EDITED BY MARTIN ROWE
AND FRAN GRANVILLE

designideas

READERS SOLVE DESIGN PROBLEMS

Microcontroller converts digital-temperature-sensor readings without floating-point arithmetic

Jordan Dimitrov, Toronto, ON, Canada

Digital temperature sensors combine a sensor, an ADC, and a serial interface in a single chip. They feature wide enough measurement range, good accuracy and resolution, no need of external parts, easy interface to microcontrollers, small size, and low price. In a review of 10 digital sensors from seven companies, all parts deliver signed-number data in two's-complement format. They feature temperature ranges of −25 to +85 or −40 to +125°C, accuracy of 0.5 to 2 or 2 to 4°C, and output data of 9 to 13 bits with 0.5 to 0.0312°C resolution. The devices' conversion time is 26 to 750 msec, and they use an SPI (serial-peripheral interface), an I²C (inter-integrated-circuit), or a 1-Wire interface. Power supplies are 1.5 to 3.6 or 3 to 5.5V, and prices range from 80 cents to $2.10 (1000).

These sensors connect to microcontrollers; hence, size, speed, and time to develop firmware are also important. The standard approach is to use a high-level language and a compiler. Development time is short, and performing even complex calculations is not a problem. However, compilers produce machine code that occupies more memory and runs at lower speed than code from an assembler. Also, compiler IDEs (integrated development environments) cost hundreds of dollars, whereas many companies offer free assembly-language IDEs. If you work on a tight budget or memory-space allotment, assembly language is the better option. The problem is to find a simple way to avoid the necessary floating-point calculations to convert sensor data into human-understandable format, both in Celsius and Fahrenheit. This Design Idea presents an effective approach.

Consider the TMP121 sensor from Texas Instruments (www.ti.com). It provides 13-bit data in a 16-bit frame with resolution of 0.0625°C/bit. Hence, the transfer function is $t_C=0.0625\times N_S$, where t_C is the temperature in degrees Celsius and N_S is the sensor data after you remove the three meaningless least-significant bits. You can easily rearrange the above **equation** to:

$$t_C=\frac{5}{80}\times N_S=\left(\frac{N_S}{2}+\frac{N_S}{8}\right)\times\frac{1}{10}. \qquad (1)$$

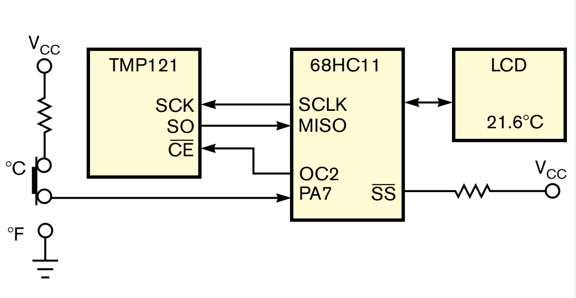


Figure 1 A small system uses a 68HC11 microcontroller to read a switch and a sensor, to convert data, and to display temperature.

DIs Inside

▶To see all of *EDN*'s Design Ideas, visit www.edn.com/designideas.

To get readings in degrees Fahrenheit, use the following **equation**, which converts degrees Celsius into degrees Fahrenheit: $t_F=[(9/5)\times t_C+32]$. Replacing t_C from the above **equations** yields:

$$t_F=\frac{9}{5}\times\frac{5}{80}\times N_S+32=\left(N_S+\frac{N_S}{8}+320\right)\times\frac{1}{10}. \qquad (2)$$

The benefit of **equations 1** and **2** is that you can perform the calculations with integer arithmetic only. They require divisions by powers of two, which you can replace with shifts, and division by 10, which you perform by introducing a decimal point in the display.

The circuit underwent testing with the popular 68HC11 microcontroller from Motorola (www.motorola.com, **Figure 1**). Besides a sensor and a controller, it includes a unit-selection switch and a dot-matrix LCD. The display resolution is 0.1°. The core of the supporting firmware is an endless loop in which the 68HC11 uses an output-compare function to generate a square-wave signal with a period of 1 sec and a duty cycle of 50%. The OC2 signal connects to the $\overline{CE}$ input of the sensor and controls its operation: When $\overline{CE}$ is high, the sensor measures temperature. The HC11 does nothing except display M on the LCD. When $\overline{CE}$ becomes low, the last measurement latches in a shift register inside the sensor. The HC11 deletes M from the display, reads the switch and the sensor, manipulates the data, and displays the temperature.

