By Richard Kappmeier RichKTech@sbcglobal.net

I play paintball… well, woodsball, mostly. One of my paintball peeves is not knowing how much CO_2 is left in my tank. I'm usually in the middle of a pretty good scrap when suddenly my gas gives out and I'm eliminated with extreme prejudice!

Unfortunately, there is no way to know exactly how many shots are left in a CO_2 tank. The actual output of a tank varies due to ambient temperature and marker (gun) efficiency but a long-standing guesstimate of tank capacity is 50 shots per ounce of liquefied gas. Based on my own experience, I thought this estimate low and decided to find out just how true this fragment of conventional wisdom might be, so I built the ShotMeter!

There were some project objectives I felt were very important. The unit had to:

- Avoid interfering with operation and maintenance of the marker
- Be as light and strong as possible
- Have an liquid crystal display (LCD) to display multiple counts and warnings
- Have a microprocessor with plenty of room for hardware and software improvements should I think of them.

Overall Concept

As I thought about this project it occurred to me that I could build some extra functionality into the system outside of tracking how many shots into a tank I was. Why not estimate current loader (motorized paintball feeder) level and track overall shots for the day on the marker? Also, I could have the ShotMeter trigger the motorized loader with a simple relay!

I wanted to track my loader level because I thought it might be pretty cool to have the ShotMeter tell me when the loader is low on paintballs by beeping at me and displaying a warning on the LCD. And, I wanted to track my overall shots per day because although I may bring 2000 rounds to the field I've been known to accidentally drop quite a few on the ground during an under-fire reload. I wanted to know how many paintballs I just plain lose.

Triggering a loader would save me some money, too. A top-of-the-line loader could cost upwards of $120.00! With the loader trigger feature I could buy a cheap constant rotation loader for around $20.00 and connect a relay (Figure A, *K1*) to control it. The load rate wouldn't be very high, about 6-8 balls per second, but it would be cheap, very reliable, and have the same functionality as more expensive loaders when driven by the ShotMeter.

Nearly all electric loaders available today are triggered by either sound or infra-red beam interrupt technology. I didn't want to duplicate a current product so I went with a magnetic field detector. Rather than a mechanical switch, like in a home alarm system, I decided a Hall Effect device (see *The Hall Effect*, sidebar p.4) would be ideal: no moving parts, an exceptionally small sealed case, and a very low current consumption. The unipolar Hall-effect sensor I chose is the OH090U manufactured by OPTEK (Figure A, *Q1*). This choice also allowed me to avoid altering the marker in a significant way.

Each time the south pole of a magnetic field comes near the front of this sensor the device sinks a 5-volt signal provided by a pull-up resistor (Figure A, *R6*). The sensor/magnet configuration on the marker is such that when the bolt is in the ready to fire position (cocked back) the south pole of the magnet is directly under the downward facing sensor, forcing the signal low. When the trigger is pulled and the bolt slams forward the loss of the magnetic field at the face of the sensor releases the signal. The release of the signal generates a low-to-high interrupt on the microprocessors' (Figure A, *U1*) external interrupt pin, *RB0*. The interrupt service routine then causes the firmware to increment/decrement all of the related counts for the device, check for any level warnings, beep if necessary, update the display (Figure A, *L1*), and engage the loader relay.

I have added a new feature to the ShotMeter recently. The unit worked well the first time I went out with it but if I lost power for any reason, for instance if the loader cable disconnected, I lost all of the accumulated data. I needed a way for the unit to detect the start of a power failure and save the data before a complete power loss crashed the system. What I came up with is the use of an MCP120-475 (Figure A, *U2*) microprocessor supervisory device, made by Microchip Technologies, to detect the loss of power. As the main 5-volt power begins to drop, it crosses the 4.75 VDC tripping point for the device. When the device is tripped, it will sink the signal provided by a pull-up resistor (Figure A, *R7*) on the **LOWVOLT** line, signaling the microprocessor to save all of the present values. The tricky part of this operation is that there has to be enough power left in the circuit to allow time for the processor to save the values before it crashes. A 470uF capacitor (Figure A, *C2*) delays the power loss just long enough for the values to be saved on the microprocessors onboard EEPROM. Originally I was using a 220uF capacitor but as the battery began to wear the delay shortened and the microprocessor was only able to save two of the three required values.

