BALLISTIC CHRONOGRAPH

If you ever wanted to know the velocity of a projectile leaving your paint gun, rifle, pistol, BB gun, slingshot, or arrow then this is a project for you. This is a great project for the marksmen or hunter that loads his own shells.

By David Collins

Last year, my son bought a chronograph. I was so intrigued by his demonstration as to how it worked, that I began to gather information from the web and finally designed my own. The info obtained pertaining to IR LEDs, IR photodiodes, LCDs, EEPROMs, and RS-232 communications were most useful in the development of this project.

FEATURES

This ballistic chronograph rivals expensive units with some *very impressive features:*

- Measures velocities (ft/sec)
- Calculates the average
- Standard deviation
- Minimum and maximum velocities
- Saves data to EEPROM and displays on an LCD

There are two velocities measured for each shot as a double check confirming the measurements. The majority of chronographs on the market today only measure one velocity. Only the very expensive ($700 to $800 range) measure two velocities. This chronograph has a maximum number of shots per series of 100 and you can record a maximum of 10 series.

The *modes of operation are:*

➤ SHOOT — measures, displays, and saves the velocities of each shot.

➤ RETRIEVE — displays the shot number, series number, the two velocities of each shot along with min, max, avg, and standard deviation for the series.

➤ SAVE TO PC — saves all the data mentioned in RETRIEVE to a file on your PC via the PC serial port; use the HyperTerminial application to receive the data and then copy it to your file (computers w/o a serial port could use a serial to USB converter).

➤ DELETE A SERIES — deletes all the data in a selected series.

➤ DELETE ALL SERIES — deletes all data in the 10 series.

All the modes are accessed via two pushbutton switches (ENTER and END) located on the front panel. In SHOOT mode, the two velocities are displayed in big numbers and readable at greater than 10 feet. The complete design is powered by a battery pack of eight AA cells. The unit will display this voltage and beep if it is less than 7.5 volts.

THEORY OF OPERATION

This project has *two major parts:*

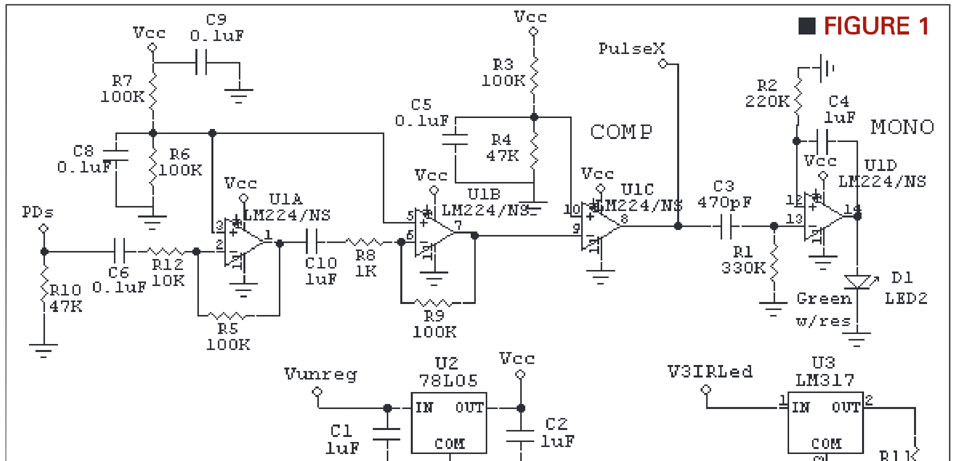
- The three screens
- The microcontroller (µC unit)

The screens consist of photodiodes and infrared LEDs. The heart of the chronograph is the photodiode whose reverse current is proportional to the amount of incident light. Therefore, if you reverse bias the photodiode and interrupt or diminish the amount of light as seen by the photodiode, the reverse current will decrease. The momentary decrease in reverse current develops a change in voltage across R10 (**Figure 1**). This change in voltage is very small (approximately 12 mV), so the op-amps U1A and U1B (LM224) amplify it and feed it to the comparator U1C, producing a TTL level trigger pulse which is sent to the µC for timing purposes. This trigger pulse is also sent to a monostable multivibrator U1D and illuminates the green LED for about one second. *Not having a single shot digital sampling scope, I added this during the development stage in order to have visual indication of a detected pulse.*

Screen: The term screen comes from the very early chronographs where they used a metallic screen material and sensed the change in resistance as the projectile passed through. The chronographs of today are mainly optical in design.