Equations 1 and **2** provide the basis for two source codes. **Listing 1**, available at www.edn.com/090305dia, generates machine code of 981 bytes. **Listing 2**, also available at www.edn.com/090305dia, generates machine code of 392 bytes. Despite the C-language approach with integer arithmetic, it needs 2.5 times more memory to do the job. The ratio is well above 10 if the C code goes with equations that need floating-point arithmetic. The benefit is clear: Modified **equations 1** and **2** and assembly-language programming let you select a microcontroller with less memory and reduce the price of your design.EDN

Discrete-component buck converter drives HB LEDs

Dhananjay V Gadre, Netaji Subhas Institute of Technology, New Delhi, India

HB (high-brightness) LEDs require a large amount of current to operate. When driving HB LEDs from a voltage source, you can set the required current with a suitable series resistor. If the voltage source is a battery, then, as the battery drains, the LED's intensity decreases. Also, a series resistance has the disadvantage of power loss through the resistor. A better option is to use a suitable dc/dc converter. If the LED's turn-on voltage is lower than the battery voltage, as would be the case with a 6V sealed-lead-acid battery, then you can use a buck converter (**references 1** and **2**). You can build a simple buck converter using only discrete components. It requires two bipolar transistors, a P-channel MOSFET, an inductor, a Schottky diode, and a few resistors (**Figure 1**).

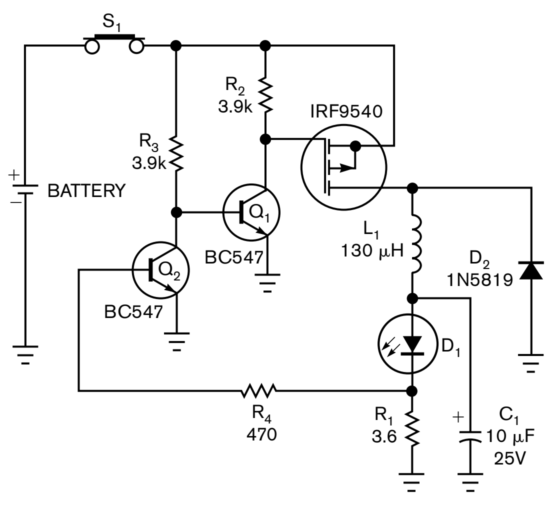


Figure 1 A buck converter provides current sufficient to drive a high-brightness LED.

When you switch on the battery voltage, the voltage across R_1, the resistor in series with the HB LED, is 0V. Thus, transistor Q_2 is off, and Q_1 is in saturation. The saturated state of Q_1 switches on the MOSFET, thereby applying the battery voltage to the LED through the inductor. As the current through resistor R_1 increases, it turns on Q_2, which turns off Q_1 and thus turns off the MOSFET. During the MOSFET's off state, the inductor continues to supply current to the LED through Schottky diode D_2. The HB LED is a 1W, white Lumiled (www.philipslumileds.com) LED. Resistor R_1 helps control the LED's

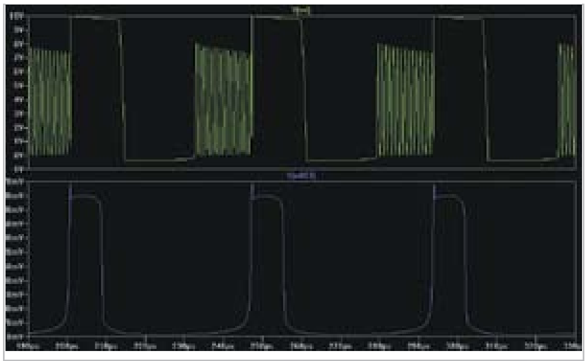


Figure 2 In a SwitchCAD simulation, the upper trace is the MOSFET-drain voltage, and the lower trace is the base voltage of Q_1.