How It Operates

On power-up the unit will display a title screen, beep for 0.5 seconds, agitate the loader for 0.5 seconds, and then display an option screen. The three options are: "Setup", "Clear Values", and "Start". Each of these options corresponds to a button on the unit. The "Start" button accesses the values saved just before the last power-down and begins normal field operation of the unit using those numbers. Pressing the "Clear Values" button displays a short "Values Cleared!" message and zeros the values saved in memory and returns the user to the option screen. "Setup" takes the user through a setup procedure for all of the settings that are required for the ShotMeter to work properly.

The setup procedure allows the user to:

- Set the loader capacity. My loader capacity is 170 paintballs.
- Set the "loader low" warning point. I set mine to 70.
- Set the pod capacity (a pod is reminiscent of a clip for a real fire arm). My pods hold 100 paintballs.
- Set the "tank low" warning point. Originally I set mine to the calculated estimate for the size tank I was using: 450 shots for my 9oz tanks and 600 shots for my 12oz tanks.
- Enable/Disable the "hopper low" warning beep. Enabled for me.

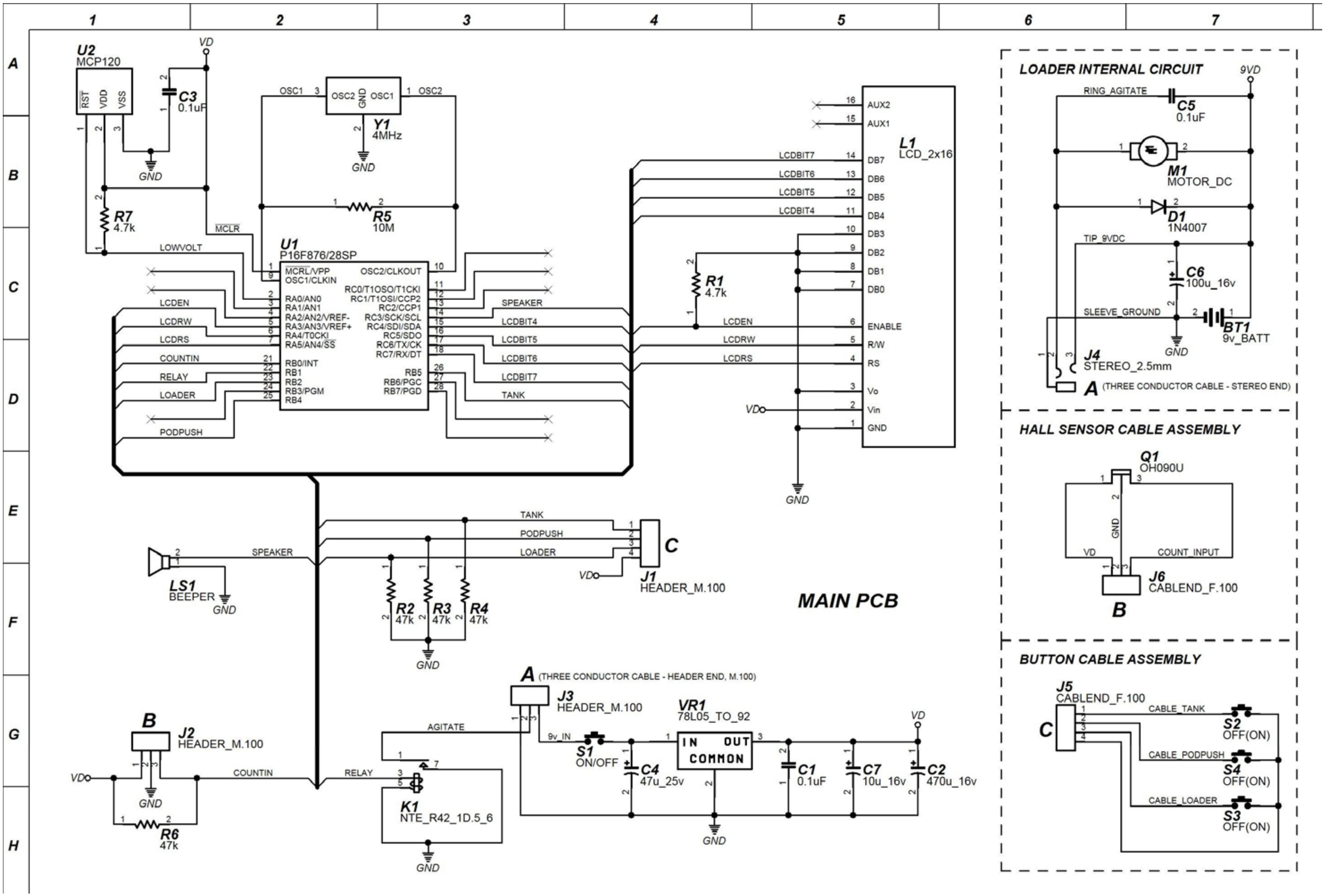


Figure A. This is a schematic of the entire ShotMeter circuit. The elements in the dashed boxes are off-PCB sub circuits: the loader internals, the sensor assembly, and the button assembly. Note that the capital letters are corresponding connection points between the main PCB and the sub circuits

- Enable/Disable the “tank low” warning beep. Disabled for me.
- Enable/Disable loader agitation. Enabled for me.
- Set the relay pulse length. Mine is set to 7 units or 175 iterations of the main loop which is a little less than a second. The warning beep is hard coded to be 200 iterations longer than the relay pulse length to avoid the beep being drowned out by the loader motor.

