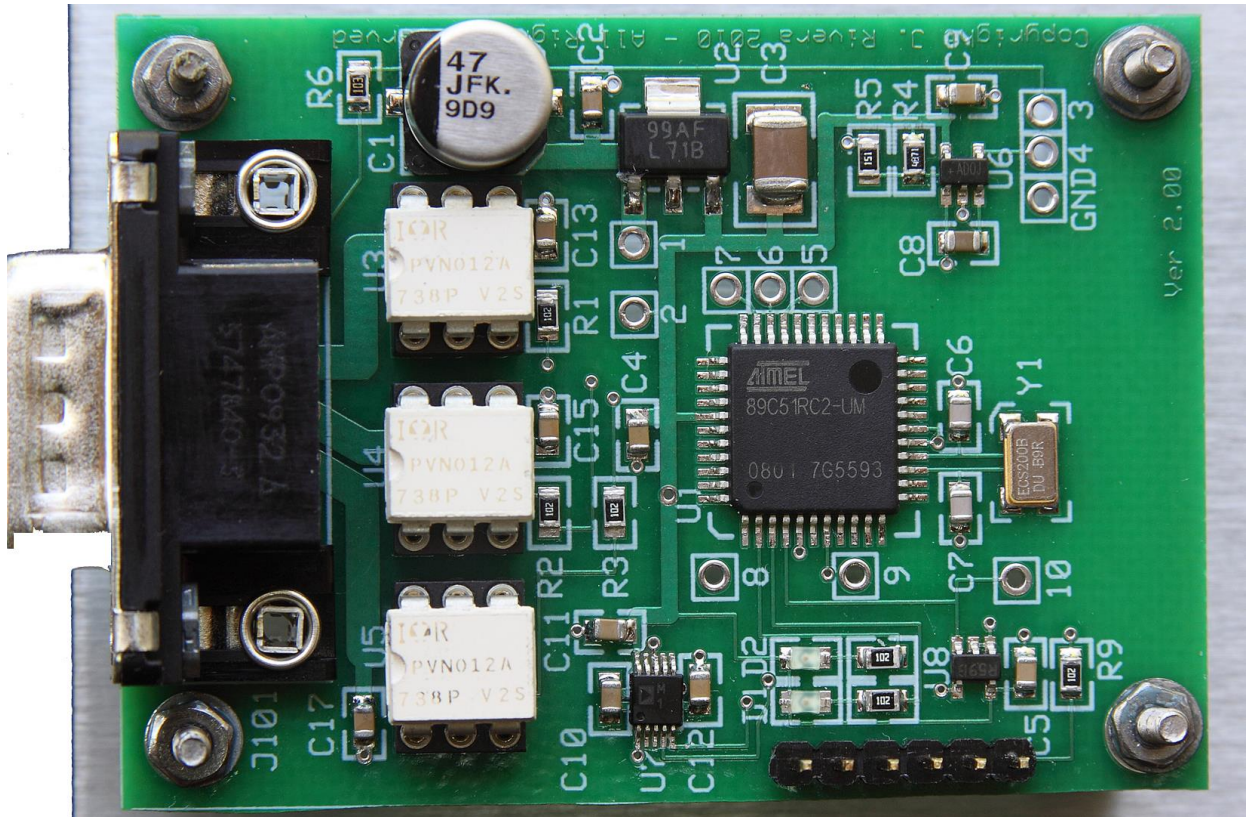


Helicycle Main Rotor RPM Alarm – For Experimental Use Only



# Helicycle Main Rotor RPM Alarm – For Experimental Use Only

15 January 2010

The first build of the software is debugged and operational. The alarm works just as I expected. The accuracy of the alarm trip points is 100 times better than it needs to be, but the important thing is that those alarm trigger points will not drift with time or temperature. The digital circuit is tied to a quartz crystal oscillator. The three alarms are the 610 RPM, the bottom of the green band (low rotor), 620 RPM, the top of the green band (high rotor #1) and 635 RPM, the red line (high rotor #2). The three alarms all triggered within a very small fraction of one RPM from their intended points. Here's the raw data:

RPM	CPS	CPS * 4	msec	Actual Trigger Point	Actual Trigger RPM
<b>Low Rotor (Bottom of Green)</b>					
609.0	10.150	40.600	24.631		
609.1	10.152	40.607	24.626		
609.2	10.153	40.613	24.622		
609.3	10.155	40.620	24.618		
609.4	10.157	40.627	24.614		
609.5	10.158	40.633	24.610		
609.6	10.160	40.640	24.606		
609.7	10.162	40.647	24.602		
609.8	10.163	40.653	24.598		
609.9	10.165	40.660	24.594		
<b>610.0</b>	10.167	40.667	24.590	40.666630	<b>609.99945</b>
610.1	10.168	40.673	24.586		
610.2	10.170	40.680	24.582		
610.3	10.172	40.687	24.578		
610.4	10.173	40.693	24.574		
610.5	10.175	40.700	24.570		
610.6	10.177	40.707	24.566		
610.7	10.178	40.713	24.562		
610.8	10.180	40.720	24.558		
610.9	10.182	40.727	24.554		
611.0	10.183	40.733	24.550		

## Helicycle Main Rotor RPM Alarm – For Experimental Use Only

### High Rotor (Top of Green)

619.0	10.317	41.267	24.233		
619.1	10.318	41.273	24.229		
619.2	10.320	41.280	24.225		
619.3	10.322	41.287	24.221		
619.4	10.323	41.293	24.217		
619.5	10.325	41.300	24.213		
619.6	10.327	41.307	24.209		
619.7	10.328	41.313	24.205		
619.8	10.330	41.320	24.201		
619.9	10.332	41.327	24.197		
<b>620.0</b>	10.333	41.333	24.194	41.333900	<b>620.0085</b>
620.1	10.335	41.340	24.190		
620.2	10.337	41.347	24.186		
620.3	10.338	41.353	24.182		
620.4	10.340	41.360	24.178		
620.5	10.342	41.367	24.174		
620.6	10.343	41.373	24.170		
620.7	10.345	41.380	24.166		
620.8	10.347	41.387	24.162		
620.9	10.348	41.393	24.158		
621.0	10.350	41.400	24.155		