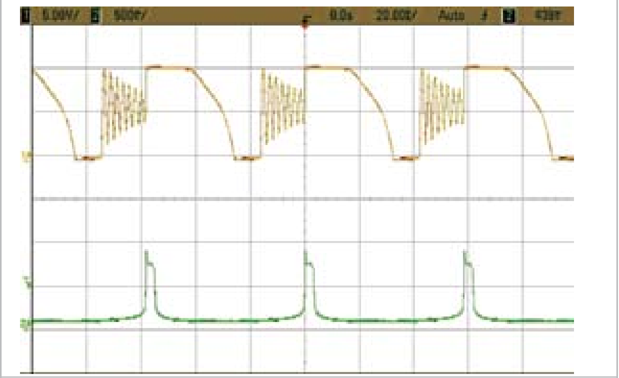


Figure 3 In an oscilloscope screenshot, the upper trace is the MOSFET-drain voltage; the lower trace is Q_1's base voltage.

intensity. Using a larger value for R_1 reduces the intensity.

The SwitchCAD-III software, which is available as a free download from Linear Technology (www.linear.com), simulated the circuit; the simulated MOSFET was an International Rectifier (www.irf.com) IRF9Z24S instead of an IRF9540 because the model for IRF9540 is not available in SwitchCAD-III. **Figure 2** plots the MOSFET-drain waveform and the voltage at Q_1's base. The circuit was wired on a prototyping board and tested for various supply voltages. **Figure 3** shows oscilloscope screenshots for the MOSFET-drain voltage and the voltage at the base of Q_1. They fairly well match the simulated waveform. Conversion efficiencies were 60 to 95% for supply voltages of 6 to 10V.**EDN**

REFERENCES

1 Saab, Alfredo H, and Steve Logan, "High-power LED drivers require no external switches," *EDN*, July 19, 2007, pg 78, www.edn.com/article/CA6459061.

2 Gadre, Dhananjay V, "Buck regulator controls white LED with optical feedback," *EDN*, Oct 25, 2007, pg 72, www.edn.com/article/CA6491146.

Drive a single-coil latching relay without an H-bridge circuit

Carlos Cossio, Santander, Spain

Single-coil latching relays find use in signal-routing, audio, and automotive systems. To maximize their usefulness and cut power consumption, these coil currents must flow in both directions. Current flowing from the latching relay's positive pin to the negative pin causes it to latch in its reset position. Current flowing from the negative pin to the positive pin latches the relay in its set position. The relay maintains its position even when you remove the coil current, which saves power after the relay latches.

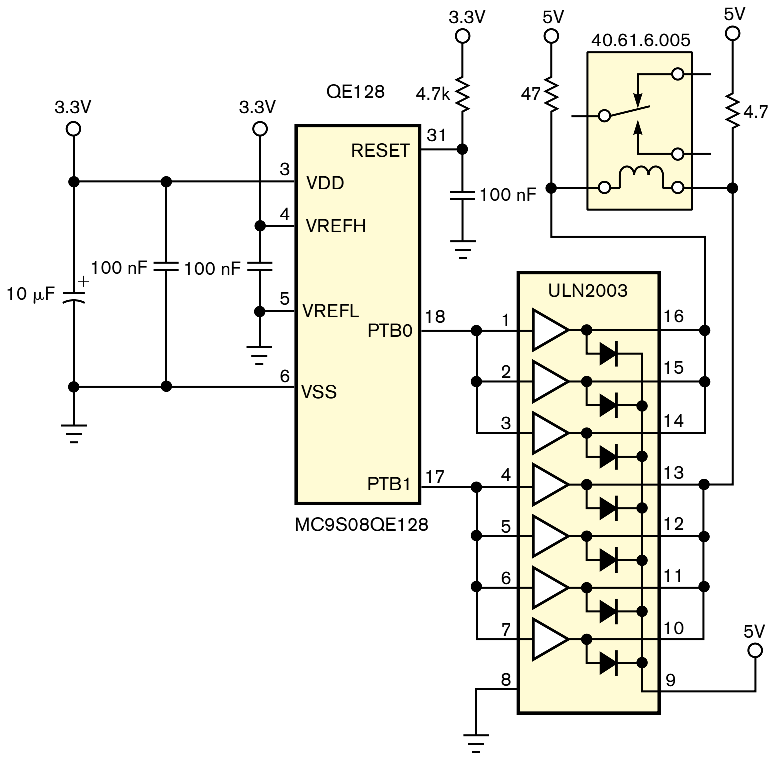


Figure 1 You can drive a single-coil latching relay without an H-bridge circuit, greatly simplifying hardware design and making the most of the low-power-consumption features inherent to latching relays in portable-system applications.