The basic operation of the unit is this:

- Each time the marker is fired the Loader count (Figure B) is decremented, the Tank and Shots/Day counts are incremented, and loader agitation is engaged.
- The unit then checks these numbers versus the loader and tank warning points
- If the loader count is below the warning level the unit beeps and displays an “L” on the left of the LCDs top line
- If the tank is low the unit beeps and displays a “T” on the right of the LCDs top line

What the operator can do when a warning occurs:

- When a loader warning occurs the operator can do one of two things
  - Load a pod of paintballs into the loader and press the Pod button (Figure B, *P*). Pressing the pod button adds one pod of balls to the loader count and resets both the audio and LCD loader warnings.
  - Top off the hopper with paintballs and press the Loader button (Figure B, *L*). Pressing the loader button sets the loader count equal to the pre-defined loader capacity and resets both the audio and LCD loader warnings.
- When a tank warning occurs the operator will replace the tank when possible and then press the Tank button (Figure B, *T*). Pressing the tank button zeros the tank count and resets both the audio and LCD tank warnings.

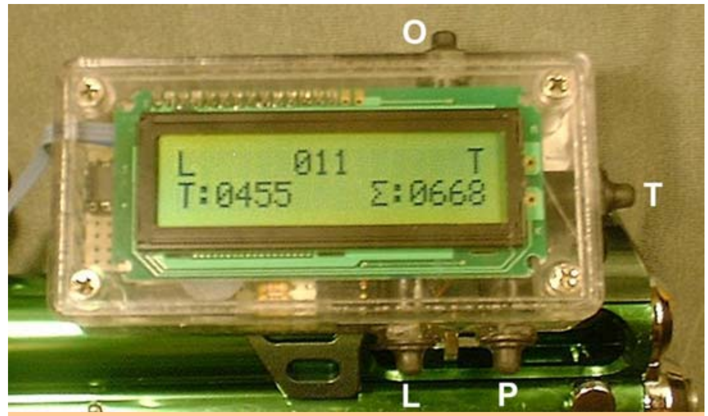


Figure B. This is the 16x2 display in the standard tracking mode. Top line (left to right): loader low warning “L”, loader level count “011”, and tank low warning “T.” Bottom line (left to right): shots used from tank “T:0455” and shots for the day “Σ:0668” totals. The button on top of the ShotMeter is the on/off switch (O), the button on the right face is the Tank (T) button, the left button on the bottom face is the Loader button (L), and the right button on the bottom face is the Pod button (P).