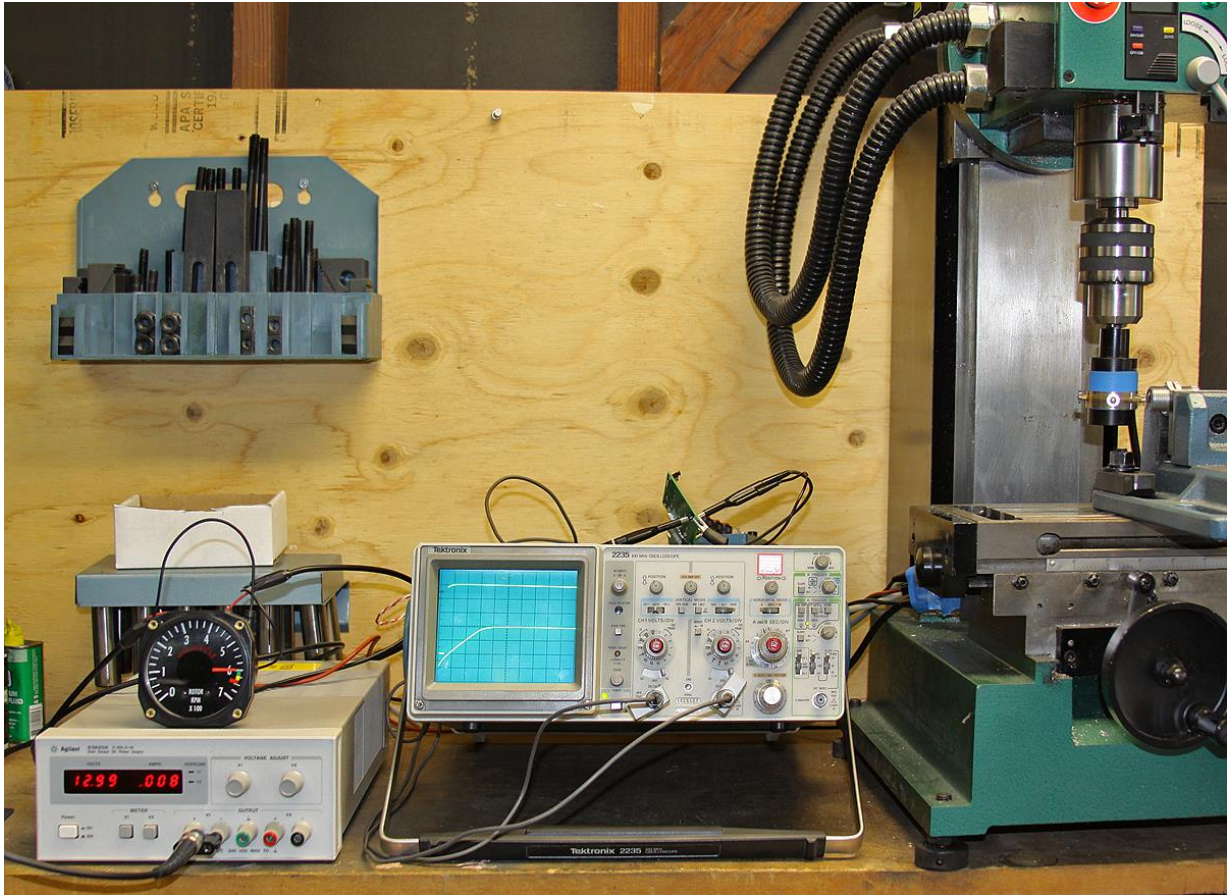
### High Rotor (Red Line)

634.0	10.567	42.267	23.659		
634.1	10.568	42.273	23.656		
634.2	10.570	42.280	23.652		
634.3	10.572	42.287	23.648		
634.4	10.573	42.293	23.644		
634.5	10.575	42.300	23.641		
634.6	10.577	42.307	23.637		
634.7	10.578	42.313	23.633		
634.8	10.580	42.320	23.629		
634.9	10.582	42.327	23.626		
<b>635.0</b>	10.583	42.333	23.622	42.333290	<b>634.99935</b>
635.1	10.585	42.340	23.618		
635.2	10.587	42.347	23.615		
635.3	10.588	42.353	23.611		
635.4	10.590	42.360	23.607		
635.5	10.592	42.367	23.603		
635.6	10.593	42.373	23.600		
635.7	10.595	42.380	23.596		
635.8	10.597	42.387	23.592		
635.9	10.598	42.393	23.589		
636.0	10.600	42.400	23.585		

## Helicycle Main Rotor RPM Alarm – For Experimental Use Only

December 29, 2009

I'm doing this log backwards, so the newest entries are at the front. Today I characterized the input waveform again by hooking up the rotor tachometer to a spare hall effect rotor RPM sensor and spinning the magnets on my mill. Here's the setup:

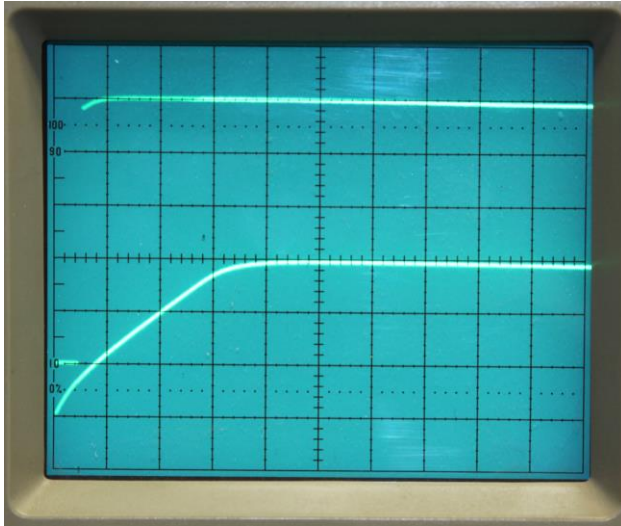


To the right you can see the magnets and the sensor held in my machinist vice. The alarm circuit board is sitting on top of the oscilloscope at an angle with two probes connected to it. To the left is the power supply that is powering the RPM indicator and the sensor. In this picture the mill is indicating 630 RPM and the RPM indicator shows 600.

My first check was to insure that connecting the alarm circuit to the indicator didn't affect the indication. It shouldn't because its high impedance and it didn't.

The next thing I wanted to do was to characterize the rise and fall times of the input pulses (the output from the hall effect sensor.)