Therefore, when a projectile passes through the first screen, Pulse1 is generated which starts a timer in the µC. As it passes through the second screen (generating Pulse2), the number of timer ticks (the clock period) is saved as Ticks12. Likewise, as it passes through the third screen, Pulse3 saves the total number of timer ticks as Ticks13.

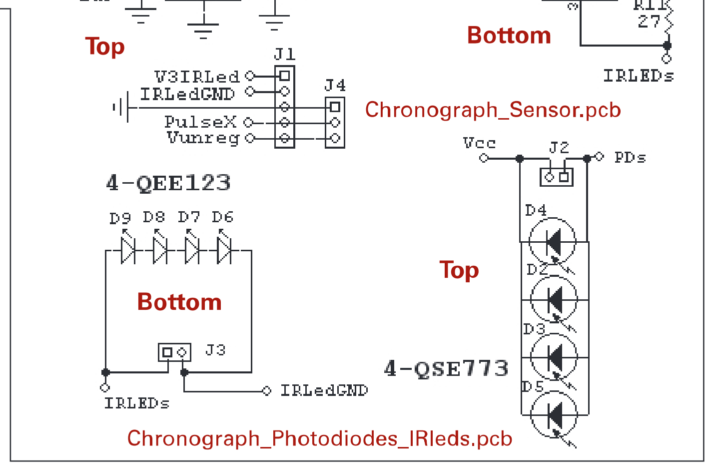
Knowing the time of each tick to be 1/Oscfreq and the distance between each screen (one foot), you can then calculate the velocities of the projectile in ft/sec as:



$$Velocity1to2 = \frac{D}{T} = \frac{1}{Ticks12 * \frac{1}{Oscfreq}} = \frac{Oscfreq}{Ticks12}$$

similarly $$Velocity1to3 = \frac{2 * Oscfreq}{Ticks13}$$

and $$Velocity2to3 = \frac{1}{(Ticks13 - Ticks12) * \frac{1}{Oscfreq}} = \frac{Oscfreq}{(Ticks13 - Ticks12)}$$



Theoretically, Velocity2to3 should be equal to or less than Velocity1to2 when the projectile has no positive horizontal acceleration. Therefore, if the two velocities do not meet these criteria, then you should be doubtful of the data. The majority of ballistic chronographs on the market rely on ambient sunlight to provide the source of illumination to the photodiodes. That's fine if you only use it outdoors on a not so cloudy day. This design is totally independent of the ambient light, relying only on the internal IRLEDs for the light source. In fact, it will operate in a totally dark environment.

The photodiodes D2-D5 used in this project are QSE773 and have peak sensitivity at a wavelength of 920 nm (infrared radiation). The IRLEDs D6-D9 used as the light source are QEE123, which has a peak emission wavelength of 950 nm. Each screen has four photodiodes flooded with four IRLEDs. A constant current generator U3 provides 46 mA to the string of IRLEDs, thus providing a constant infrared illumination of the photodiodes. This constant current is turned on/off by the µC only during the SHOOT mode thus saving a significant amount of the battery charge.

The µC U5 (**Figure 2**) performs the math operations, sends the data to the EEPROM U4, LCD U3, and/or the RS-232 drivers/receivers U1. These four ICs are powered by the five volt linear regulator U2. Note that the design also provides for a serial LCD[1] which can be connected to J7, although changes in the code would be required.

SOFTWARE

The BASCOM-AVR[2] complier version 1.11.9.2 was used to write the code (complete listing can be downloaded from **www.nutsvolts.com**) and program the µC. The current code uses about 75% of the available programmable Flash program memory so there is space remaining for your additions and/or modifications. An in-system programming connector J11 is available on the Chronograph_uC board for this purpose. **Figure 3** is the flow diagram of the present code.

All the data is stored in the 24LC65 EEPROM U4, which has a capacity of 8Kx8. These data are all defined as 16 bit words except Series_num and Shot_num and require two addresses. For that reason, one address is for the high eight bits and one for the low eight bits. The equations defining the used EEPROM addresses are:

Series_num addr = 500 + 600*(Series_num - 1) = 600*(Series_num) - 100

Shot_num addr = above+5*Shot_num - 4 = 600*(Series_num) + 5*Shot_num - 104

High(Velocity_1) = above + 1 = 600*(Series_num) + 5*Shot_num - 103

Low(Velocity_1) = above + 1 = 600*(Series_num) + 5*Shot_num - 102

High(Velocity_2) = above + 1 = 600*(Series_num) + 5*Shot_num - 101

Low(Velocity_2) = above + 1 = 600*(Series_num) + 5*Shot_num - 100

High(Velmax) = address for Series_num + 501 = 600*(Series_num) + 401