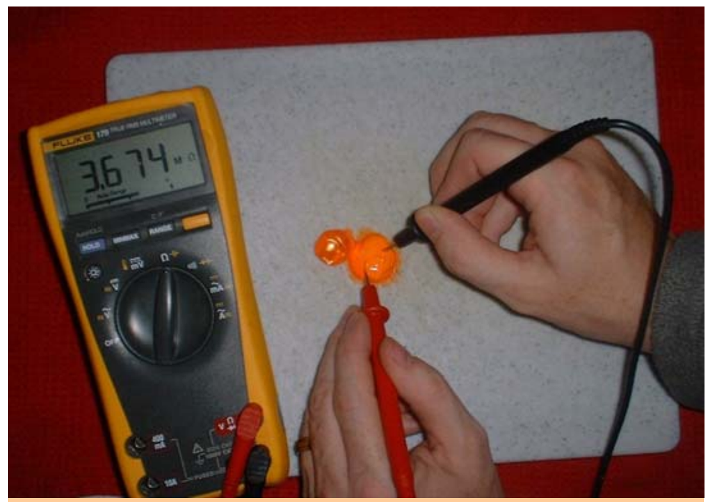
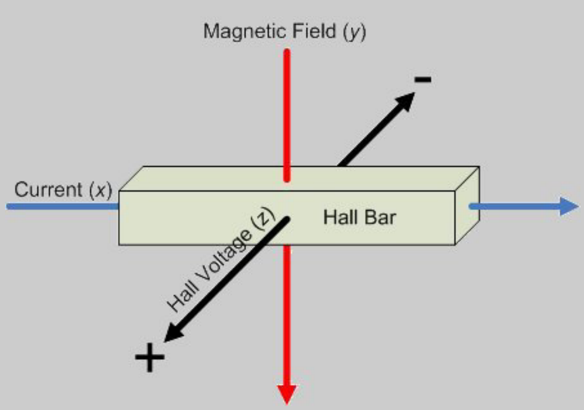


Figure C. Measuring the resistance of a glob of paint.



The Hall Effect

The Hall Effect occurs when a current (*x*) is passed through a thin sheet of conducting material, a Hall Bar (or van der Pauw element), and a magnetic field (*y*) is applied perpendicular to the current path. A potential difference, or Hall Voltage (*z*), is created across the conductor at right angles to both the current path and the magnetic field.

What many Hall Effect sensors do is amplify the Hall Voltage and then pass that amplified voltage through a comparator to provide the digital output signal.

Hall Effect Applications

The most common application for a Hall-Effect sensor is binary position sensing. There are also analog Hall sensors that can be used for absolute position sensing though these are somewhat less widespread. The advantage to using a Hall sensor is the fact that there are no moving parts within the sensor. This allows an extremely high life cycle for the sensor in comparison to a mechanical sensor, such as a magnetic reed switch.

Industrial applications include safety lockout systems for large machinery or limit switch type applications for CNC machinery. In robotics, these sensors are commonly used as rotation sensors in gear trains or wheels. Placing a magnet, or group of magnets, within a robots wheel and then positioning the Hall sensor appropriately will yield a very practical rotation sensor.

Figure D. The underside of the sight rail.

Constructing the ShotMeter

As you may imagine, the paintball field is not an ideal place for electronics. There's plenty of dirt and lots of physical shock from running and dodging but the main hazard to field electronics is the paint itself.

The paint is water-based and I felt that could be detrimental to proper field operation. The problem was that I didn't know exactly how dangerous it might be. After breaking open several paintballs from different manufacturers onto a plastic cutting board (Figure C) I measured the resistance across each glob of paint. The globs were all roughly 1 inch across and between 0.13" and 0.5" high. The resistance of the globs measured between 3.69 and 16.37 MΩ. Though resistances that high probably won't cause a fatal smoking short it could make trouble for digital data signals and capacitors. I found it interesting that the thickness of the paint seemed to be directly proportional to the resistance exhibited.