## Helicycle Main Rotor RPM Alarm – For Experimental Use Only

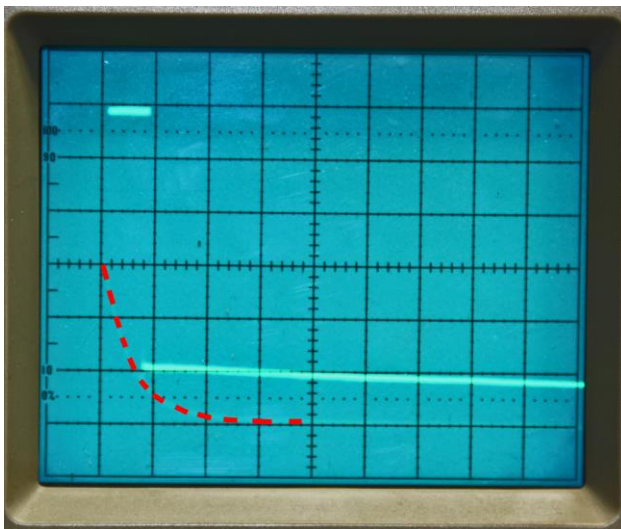


In this scope shot I'm triggering on the leading edge of the sensor pulse and looking at both channels. The oscilloscope is set as follows:

- Chan 1 (hall effect output): .2 V/div
- Chan 2 (output of comparator): 1.0V/div
- Sec/div: .5 ms

The top trace is the output of the comparator with a very fast rise time as expected. The bottom trace is the output of the hall effect sensor and the rise time is about 1.5 ms. You can see that the comparator is triggering at about 0.15 Volts which is what I designed the circuit

to do. This will give it some noise immunity since anything below the trigger point will be ignored..



Here are the waveforms when triggering on the trailing edge of the pulses. The sec/div is set to 20  $\mu$ s and the hall effect trace is very faint. I've added some red dashes to make it visible. The trigger is offset by one division.

Again, the comparator produces very fast fall times as expected, and the hall effect sensor is about 30  $\mu$ s (millionths of a second.)

The microcomputer's counter triggers on the leading edge of the comparator pulses so it will be very happy. And now I know where to set the arbitrary waveform generator to faithfully reproduce the hall effect sensor input when

testing in the lab. The hardware design is essentially complete.



Lastly I decided to check the accuracy of the indicator. It was way off but there is an adjustment in the back. I calibrated it over the range from 600 to 650 RPM. With the mill indicating 650 RPM, 43.469 Hz on the Fluke 287 translates to an actual RPM of 652.04. I'm shooting from slightly below to avoid a reflection so the parallax error would make the indicator look slightly low. Bottom line – it's now close enough.

## Helicycle Main Rotor RPM Alarm – For Experimental Use Only

December 19, 2009

I've added a program flow chart on the next page to describe how the alarm circuit is going to function. This is the first cut and they'll be many tweaks as the alarm is tested.

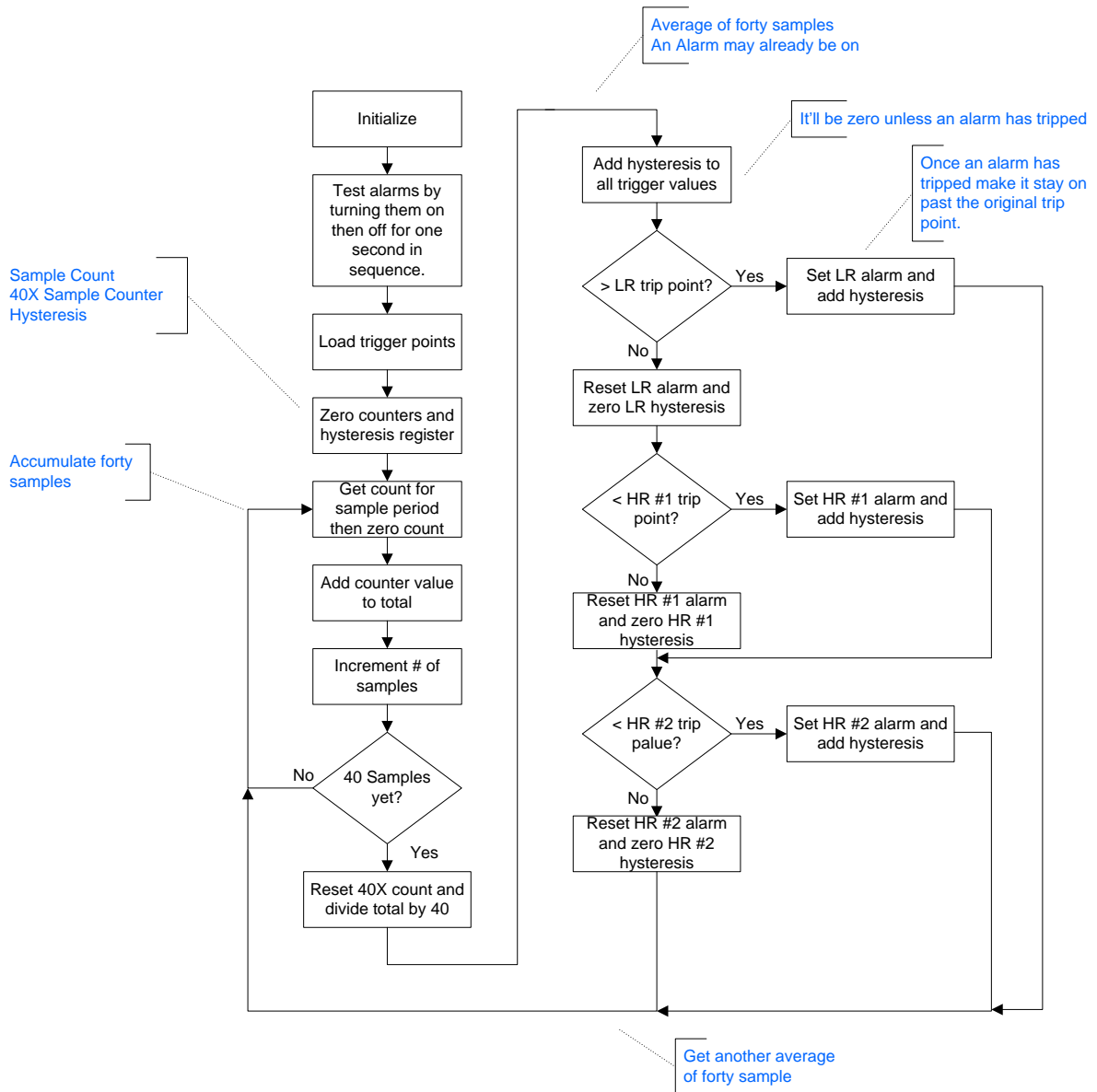
The microcomputer is interrupt-driven, meaning that it doesn't have to poll its inputs to see if there is data available. In this circuit the leading edge of a rotor tachometer pulse will cause an interrupt and then the microcomputer will go service it. That makes life easy.

Because the magnets may not be exactly evenly spaced at 90-degree angles around the shaft, and may not be of equal strength, the duration of the four periods for each revolution of the rotor shaft might not be the same. This could cause complications when attempting to determine the RPM of the rotor. I'll average many samples to smooth the values, and to get around this potential irregularity between magnets I'll simply average multiples of four. That way I'll always get complete sets of four samples and the individual magnet offsets won't make any difference.