Latching relays have advantages over classic relays because, as soon as the relay switches, it remains in that position without consuming energy. Thus, no current consumption means less heat, smaller heat sinks, and a dramatic increase in battery life for portable devices. In some cases, the use of a latching relay lets your greatly simplify a circuit.

Although latching relays boast significant advantages over classic relays, their use appears limited to niche applications because they require more attention to design details. In general, a latching relay's drive circuit is slightly more complex than that of a classic relay. The traditional approach to driving a latching relay is to use an H-bridge circuit, which can be costly and difficult to handle. In addition, you must design a demagnetization circuit using a special resistor to limit the current in compliance with the manufacturer's specifications.

Figure 1 shows a simple circuit using the MC9S08QE128 microcontroller from Freescale (www.freescale.com) to drive a Finder (www.findernet.com) 40.61.6.005 single-coil latching relay with a standard ULN2003 Darlington driver with open-drain outputs and inductive-kickback protection. Clamping diodes on each ULN2003 output pin catch high-voltage transients that occur when you interrupt the coil current. Because demagnetization uses low-value resistors, you must wire at least two open-drain buffers of the ULN2003 to both endings of the relay coil to ensure enough current when the microcontroller pulls down.

Listing 1, which is available in the Web version of this Design Idea at www.edn.com/090305dib, shows the software procedure to latch the relay to its set or reset position by turning on the corresponding microcontroller output for at least 50 msec. The current flows into the ULN2003 open-drain output and latches the relay to its set or reset position, according to the direction of the coil current. As soon as the relay latches, drive the corresponding microcontroller output low to turn off the ULN2003 open-drain buffer to ensure the lowest power consumption. You must, however, take into account the set/reset timing. Pull the microcontroller output low only after the required time has elapsed. Waiting ensures that the relay will properly latch to its intended position.**EDN**

Limit switches control dc-motor H bridge

Andreas Grün, Wedemark, Germany

You use limit switches to switch off a motor if it reaches one of its two end positions. Even if you build a microprocessor-based motor controller, you should switch off a motor with hardware by building a safety interlock. Such a circuit works by inhibiting movement in one direction but allowing movement in the other direction when the motor retracts from its end position. **Figure 1** shows the circuit with mechanical switches. However, this ancient mechanical approach may be unsuitable in some cases because the motor cur-

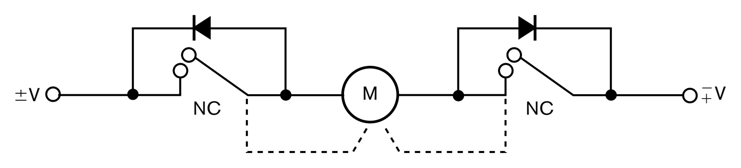


Figure 1 This circuit works by inhibiting movement in one direction but allowing movement in the other direction when the motor retracts from its end position.

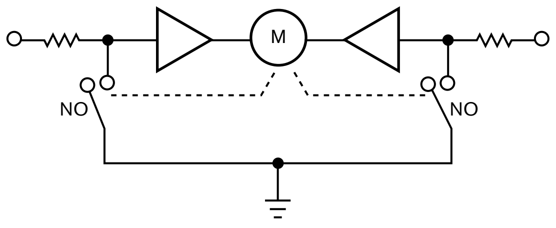


Figure 2 This circuit shortens one input of the H bridge to ground so that movement is possible only in the other direction by turning on the other input.

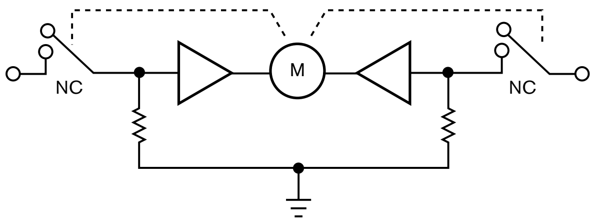


Figure 3 This circuit interrupts the connection to the driving circuit of one input and sets the input to low using a pull-down resistor.

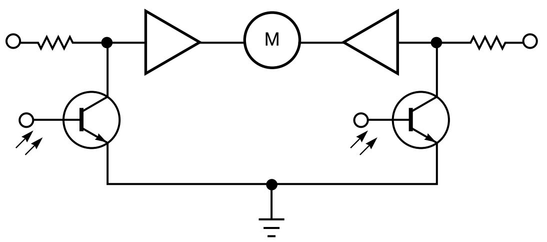


Figure 4 This circuit is the same as that in Figure 3, and it works with phototransistors without modification.