Paintballs, which usually travel at between 250 and 300 feet per second, tend to shoot their insides through remarkably small gaps in things when they strike. I found rubber covers (boots) for my buttons and housed the unit in a transparent ABS plastic enclosure. I filed a shallow groove for the wire access in the enclosure lid and body with a fine finishing file. The ribbon cables are lightly pinched between the enclosure lid and body. The holes for the buttons were drilled on a standard drill press using a 0.25" bit. I couldn't find a through-hole mounted piezo beeper small enough to fit inside so I affixed a surface mount beeper (Figure A, *LS1*) to the wall of the enclosure with double-sided tape and ran wires from the beeper to the microprocessor.

The Hall-Effect sensor is set in a pair of shallow overlapping holes (0.25" diameter by less than 0.25" deep) I drilled into the bottom of the sight rail. I then covered the sensor with hot glue (Figure D) to keep it in place and ward off any moisture. I ran the sensor wires under the site rail and up the front face and then into the ShotMeter. The completed assembly (Figure E) is mounted on the sight-rail via a home-made aluminum bracket.

Figure E. The marker with the electric loader on top and the ShotMeter mounted on the sight rail between the grip and the loader.

I used a "super strong" neodymium magnet I happened to have stuck on my refrigerator to activate the sensor. I have no idea where I got this specific magnet but I have seen similar ones at craft and hardware stores. The magnet is slightly less than 0.25" in diameter and 0.13" high and fits conveniently into the top of a hole bored through the aluminum bolt (Figure F). The hole is mostly filled with the upper half of a steel pin, the bolt pin, that sticks out the bottom of the bolt. The top 1/4 of this hole is empty which allows the magnet to fit snuggly within. Also, the steel of the pin provides an excellent magnetic bond which keeps the magnet in place during those violent bolt cycles; no adhesive necessary! Further, because the marker body is also aluminum the magnet isn't attracted to it and the magnetic field passes through it with little degradation.

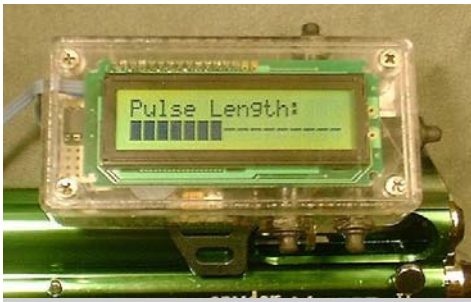
Modifying the Loader

The loader I chose to use is a Ricochet Rhino (Figure E, topmost part of the marker) which I picked up for $22.00 on-line. I chose this loader for two reasons: it's very inexpensive and it has a great look to it.

Figure F. Two aluminum bolt examples: the top bolt showing the Neodymium magnet installed and the bottom bolt showing the hole where the upper surface of the steel bolt pin is visible.

What I did was use a 2.5mm stereo (Figure A, *J4*) headphone jack as the connector from the ShotMeter. I drilled a 0.25" counter-sunk hole through the bottom of the motor compartment.

I disconnected the battery from the loaders original control circuit by cutting the ground trace to the rest of the nearly unpopulated printed circuit board (PCB). Then I soldered wires to the edge of the battery contacts on the original PCB and ran those wires to the headphone jack; positive to the tip and ground to the sleeve.



A Simple LCD Bar Graph

This code snippet describes a method of creating a simple bar graph display on a Hitachi based Ampire AC162BYA53 LCD display. A display address that is loaded with 255 shows a solid 5x7 block of pixels. For example, if the pulse length desired were 7, this subroutine will assign blocks to display addresses 0-6 and dashes to addresses 7-F of the second line of the display. Please note that I have, in addition to making this snippet more pseudo-code, also made it more linear in nature by re-integrating some subroutines. For the original version please consult the downloadable MBasic file on this articles Warpig.com page.