Once an alarm condition has occurred I want to make sure the alarm is activated long enough to be observed and not flicker on and off if the rotor RMP is right on the edge of the trigger point. I'll deal with that by adding hysteresis when an alarm is activated. This means that the RPM will have to get past the original trigger point by the hysteresis offset before the alarm condition can be cleared. For example, if the low rotor alarm was triggered at 610 RPM (the bottom of the green) I can make the alarm stay active until the RPM has risen to 612 RPM. Of course, the final settings will have to wait for flight testing to see what works best.

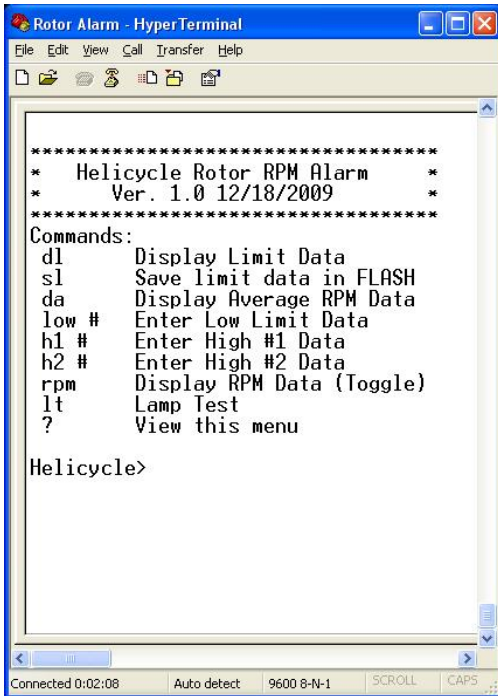
Well, that's it for today. Tomorrow I'll go back to my mill and run the actual Hall Effect sensor and magnets that we use on the helicycle to characterize the pulses. The rise and fall time of the pulses can now be accurately simulated in the lab using the Agilent arbitrary waveform generator.

## HELICYCLE ROTOR ALARM FLOW CHART



J. Rivera – 12/19/2009 – ver 1.00

## Helicycle Main Rotor RPM Alarm – For Experimental Use Only



```
Rotor Alarm - HyperTerminal
File Edit View Call Transfer Help
*****
* Helicycle Rotor RPM Alarm *
* Ver. 1.0 12/18/2009 *
*****
Commands:
dl      Display Limit Data
sl      Save limit data in FLASH
da      Display Average RPM Data
low #   Enter Low Limit Data
h1 #   Enter High #1 Data
h2 #   Enter High #2 Data
rpm     Display RPM Data (Toggle)
lt      Lamp Test
?       View this menu

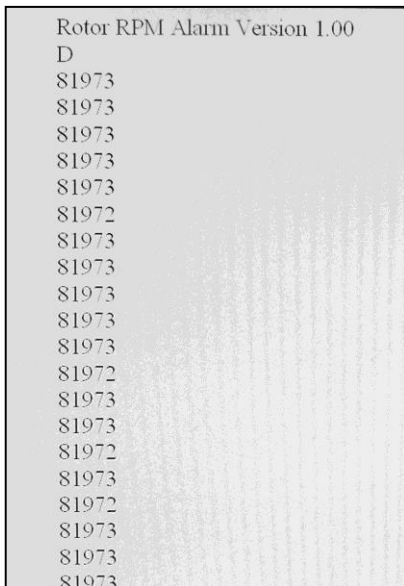
Helicycle>
```

December 18, 2009

How cool is this? The Rotor Alarm now has its own mini operating system. I'll be able to do the setup and debugging from my dumb terminal. I don't intend to make this available to users (if there are any!) since I want to ensure that the unit is functioning as I intend. But it's still cool. I should mention that I'm not doing the programming. I have an expert friend at work that's doing the heavy lifting for me.

December 16, 2009

The circuit board is fully functional, so Version 1.30 is going to get the job done. All the major functions have been tested and a preliminary program is now up and running. I'll add the details at the end of this write-up starting on page 9, but here's where the rubber meets the road:



```
Rotor RPM Alarm Version 1.00
D
81973
81973
81973
81973
81972
81973
81973
81973
81973
81972
81973
81973
81972
81973
81972
81973
81973
81973
```

The alarm works like a stopwatch. Every time a magnet passes under the Hall Effect sensor it starts or stops a counter. The counter is being fed with a stream of very fast pulses from a crystal-controlled clock. The period between pulses at 610 RPM is only .041 seconds – .040666666 to be more precise. In the screen capture at the left you see the accumulated count in the counter for each counter period. The counter is managing to count all the way to 81,973 plus or minus one count. This allows the circuit to measure the RPM very accurately. For example, at 609.5 RPM the count is 82040, a difference of 67 “ticks”. So this circuit can easily resolve 0.5 RPM.



## Helicycle Main Rotor RPM Alarm – For Experimental Use Only

<b>RPM</b>	<b>Period Between Pulses</b>	<b>Count</b>
610 (Bottom of Green)	40.666666 mSec	81973
620 (Top of Green)	41.333333 mSec	80650
635 (Red Line)	42.333333 mSec	78746

Now that the critical alarm points have been determined (see pages 9 and 10) the program can be completed and tested. Once it's checked out I'll describe it in detail.

If you're interested in picking one up drop me a note...

## Helicycle Main Rotor RPM Alarm – For Experimental Use Only

November 28

While I'm waiting for the next, and I hope the last, revision of the circuit board and a few more components to arrive I built a test box so I won't have quite so many loose wires and leads on my bench while I'm working on the alarm circuit board.



This box contains most of the wiring and components that will go into the instrument panel. I'm still debating whether to keep the second rotor alarm indicator light or replace it with the loudest Sonalert that I can find. I'll have one in a few days to play with. It's rated at 95 to 105 dB at two feet. Loudness is subjective but this is in the range where sustained exposure can cause hearing loss and is just short of a power mower at 3 feet. I'm curious to see if my Bose aviation headset can cancel out enough to make it unlikely to be heard in flight. If I

had to I could mount it right behind my head. I want to find a level that will get my attention without causing a heart attack.

If the Sonalert looks like it will get the job done I'll only have two indicator lamps – a low rotor that comes on below 610 RPM and a high rotor that comes on above 620 RPM (the bottom and top of the green.) The Sonalert will start putting out a continuous screech above 635 RPM (red line.) So you'll have two separate warnings when you hit red line – the high rotor indicator lamp and the Sonalert.

I'm planning to power this alarm from the instrument bus. When power is applied the circuit will step through all three alarms in sequence as a self test. After the self test completes the low rotor indicator will illuminate until the rotor gets into the green and then it should extinguish. Between the initialization self test and the low rotor alarm activating, the pilot should have a very good confidence check that the circuit and all three alarm indicators are functional before lifting off. There will also be a manual reset button so the alarm can be tested at any time.

