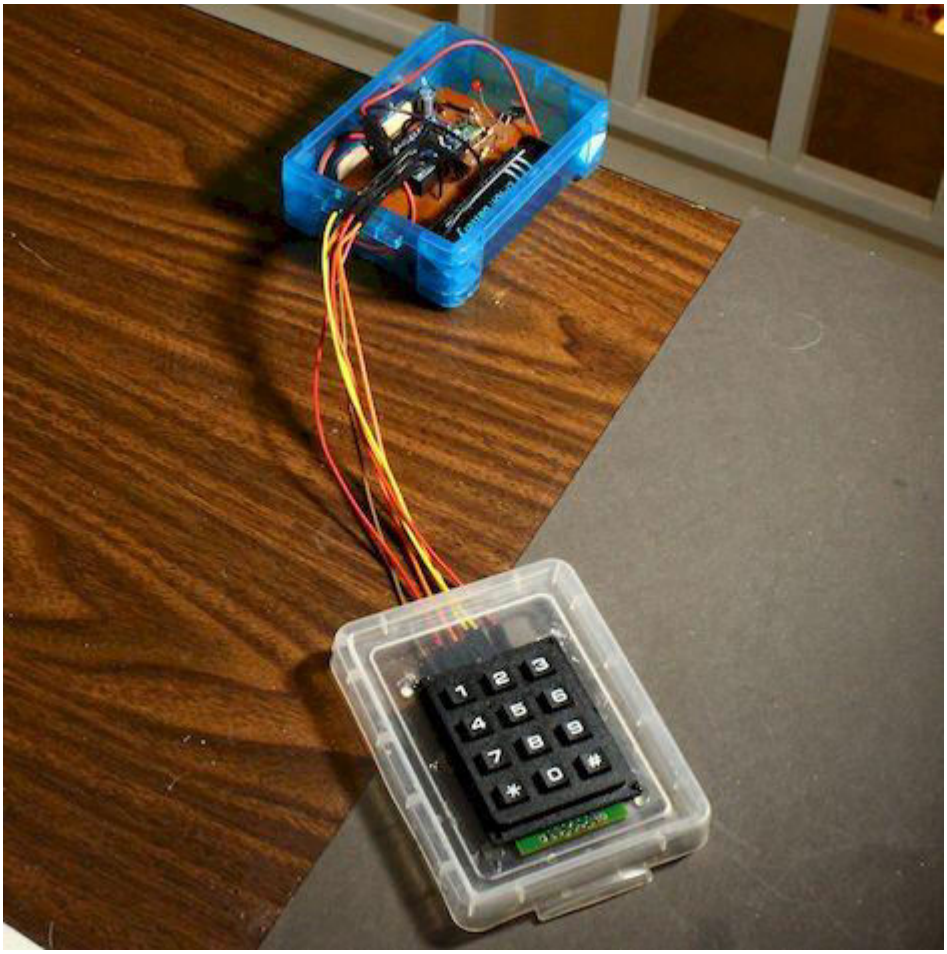
v1.1

Timed Mute TV Remote

This is a special purpose TV remote project. My office is in an alcove at the top of the stairs, looking down on the TV below about 30 feet away. I often have the TV on, listening more than watching, and wanted to be able to reach over and press a numerical key on a keypad, and have the remote controller mute the TV audio for that number of minutes, then unmute automatically. I didn't want to have to pick up the remote and point it at the TV. Just press the key. My primary purpose was to mute commercial blocks, but not have to remember to unmute the audio later.

For this to work, the IR LED has to be placed in a fixed position pointed at the TV all the time. But the keypad can be at a more convenient location. I ended up putting the LED, the circuit, and the batteries into a Walmart crayon box, which I placed at the rear edge of the desk with the LED pointed correctly, and mounted the keypad on the removable top of the box, which just lies on my desk, within easy reach.



This device uses a standard 4x3 keypad, an Arduino Pro Mini 3.3V 8MHz, and an IR LED to form a remote control. Entering a number on the keypad mutes the TV for that many minutes, then unmutes it. I also added some other functions, which include adjusting the volume, powering the TV on or off, and changing the channel. Here's the menu:

```
{N}       - Mute TV for {N} minutes
{N}*      - Mute TV for {N + 0.5} minutes (0* = 30 seconds)
          - Hit any key to unMute early
 * long   - Toggle TV Mute
 # long   - Toggle TV Power
 *        - Soundbar audio: minus one tick
 #        - Soundbar audio: plus one tick
{N}#      - Cable box channel number entry
```

IR HEX Codes

In my case, these functions required communicating with three devices - the TV, the sound bar, and the cable box. You will have to change the protocols and IR hex codes, or address/command codes, to match your devices. The **IRremote.h** library will be helpful in receiving and interpreting your remotes' keypresses. You will need an IR receiver, such as the TSOP38438, to do that. I've also included in the Utilities folder a sketch named **IRCaptureRaw.ino**, which is used with a receiver to give you the raw output of a remote control key. The patterns produced may help identify the protocol if IRremote.h isn't able to do so. Also included is a sketch named **hexMSBtoLSB.ino**, which converts a hex code in MSB-first order to LSB-first, or vice versa, in case that is needed.