CODE	COMMENTS
BEEPLENGTH:	; section - subroutine initialization
Lcdwrite , line1 = "Pulse Length:"	; write "Pulse Length:" to the first line of the LCD
OBJECT = KONFIG.nib1	; copy high nibble of KONFIG variable into OBJECT byte variable.
	; The high nibble (bits 4-7) of the KONFIG byte is where the pulse
	; length is stored after it is read from the on-board eeprom
	; OBJECT is a word sized temporary variable
INDEX = 0	; copy 0 into INDEX byte variable
BEEPBAR:	; section - load characters into BARGRAPH 16-byte array
if INDEX <= OBJECT then	; when INDEX <= OBJECT copy 255 into byte numbered INDEX
BARGRAPH(INDEX) = 255	; of the BARGRAPH array. Assigns "blocks" to pulse count.
else	; when INDEX > OBJECT copy "-" into byte numbered INDEX
BARGRAPH(INDEX) = "-"	; of the BARGRAPH array. Assigns "-" to digits > pulse count
endif	; end of if…then
INDEX = INDEX + 1	; increment INDEX
if INDEX < 16 then BEEPBAR	; when INDEX = 16 drop through to next section, otherwise
	; loop to BEEPBAR
INDEX = 0	; copy 0 to INDEX
BEEPDISP:	; section - display bar graph on line2
lcdwrite, line2 = [BARGRAPH \ 16]	; write all 16 bytes of BARGRAPH array to line2 of LCD
BEEPBUTT:	; section - button loop
pause 200	; debounce pause for buttons, 200ms
if PODBUTT = 1 then BEEPPLUS	; when POD button pressed branch to BEEPPLUS
if LOADERBUTT = 1 then BEEPMINUS	; when LOADER button pressed branch to BEEPMINUS
if TANKBUTT = 1 then BEEPEXIT	; when TANK button pressed branch to BEEPEXIT
goto BEEPBUTT	; loop
BEEPPLUS:	; section - increment OBJECT byte variable
OBJECT = OBJECT + 1	; increment OBJECT
if OBJECT > 15 then	; when OBJECT > 15
OBJECT = 0	; wrap around to 0
endif	; end of if…then
goto BEEPBAR	
BEEPMINUS:	; section - decrement OBJECT byte variable
OBJECT = OBJECT – 1	; decrement OBJECT
if OBJECT > 15 then	; when OBJECT > 15
OBJECT = 15	; wrap around to 15
endif	; end of if…then
goto BEEPBAR	
BEEPEXIT:	; section - store OBJECT in high nibble of KONFIG byte
KONFIG.nib1 = OBJECT	; copy OBJECT to high nibble of KONFIG byte
END OF CODE	

Next, I cut the connector from the end of the motor leads and connected the red wire to the battery positive contact and the black wire to the ring contact of the headphone jack. The relay in the ShotMeter grounds this ring contact wire to make the motor rotate. Also, I added a noise filtering cap and a clipping diode (Figure A, *C5, D1*) to the motor to mitigate any possible noise or back EMF issues.

The Firmware

When I started the project I tried to use a PIC 16F628 I had in my parts box. Unfortunately, there was barely enough program space to house the minimum features I wanted and no room for hardware expansion. I then upgraded to the PIC16F876 processor by Microchip Technologies because it has plenty of program space, more I/Os than I need, and an internal EEPROM I could save settings on. The program itself currently requires only about half of the total memory available.

I listed a bit of code I thought might help some fellow tinkerers out there with their own projects. The code (*A Simple LCD Bar Graph*, sidebar this page) is a way to show a bar-graph style display on the second line of an LCD. This program listing was originally written in MBasic Pro from Basic Micro but I've stripped out the MBasic specific parts to make it more "pseudo-code."

The complete MBasic Pro file, which could be a good blueprint for those interested in creating their own ShotMeter, as well as the PIC .hex file can be downloaded from the links provided by Warpig.com. Please take note that the line names on the schematic match variables or constants in the program; e.g. the **LOWVOLT** line on the schematic corresponds to the **LOWVOLT** i/o constant in the program.