November 22

I've been working on this circuit for a while now and I'm on my 5<sup>th</sup> circuit board. Although the microcomputer design documents say that the processor can be reset using only a capacitor tied to the reset pin, this has proven to be unreliable. I'm going to use a power-on reset (POR) device to explicitly reset the processor. This POR circuit adds an additional feature to the design. It allows a reset button to be installed on the instrument panel. The button will reset the processor which will cycle through the three alarm indicators. It can be used to self-test the circuit at any time.

## INTRODUCTION

There are three critical main rotor RPM limits:

- Low Rotor (Bottom of green at 610 RPM)
- Normal High Rotor (Top of green at 620 RPM)
- Autorotation High Rotor (Red line at 635 RPM)



Because the main rotor tachometer is not an expanded scale indicator, the green band and the red line are very close together as you see. In addition, all analog indicators suffer from several sources of error including calibration, drift, and parallax. Since maintaining proper main rotor RPM is so critical to safe flight I have decided to design a digital rotor RPM alarm circuit that will constantly monitor rotor rpm and produce alarm outputs for all three alarm conditions. Since the circuit will be digital, and referenced to a quartz crystal, it will be extremely accurate. Since it will not undergo the exhaustive testing that certified avionics must pass, it should be treated as an experimental circuit only and not depended on for safety. It's only intended as a second independent way to monitor rotor RPM.

The circuit will bridge across the existing Hall Effect transducer's output at the back of the rotor tachometer in the instrument panel using a high impedance circuit that will not load down the indicator. This will prevent the alarm circuit from adversely affecting the accuracy of the existing rotor tachometer.

There are already various tachometers on the market that have alarm outputs so why am I doing this? The answer is simple – because I want to. It's an interesting challenge and I'm enjoying the process. The skills required to design, build, program, and debug a circuit like this are very volatile and I don't get much opportunity to keep myself current at work. This is my way of maintaining and expanding my skill set in this area. I have no idea if this circuit will be an improvement over existing alarms and I really don't care.

## **Rotor RMP Alarm Features**

The rotor RPM alarm circuit utilizes an in-system programmable microcontroller. Digital rotor tachometer pulses from the Hall Effect transducer are cleaned and shaped by a digital comparator and then fed to the microcontroller. The microcontroller uses these tachometer pulses to gate a counter on and off like a stopwatch. The input to the counter is a crystal-controlled clock. By counting the number of clock pulses that accumulate in the counter the RPM of the rotor can be very accurately measured. When a trigger point is reached for any of the three alarm conditions the microcontroller enables a solid-state relay that provides +13 volts to activate a light or other alarm. The microcontroller is programmed using a standard serial RS-232 link to a PC. Power to the circuitry is conditioned and regulated by an on-board voltage regulator. All components are surface mount and are soldered to a double-sided circuit board.

### **U1**

U1 is a high-performance, 8-bit, 44-pin microcontroller with 32K of flash memory. It utilizes a 20 MHz crystal oscillator (Y1) for precise timing. U1 is programmed using a standard RS-232 serial interface to a PC running Atmel's standard programming environment called, "FLIP". TTL-level serial data is converted to standard bipolar RS-232 levels by U7. In addition to the input from the Hall effect transducer and the three output lines driving the solid-state relays, two light emitting diodes (D1 and D2) are used to monitor the state of the processor, and one spare digital input is included for future use (J2, Pin-4.)

### **U2**

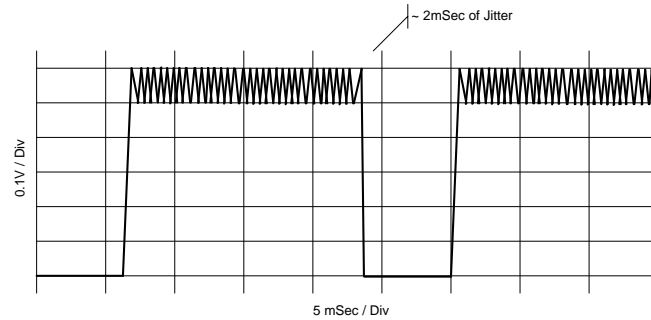
U2 is the on-board voltage regulator. It takes a nominal +13 VDC from the ship and converts it to +5 VDC to power the remaining circuitry. As the name implies, the five-volt output is regulated and filtered and remains stable over an input range from 6.5 to 30 volts. This ensures that the circuit will function even while being exposed to a wildly varying input voltage.

### **U3, U4, U5**

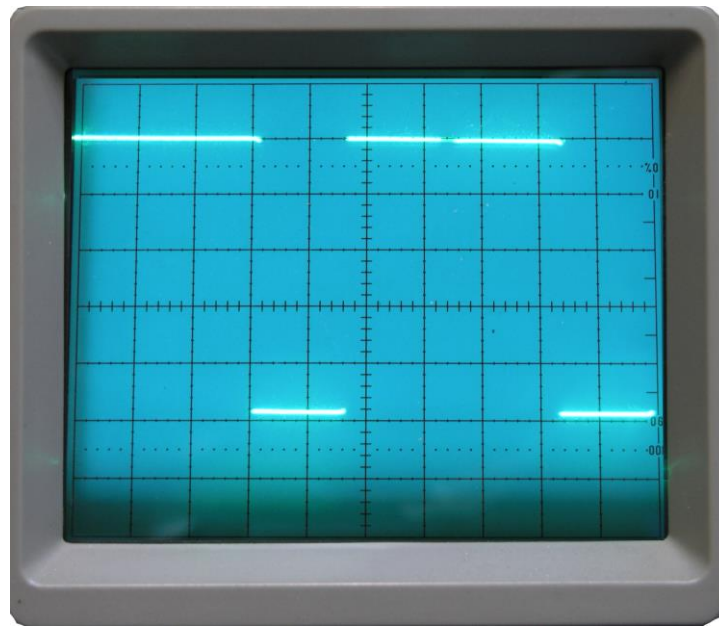
These are solid state relays that can switch up to six amps. An internal fast-acting fuse on the circuit board (F1) will blow at 4 Amps, providing ample protection to the relays and allowing them to provide +13 VDC to any reasonable alarm device such as a warning horn or light.

## U6

U6 is a digital comparator. It is configured to switch its output from zero to +5 volts when the input rises above +0.15 volts. Its purpose is to remove the noise from the Hall Effect transducer 0 to +0.6-volt signal and convert it to a clean 0 to +5 volt signal for introduction to U1 (see below.)



The Hall Effect transducer signal is noisy and too low to feed to the microcomputer.



The output of U6 is very clean with sharp rise and fall times and is raised to a 5-volt signal...