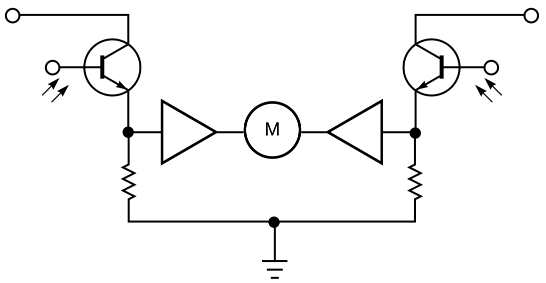


Figure 5 This circuit is the same as that in Figure 4; the value of the resistors depends on the parts you use.

rent may be too high, or the switches may be closing switches or light barriers.

If you use an H bridge to drive the motor, you can achieve the same operation in a more versatile way. The circuit in **Figure 2** shortens one input of the H bridge to ground so that movement is possible only in the other direction by turning on the other input. If the switches are opening at the end positions, the circuit in **Figure 3** interrupts the connection to the driving circuit of one input and sets the input to low using a pulldown resistor. The same circuit works with phototransistors without modification (**figures 4** and **5**). The value of the resistors depends on the parts you use; a value of 10 kΩ should work in most designs.**EDN**

Implement a clip-detection circuit for BTL Class D amplifiers

Dimitri Danyuk and Rich Lenser, Niles Audio, Miami, FL

Clip detection is a convenient feature in Class AB amplifiers. It produces a signal from a clip-detection pin that drives an automatic volume control, which reduces gain compression and distortion when the amplifier is overdriven. Class AB amplifiers, such as the STMicroelectronics (www.st.com) TDA7293, TDA7396, STA7360, and STA540 and the Toshiba (www.toshiba.com) TA8275 and TB29xx, have on-chip clip-detection circuits. Newer Class D automotive amplifiers, such as the four-channel STMicroelectronics TDA7454 and the Texas Instruments (www.ti.com) TAS5414/5424, have on-chip clip-detection circuits, but these ICs use a common clip-detection pin, comprising hardware ORed inside the IC, for all four channels. Other Class D amplifiers lack the clip-detection feature altogether, but you can implement it with external components.

An analog-input Class D amplifier comprises PWM (pulse-width-modulation) logic, gate-drive circuits, and a power stage. The PWM logic transforms the analog-input signal into a PWM signal. The power stage with

