

---

# Morse Code Buzzer Controller

Revision v0.6 01/25/18

---

Seth Neumann, [seth@modelrailroadcontrolsystems.com](mailto:seth@modelrailroadcontrolsystems.com)

## Introduction

This document describes the Morse Code Buzzer Controller and how to assemble and install it.

## Revision History

v0.4 9/25/17 SCN

v0.6 1/15/18 SCN added another system wiring diagram, updated recommended phone wiring

DRAFT

---

# Table of Contents

<b>1</b>	<b>INTRODUCTION</b> .....	<b>3</b>
<b>2</b>	<b>IDENTIFICATION AND INFORMATION</b> .....	<b>4</b>
2.1.	VERSION 1.0.....	4
2.2.	VERSION 2.0.....	6
2.3.	VERSION 3.0.....	8
<b>3</b>	<b>OPTIONS</b> .....	<b>10</b>
3.1.	PRO-MINI VARIANTS.....	10
3.2.	CONNECTION INFO:.....	11
3.3.	ULN2803.....	12
3.4.	CONNECTORS.....	12
3.5.	DETAILED ASSEMBLY.....	13
<b>4</b>	<b>SOFTWARE (FROM STEVE WILLIAMS)</b> .....	<b>14</b>
4.1.	OVERVIEW.....	14
4.2.	GOALS.....	14
4.3.	SPECIFICATIONS.....	15
4.3.1.	Call Behavior.....	15
4.3.2.	Ambience stations.....	16
4.3.3.	Buzzers.....	17
4.3.4.	Off-hook inputs.....	17
4.3.5.	Called inputs.....	17
4.4.	SOFTWARE CONFIGURATION.....	17
<b>5</b>	<b>TESTING</b> .....	<b>22</b>
<b>6</b>	<b>INSTALLATION AND CONNECTIONS</b> .....	<b>23</b>

# Table of Figures

FIGURE 1 - REV 1.0 BOARD WITH PRO-MINI AND LAYOUT.....	4
FIGURE 2 - REV 1 SCHEMATIC.....	5
FIGURE 3 - REV 2 SCHEMATIC.....	7
FIGURE 4 - REV 3.0 BOARD WITH PRO-MINI AND LAYOUT.....	8
FIGURE 5 - REV 3 SCHEMATIC.....	9
FIGURE 6- PRO-MINI IDENTIFICATION.....	10
FIGURE 7 - STATION STATE DIAGRAM.....	15
FIGURE 8 - CONNECTIONS FOR REV 1 BOARDS.....	23
FIGURE 9 - REV 2 BOARD WITH STATIONS 1 AND 6 CONNECTED SHOWING INTERNAL PHONE CONNECTIONS TO SPEECH NETWORK.....	24
FIGURE 10 - REV 3 BOARD WITH CONNECTIONS.....	24
FIGURE 11 - TYPICAL TELEPHONE CONNECTIONS.....	26
FIGURE 12 - MCBC WIRING TO STATION.....	27

# Table of Tables

TABLE 1- BILL OF MATERIALS REV 1.0.....	5
TABLE 2 - BILL OF MATERIALS REV 2.0.....	6
TABLE 3 – BILL OF MATERIALS REV 3.0.....	8
TABLE 4 – REV 1,2 PIN MAPPING.....	11
TABLE 5 - REV 3 PIN MAPPING.....	11

---

# 1 INTRODUCTION

---

This board is an Arduino Pro Mini breakout primarily designed to support Steve Williams' coded buzzer sketch. It also serves as a break out boards for other applications.

- Rev 1 and 2 operate on 5 volts regulated DC, draws about 40 mA, Rev 3 uses 6-12 VDC regulated, wall wart, we recommend 7.5 – 9V to reduce the load on the on board regulator on the Pro-Mini
- Rev 1 and 2 support up to 6 stations (to use the 6<sup>th</sup> you must remove a resistor on the Pro-Mini by unsoldering)
- Each station uses two inputs and one output
- Inputs are active low\* with on board pull ups, the output can sink up to .5 A at 50V DC (suitable for many sounders at 24V, but check the current draw on your sounder)
- The Sketch allows up to 7 stations to be active at any one time, but only one is buzzed at a time in order to keep the noise in the layout room reasonable and to allow operators to more accurately localize their buzzer/sounder.
- uses standard semiconductors available from Jameco and similar suppliers
- Rev 1 and 2 are based on a board that uses 6 analog capable inputs, 6 digital inputs ( or outputs) and 6 PWM capable outputs that are connected to Darlington Drivers capable of sinking up to .5A at 50V, with internal clamps. The board is also suitable for projects such as using an analog pot to control a moderate current load with PWM (LED Lighting, Motor Control, etc). This arrangement limits the board to 6 circuits and requires modifying the Pro-Mini to use the 6<sup>th</sup> circuit.
- Rev 3 is optimized for Steve's Sketch and supports up to 7 circuits plus and "ambience" output which randomly plays a selection of user-entered messages at random intervals. No modification of the Pro-Mini is required but there are several footprints of Pro-Minis around so be sure you use a board with original Sparkfun footprint. Almost all of these are compatible on at least the first 4 circuits. The Rev 3 board retains the Darlington Drivers so you can drive buzzers or sounders. See below for details.