Hardware

As shown in the schematic, the circuit combines a keypad, an Arduino Pro Mini, a mosfet LED driver, and the IR LED, all powered by two AA alkaline batteries. In addition, you will need an FTDI USB-to-UART adapter to program the Pro Mini since it doesn't have a USB socket, and a second Arduino to use in flashing a new fuse byte to the Pro Mini (see below).

You should remove the power indicator LED and the voltage regulator from the Pro Mini. These parts consume power, but aren't needed in this circuit. After removal, the Pro Mini will consume less than 1uA of current in Power-Down sleep mode. See Ed Mallon's video in the Links section below for how to remove these parts. He also removes the reset button, but you shouldn't do that.

Unfortunately, there are now counterfeit ATmega328P MCUs that sometimes appear on Pro Minis. These chips sleep at 140uA, which reduces battery life. Included in the Utilities folder is a sketch by Kevin Darrah named **TestForFake328p.ino** which you can run before modifying the Pro Mini. It will print out some of the serial number page of the 328P, which may help determine if the chip is counterfeit. Instructions for using it are found in Kevin's video in the Links section below. And after removal of the LED and regulator, the sketch **SleepCurrentTest.ino** can be used to actually measure the Power-Down sleep current of your Pro Mini with your multimeter. Unfortunately, removal of the parts makes the Pro Mini non-returnable, so if you detect a bad one at this point, it's probably just a loss, and you'll need to try a different supplier. To avoid these problems you can order your Pro Mini and headers from Sparkfun at a higher cost.

The circuit is typical of an IR remote. PWM (38KHz, 30% duty cycle) is output on D3. That drives the gate of an N-channel mosfet, which in turn switches the IR LED current on and off. There is an additional red LED that shares the same drive. It's just a visible indicator that IR is being transmitted, and is optional. The IR LED is the TSAL6100 or 6200. They are the same LED except the 6100 has a narrow beam and the 6200 has a bit more spread. The TSAL6100 would be preferred for long range use.

Keypad

My first keypad was a blue/red membrane keypad measuring 78 x 70 mm. I chose a plastic crayon box at Walmart that matched that size. But that keypad isn't very good. There's no feedback to indicate whether you've pressed the key or not. So I switched to a black keypad measuring 64 x 51 mm, which has actual keys with springs. It feels much better to the touch. Sources for both are in Links below.

IR LED Power

The range of the remote is determined by the current flowing through the IR LED. As seen in the schematic, I used two 10Ω resistors in parallel for a net value of 5Ω for R1, which produces an LED current of about 250mA on fresh AA batteries (1.59V each). Batteries discharged down to 1.33V produce about 190mA. If your setup doesn't require great distance, you could use a single 10Ω resistor instead, or even two 10Ω resistors in series, to operate at a lower current with better battery life. You may have to experiment to find the resistance that works best. You want the least current that still works reliably. Two AA alkaline cells in series should be enough for this remote. AAA alkaline or AA NiMH rechargeables may also work, but I haven't tested them.