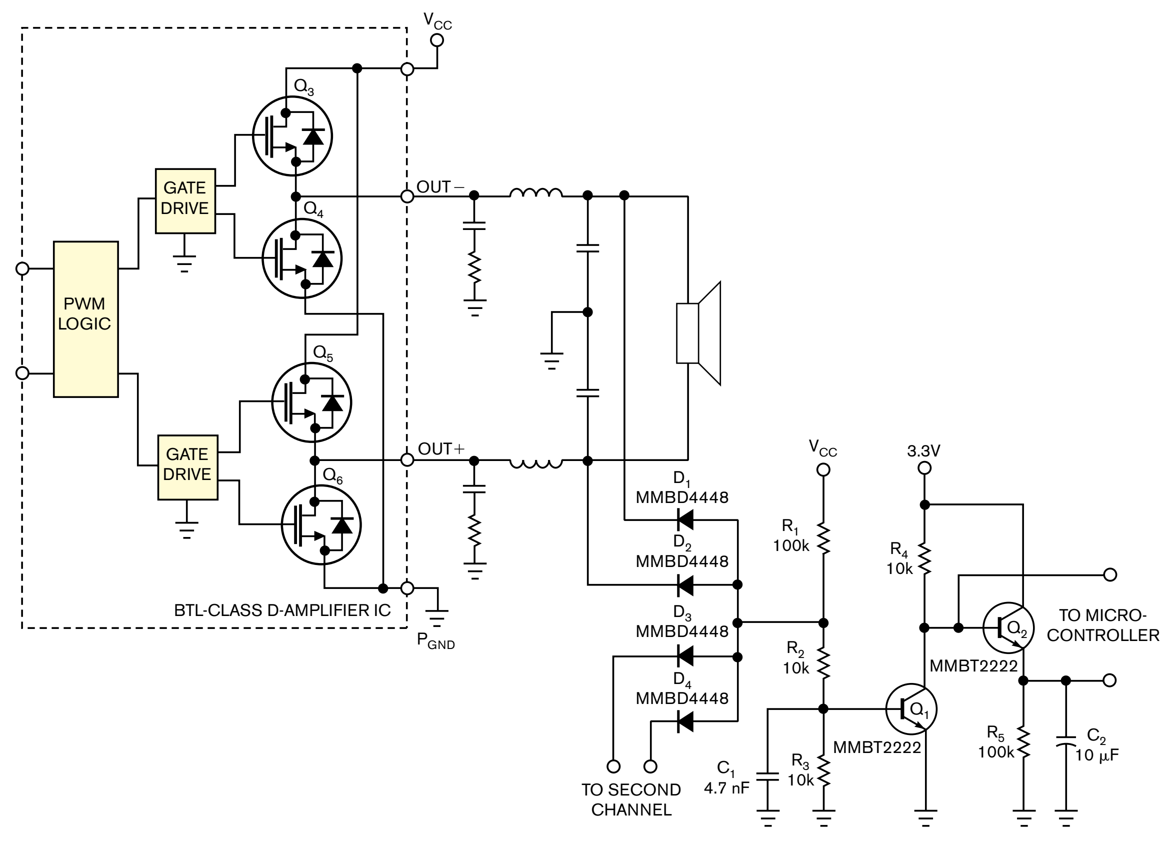
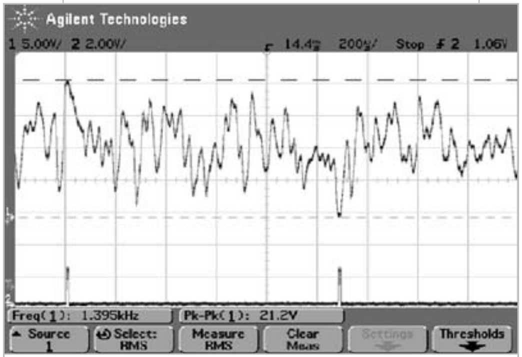


Figure 1 Adding several components to the BTL-Class D-amplifier IC provides a clip-detection function. A peak detector, comprising Q_2, R_5, and C_2, is optional.

the gate drivers transforms the low-power PWM signal into a high-voltage, high-current PWM sequence. A BTL (bridge-tied-load) amplifier basically comprises two gate-drive circuits and two power stages, which the same PWM signal drives. The signal directly drives one gate-drive circuit and phase-inverts the other. In theory, a BTL amplifier can produce four times more power into the same load than a single-ended amplifier.

Figure 1 illustrates the implementation of an external clip-detection circuit to a BTL-Class D-amplifier IC. The voltage swing on each output is symmetrical and is within the range of voltage drop on the on-resistance of MOSFET Q_6 to the common-collector voltage, V_{CC}, minus the voltage drop on the on-resistance of MOSFET Q_3. When the output voltage reaches a certain threshold, Q_1 turns off. The component values of R_1, R_2, and R_3 and the voltage drop across diodes D_1 through D_4 set this threshold, which is 0.5V with respect to power ground, P_{GND}, for the given component values. A positive-going pulse appears on the collector of Q_1 whenever the output voltage is below the threshold with respect to power ground. This pulse alerts the host microcontroller to the existence of clipping (**Figure 2**). Capacitor C_1 filters the residual of the switching- and high-frequency content of the audio signal.



Freq(1): 1.395kHz
Pk-Pk(1): 21.2V

Figure 2 A positive-going pulse appears on the collector of Q_1 whenever the output voltage is below the threshold with respect to power ground.

A simple application involves filtering and integrating the pulses with further automatic reduction and restoration of the volume setting using the microcontroller's driven-volume control to counteract the clipping distortion. You can also implement more sophisticated algorithms (**Reference 1**). A suitable peak detector comprising Q_2, R_5, and C_2 allows the circuit to hold the short clipping pulses for a longer time. You can add LED circuitry to provide a visual clipping indication.**EDN**

REFERENCE

1 Person, Andrew P, and James P Muccioli, "Adjustable Clip Detection System," US Patent, 5,453,716, Sept 26, 1995, US Patent Office, http://patft.uspto.gov.