ShotMeter Limitations

The ShotMeter is only as accurate as the operator makes it. I for one will not be ensuring that every pod has exactly 100 paintballs in it or the loader is always topped off with exactly 170 balls. The loader level and the shots/day ball counts will continually be best-guesses as these are indirect counts; the ShotMeter counts cycles of the bolt whether there is a ball in the breach or not. The tank count will be accurate as it is based on the bolt cycles themselves and meant to measure gas used rather than paintballs.

The Data

After comparing the shots per day count to the number of paintballs I brought to the field I found that when I do lose balls I usually drop 100-300 a day.

In regards to the "50 shots per ounce" estimate, it seems that this approximation *is* notably low. The data in the ShotMeter Data chart (Figure G) was collected over 3 days of play with temperatures between 75° and 98° Fahrenheit. The calculated average based on the collected data is 86 shots per ounce! More than half again as many shots per ounce than the long standing estimate had indicated!

Figure G

ShotMeter Data

Tank Capacity in Ounces	Shot Estimate	Shot Actual	Shots/Ounce CO2
12	600	1068	89.00
12	600	1078	89.83
9	450	706	78.44
9	450	786	87.33
		Average Shots/Ounce CO2:	**86.15**

Next Steps

As the current version of the circuit is a hand soldered proto-board my next step may be to port everything over to a smaller surface mount design and free-up some room inside the enclosure for another upgrade or two; perhaps a digital compass or ballistic trajectory calculator? Or, I may just try to make it as small and light as possible.

I've found the "ammo low" warning to be very helpful on the field and knowing approximately how far into a tank I am has been very handy, as well. I hope you have found this project interesting and useful. Please, contact me with any questions or comments at the address listed in the byline at the beginning of this article.

ShotMeter Parts List

Qty	References	Value	Mouser Part Number	Each
3	C1,C3,C5	0.1uF	594-K104K15X7RF53L2	$0.17
1	R5	10M	594-5053DM10M00J	$0.19
1	C7	10u_16v	594-2222-030-26	$0.45
1	D1	1N4007	512-1N4007	$0.05
1	R1,R7	4.7k	271-4.7K-RC	$0.09
4	R2,R3,R4,R6	47k	271-47K-RC	$0.09
1	C4	47u_25v	594-2222-021-26479	$0.21
1	Y1	4MHz	815-AWCR-4.00MD	$0.38
1	VR1	78L05_TO_92	511-L78L05ACZ	$0.54
1	BT1	9v_BATT	525-522	$3.10
1	LS1	BEEPER	539-ASI301	$4.85
1	J5	CABLEND_F.100, 4 COND.	HAND MADE ASSEMBLY	N/A
1	J6	CABLEND_F.100, 3 COND.	HAND MADE ASSEMBLY	N/A
1	J1	HEADER_M.100, 4 COND.	571-640453-4	$0.27
2	J2,J3	HEADER_M.100, 3 COND.	571-640453-3	$0.22
1	L1	LCD_2x16	HDM16216L-5-L30S	$14.32
1	M1	MOTOR_DC	INTEGRAL TO LOADER	N/A
1	K1	NTE R4211D.5-6	526-R42-1D.5-6	$4.93
3	S2,S3,S4	SWITCH OFF(ON), MOMENTARY	506-MPA103C	$3.10
1	Q1	HALL SENSOR, OH090U	828-OH090U	$2.16
1	S1	SWITCH, ON/OFF, LATCHING	506-MPA103D	$5.80
1	U1	P16F876/28SP	579-PIC16F876-20/SP	$5.44
1	J4	STEREO_2.5mm	171-3325-EX	$1.16
3	N/A	BOOT SEAL, BLACK	506-BPA14400	$3.43
1	N/A	CLEAR ENCLOSURE	546-1591AC	$1.39
1	U2	MCP120-475	579-MCP120-475DI/TO	$0.35
1	C2	470uF_16v	75-94SP827X0004FBP	$2.46
1	C6	100uF_16v	75-94SA107X0016EBP	$2.28