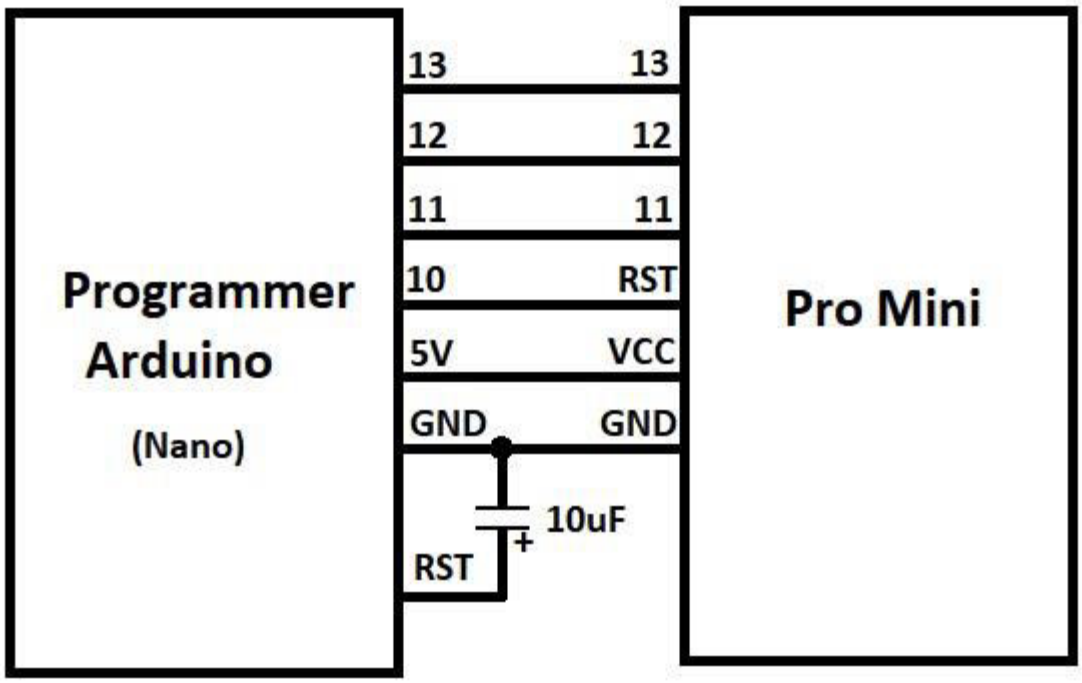
Low Voltage Operation

The Pro Mini's brownout detector defaults to 2.8V, below which the processor shuts down. However, the 328P specs say operation at 8Mhz is guaranteed reliable down to 2.4V, and in practice it may work at even lower voltages. So to maximize battery life, you will need to lower the BOD voltage down to 1.8V. That requires changing the "Extended" fuse byte.

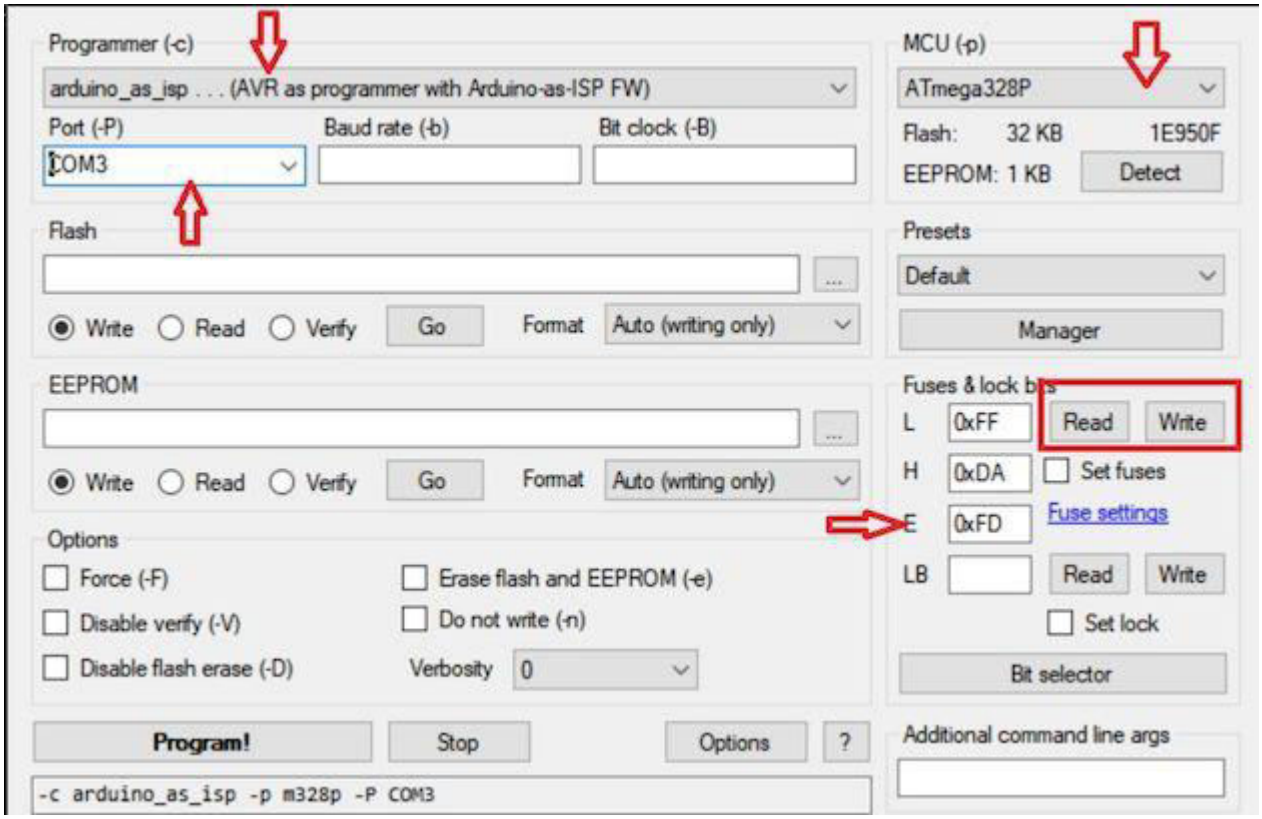
The best way to burn the new fuse value is to use the AVRDudess software by Zak Kemble, and a second Arduino such as a Nano as the "programmer".