## U7

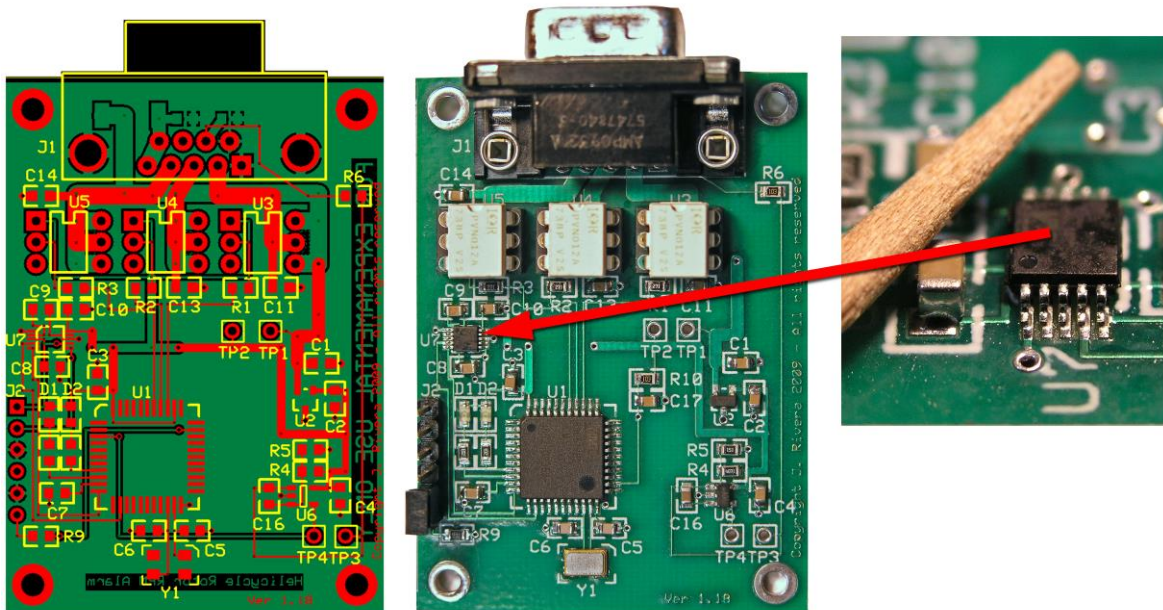
U7 does the RS-232 level conversion between U2 and the PC. It is only used during programming, debug, and test.

## U8

Power-On Reset -- Discussed earlier...

## PCB

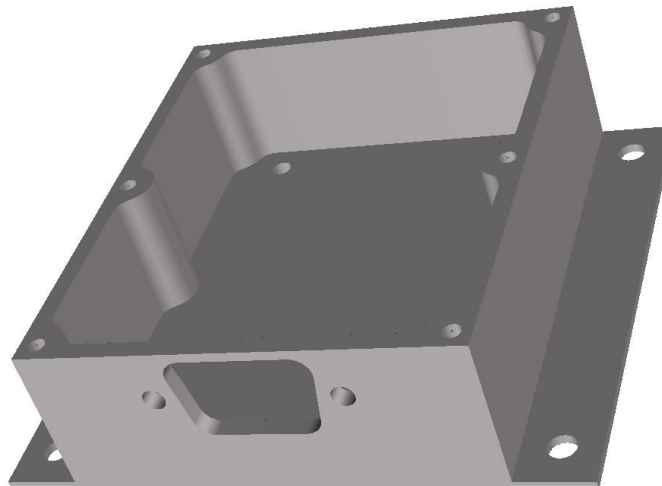
The printed circuit board is standard double-sided G-10 epoxy glass with plated-through holes, a solder mask, and a silk screen component legend. It measures 1.8” x 2.5”.



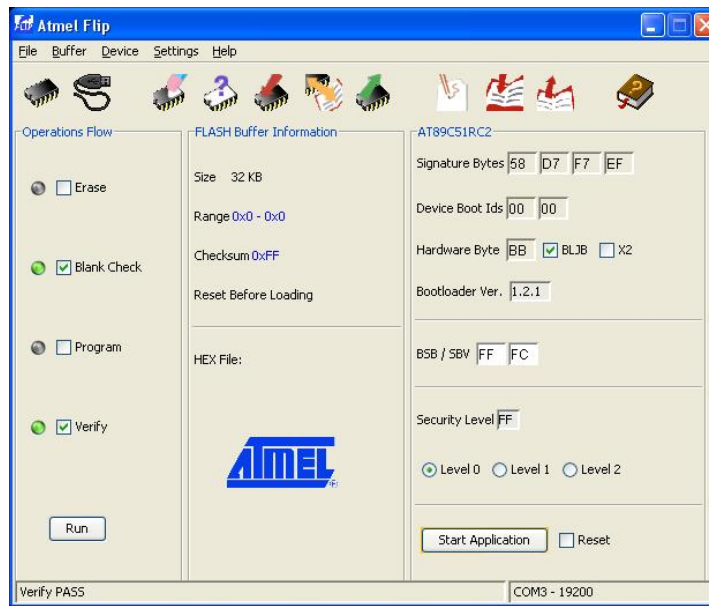
The artwork showing both the top and bottom layers is at the left and the actual board is in the middle. The detail at the right shows U7 next to a toothpick for scale. These parts must be soldered under a microscope. (This is an earlier version without U8.)

## Enclosure

The milled enclosure design is shown below. It measures 2.75” x 3.15” x .95” and will weigh 2.83 Ounces. The top will attach with six 4-40 screws.



## Programming



The Atmel Flexible In-system Programmer (FLIP) is the programming environment that is used to program U1, the microcomputer used in my circuit. The main user interface is shown above. The programming of the alarm is about to begin now that the hardware is debugged and functional. During initialization U1 will activate each alarm output in sequence for two seconds as a self test.

