little scraps of information can tell volumes. With a trained eye and a good understanding of basic circuitry, one can even debug a circuit like this just by probing various points with an LED + resistor and looking at the brightness of the light.

If you have any questions or comments about this kit please email [chris@hansenhobbies.com](mailto:chris@hansenhobbies.com).