1. Before connecting anything else to it, use the IDE to burn the "ArduinoISP" example sketch to the programmer Arduino. Then disconnect it from USB and shut down the IDE.

2. Connect the programmer Arduino to the Pro Mini with six jumpers, and install a 10uF capacitor to ground from the programmer's Reset pin. The Pro Mini's VCC pin should be driven by whatever voltage the programmer Arduino is running at - 5V for the Nano.



3. Connect the programmer Arduino to your computer's USB, boot AVRDudess, fill in the Programmer, Port, and MCU fields. Click on the fuses READ button to load in the current fuse settings, change the Extended fuse value from 0xFD to 0xFE, then click on the WRITE button. Then READ again to make sure it worked. Then disconnect everything. Your Pro Mini's BOD level is now 1.8V.



Software

The **IRremote.h** library is used to generate the IR signals. I was able to figure out v4.4.1 enough to make it all work, but I know many have stayed with v2.x. That's fine, but you will need to change the code to work with v2.x. The number of code lines dealing with IR are very few, so it's fairly easy to make any needed changes..

But the **Keypad.h** library is another matter. It was designed to be non-blocking, but that provides no benefit here, and it introduces some strange behavior. I was able to work around that by adding some delay at key points, but I don't really understand why the delays are needed. Along the way I wrote another version that doesn't use the library, but deals directly with the keypad lines instead. It is blocking code, but for this project that doesn't matter. I've included both versions of the sketch. They function identically.

The loop() begins by setting the column pins to OUTPUT LOW, then makes sure no key is currently being pressed. Then it puts the processor to sleep with pin change interrupts enabled on the row pins. When a key is pressed, that key's row pin goes low, triggering the interrupt. The column pins are returned to INPUT mode, and the key triggering the interrupt is identified.

If that key is "*" or "#", it's necessary to determine if the keypress is short or long, and the appropriate command is then executed. If the first key is a number key, then it's the beginning of either a mute time or a channel number, in which case any additional keypresses are read in, and the final result is executed.

The built-in LED on D13 is either lit up solid, or blinking, whenever the Pro Mini is awake. It blinks during any mute period. So if it's dark, the Pro Mini is asleep.

Links

Counterfeit 328P test: Here is Kevin Darrah's video testing for a counterfeit processor. His sketch is included here in Utilities as **TestForFake328P.ino**.

https://www.youtube.com/watch?v=eeDC1m7ANJI

Pro Mini mods: Here is Ed Mallon removing the power LED and voltage regulator from the Pro Mini. He also removes the reset button, but you shouldn't.

https://youtu.be/58ps9fUyY0Q?si=Lq8arIt_leUYYNnu&t=1495

AVRDudess:

https://blog.zakkemble.net/avrdudess-a-gui-for-avrdude/

Arduino Pro Mini 3.3V 8MHz clone:

https://www.ebay.com/itm/200914924969

IR LEDs:

https://www.digikey.com/en/products/detail/vishay-semiconductor-opto-division/TSAL6100/1681338

https://www.digikey.com/en/products/detail/vishay-semiconductor-opto-division/TSAL6200/1681339

IR Receiver:

https://www.digikey.com/en/products/detail/vishay-semiconductor-opto-division/TSOP38438/4073478

N-Channel Mosfet:

https://www.digikey.com/en/products/detail/alpha-omega-semiconductor-inc/AOSS32136C/11567433

SOT23-3 Adapter Board for Mosfet:

https://www.ebay.com/itm/267022324448

2 x AA Battery Holder:

https://www.ebay.com/itm/315742626644

Capacitor:

https://www.digikey.com/en/products/detail/panasonic-electronic-components/EEU-FR0J221/9921019

4x3 Keypad:

https://www.ebay.com/itm/354987302599
Rows = 1,2,3,4. Cols = 5,6,7

https://www.aliexpress.us/item/3256805628620343.html
Rows = 2,7,6,4. Cols = 3,1,5 NC = 0,8

Protoboards:

https://www.ebay.com/itm/265276759831

Project Box:

https://www.walmart.com/ip/Super-Stacker-Crayon-Chalk-Box-1-6-X3-5-X4-8-Assorted-Colors/131564543

FTDI Adapter:

https://www.ebay.com/itm/201543906640