## Timing

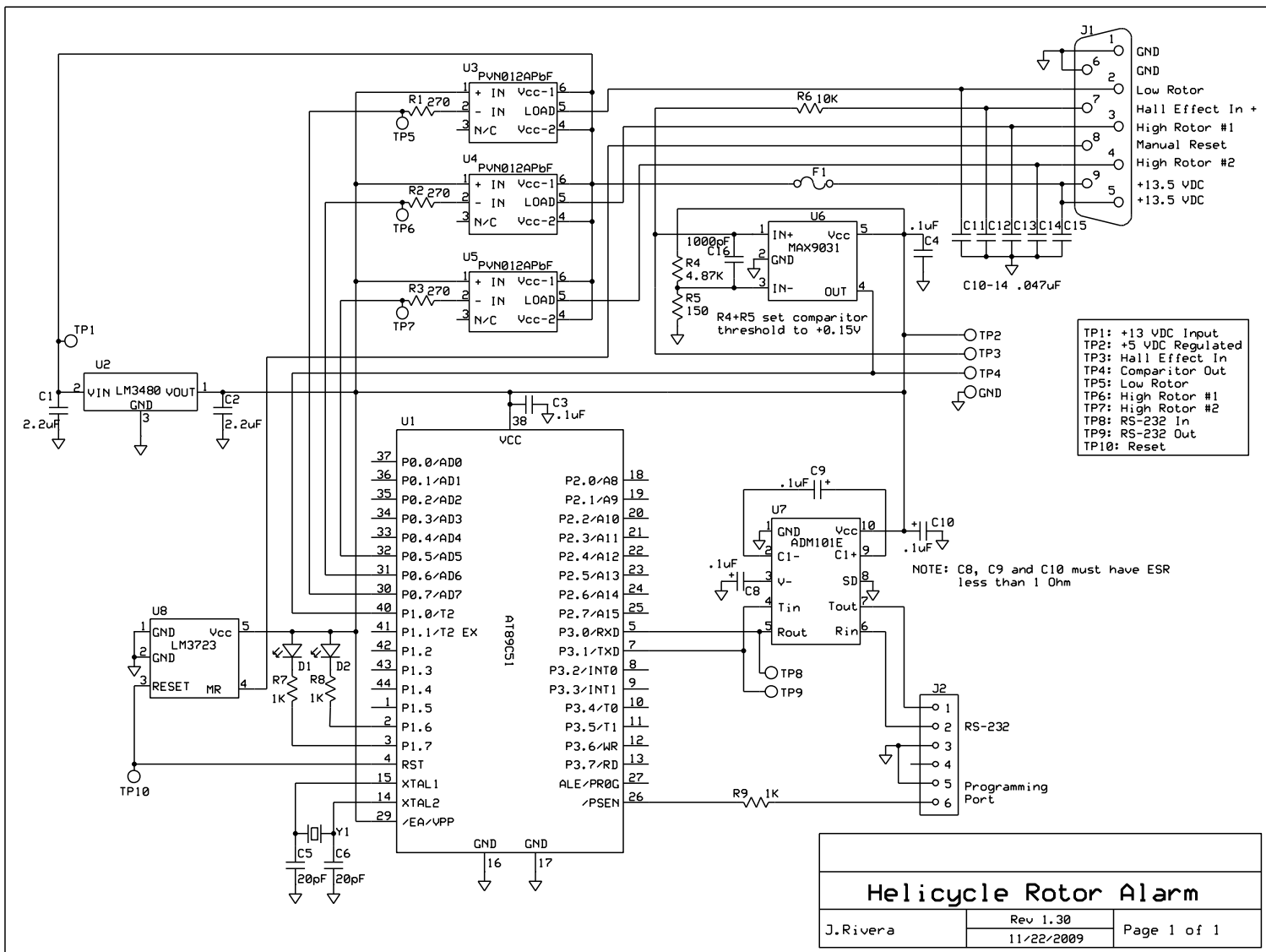
The hall effect sensor produces four pulses per revolution (one magnet every 90 degrees.) The table is based on dividing the pulses by four to produce one output pulse per revolution.

Alarm Condition	RPM Limit (RPM)	Revolutions Per Second	Duration (Msec)	100 kHz Count	100 kHz Rounded Off	Actual RPM Limit Using Rounded Off Value
Low Rotor	Less than 610	10.167	98.361	9836.066	9836	610.004
High Rotor #1	More than 620	10.333	96.774	9677.419	9677	620.027
High Rotor #2	More than 635	10.583	94.488	9448.819	9449	634.988

With a 100 kHz clock, a count exceeding 9836 will activate a low rotor alarm, a count of less than 9677 will activate High Rotor Alarm #1, and a count of less than 9449 will activate High Rotor Alarm #2.

The Rotor RPM Alarm Enable line will be pulled low from +12 VDC when the clutch is fully engaged. This will prevent false alarms during start-up.

# Helicycle Main Rotor RPM Alarm – For Experimental Use Only

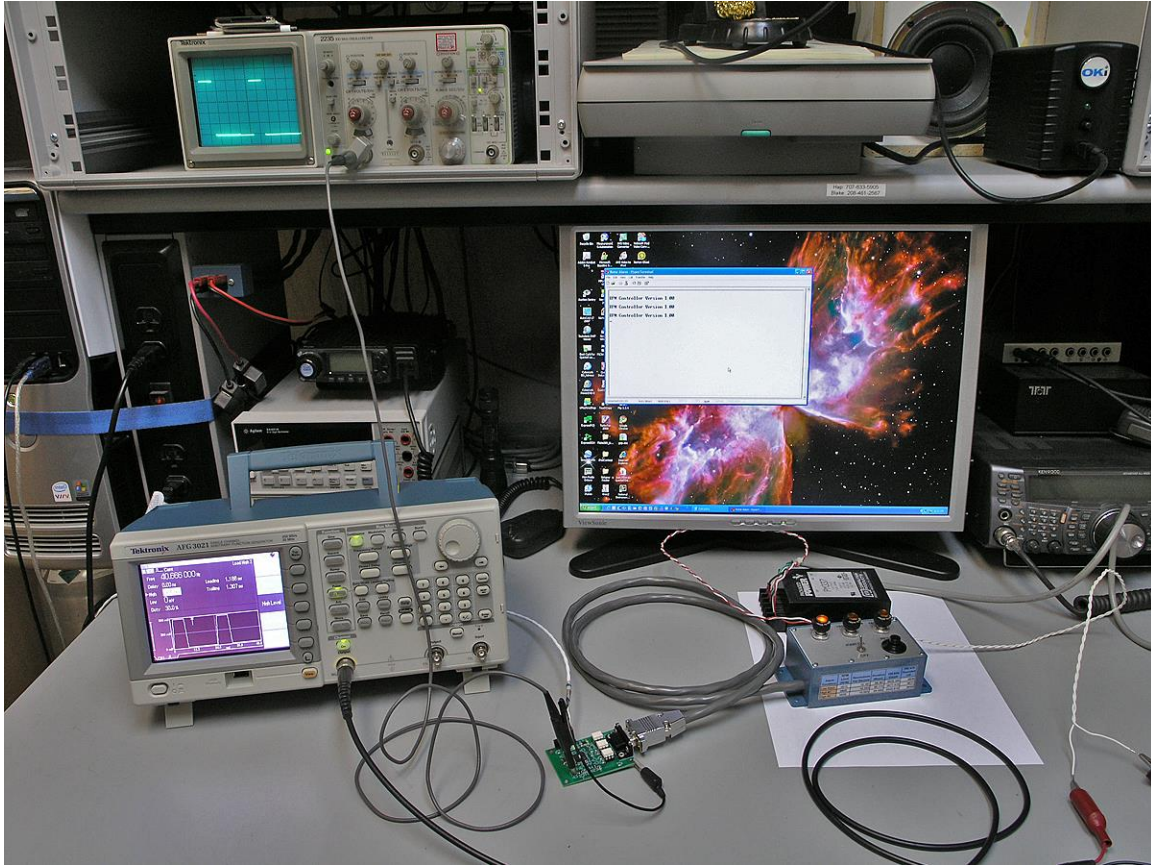




## Helicycle Main Rotor RPM Alarm – For Experimental Use Only

16 December 2009

The board is fully functional, and a first build of the software is installed. Now it's time to test and dial in the alarm points. Here's what my test setup looks like:

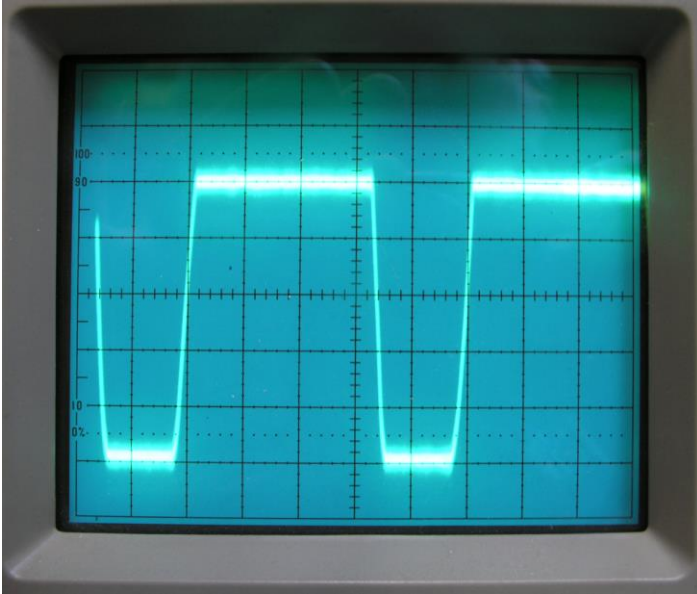


I borrowed an arbitrary waveform generator from work so I can generate very precisely timed pulses. It's the unit at the left of the picture.



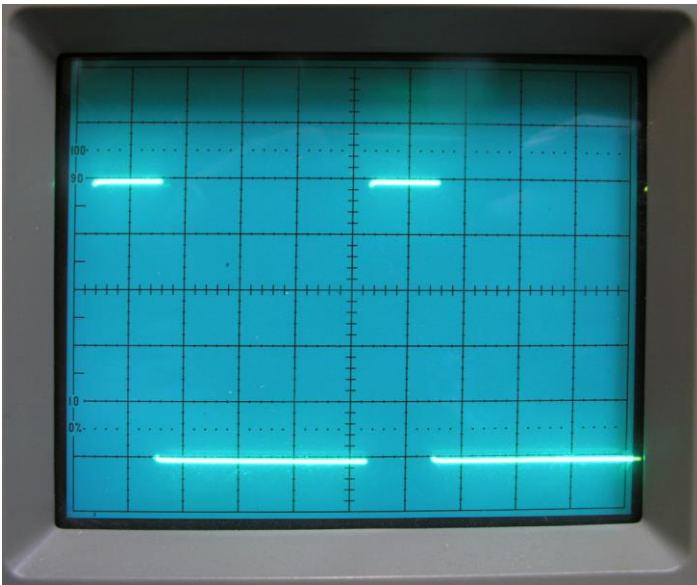
Here you can see that the frequency is set to 40.666 Hertz. That corresponds to a rotor RPM of exactly 610. I can also shape the pulses to look just like the output of the Hall Effect sensor.

## Helicycle Main Rotor RPM Alarm – For Experimental Use Only



Here's that waveform as it appears on test point 3 at the input of comparator U6 (see the schematic on page 8.)

I've sloped the edges of the pulses to make it as similar to the actual input as I can.



Here's the pulse after it's cleaned up and amplified at the output of the comparator at test point 4. Now the edges of the pulse have a very fast rise and fall time. That's necessary so the microcomputer will be able to trigger properly.

Once the trip points are programmed then the only thing left is to add in some signal averaging and test it out!

Helicycle Main Rotor RPM Alarm – For Experimental Use Only

REF DES	DESCRIPTION	MFG P/N	DIGIKEY P/N
U1	MCU FLASH 16K 44VQFP	AT89C51RB2-RLTUM	AT89C51RB2-RLTUM-ND
U2	IC VREG, 5V,100mA, SOT23-3	LM3480IM3-5.0/NOPB	LM3480IM3-5.0CT-ND
U3,U4,U5	IC RELAY, 20V, 6A, 6-DIP	PVN012APBF	PVN012APbF-ND
	CONN SOCKET,6-PIN	BU060Z-178-HT	ED2201-ND
U6	COMPARITOR, MAX9031	MAX9031AUK+T	MAX9031AUK+TCT-ND
U7	TX.RX,RS232,10MSOP	ADM101EARMZ	ADM101EARMZ-ND
U8	Power-On Reset	LM3723	LM3723EM5-4.63CT-ND
Y1	20MHz SMD XTAL,ECX-3S	ECS-200-20-30B-DU	XC1514CT-ND
D1,D2	LED,GRN,5mA,0805	LN1371SGTR	P516CT-ND
F1	FUSE,FAST,4A,125V,SMD	0459004.UR	F1171CT-ND
R1,R2,R3	270 Ohm,1/8W 5%,0805	ERJ-6GEYJ271V	P270ACT-ND
R4	4.87K,1/8W 1%,0805	ERJ-6ENF4871V	P4.87KCCT-ND
R5	150 Ohm,1/8W 5%,0805	ERJ-6GEYJ151V	P150ACT-ND
R6, R10	10K,1/8W 5%,0805	MCR10EZHJ103	RHM10KACT-ND
R7,R8,R9	1K,1/8W 5%,0805	ERJ-6GEYJ102V	P1.0KACT-ND
C1-C3, C4, C7	.1uF,CER,100V,X7R,0805	GCM21BR72A104KA37L	490-4789-1-ND
C8-10	.1uF TANT, 20V, 10%	TAJR104K020RNJ	478-3286-1-ND
C5,C6	20pF,CER,250V,0805	251R15S200JV4E	712-1381-1-ND
C11-C15	47,000pF,CER,100V,X7R,0805	GRM21BR72A473KA01L	490-3325-1-ND
C16	1000pF,CER,100V,X7R,0805	ECJ-2VC1H102J	PCC102CGCT-ND
J1	CONN,DB9 MALE, THRUHOLE	5747840-3	A32091-ND
PCB	EXPRESSPCB	N/A	N/A

## Bill of Materials