\* These defaults can be changed by modifying the sketch, -- see "Software" below

All components are through-hole technology for ease of assembly and repair.

Input connection pads are standardized on .100" centers. This provides a wide range of interconnect options and components. Connection schemes include screw terminal blocks, header pin connectors (male and female), soldered right angle headers, and direct soldered wires.

Outputs are on 3.5mm connectors to handle larger wires for more current.

Power can be supplied on either a 2.1mm center positive barrel connector or a 5mm terminal block. A separate 5mm terminal block is provided for load ground and supply (if you intend to use the internal clamp diodes).

## 2 IDENTIFICATION AND INFORMATION

### 2.1. VERSION 1.0

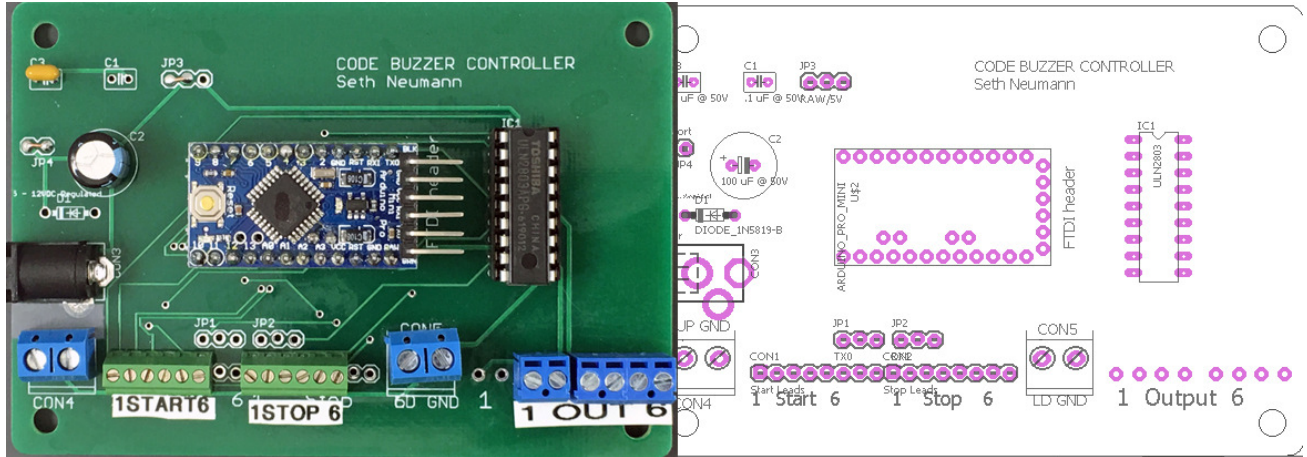


Figure 1 - Rev 1.0 Board with Pro-Mini and Layout

The Revision 1.0 board has some issues with labels not aligning properly and was shipped with label tape over the board labels. These were all green. They can also be distinguished by the vertical line of 6 pads (not stuffed) next to the “FTDI header” label. These boards support up to 6 stations and, for the hackers, have start (“called” in the sketch) leads on Analog capable inputs. A separate 5mm jack is provided for output ground and for a connection to the load supply so the clamp diodes in the ULN2803 Darlington driver will protect the outputs. Using the 6<sup>th</sup> circuit, requires removing the resistor for D13 from the Arduino Pro-Mini. This board uses 5VDC regulated power.

A quantity of zero denotes an alternate component or a component that may be used in a particular configuration.

Qty	Value	Device	Parts	Description	Vendor	Part #
2	.1 uF @ 50V	C-US025-030X050	C1, C3	Cap, monolithic	Jameco	25523
1	100 uF @ 50V	CPOL-USE3.5-10	C2	POLARIZED CAPACITOR, American symbol	Jameco	29962
2	1751264	4 Position Header	J1, J2	MKDS 1/ 4-3,5 Printklemme	Jameco	212411
1	ARDUINO_PRO_MINI	ARDUINO_PRO_MINI	U\$2		eBay	
1	DIODE_1N5819-B	DIODE_1N5819-B	D1	1.0A SCHOTTKY BARRIER RECTIFIER	Jameco	177965
1	Power	CONNECTOR-DC-POWER-RA	CON3	DC POWER JACK	Jameco	101178
2	Power, load	CONNECTOR-M02508-LOCK	CON4,5	PHOENIX CONNECTOR	Jameco	2094485
1	RAW/5V	JUMPER-3PTH-LOCK	JP2,3,4		Banggood	
2	12 pos 0.100 female	12 pos 0.100 female	JP5,6	for arduino	Pololu	1030
1	2 pos 0.100 female	2 pos 0.100 female	JP7,8	for arduino	Pololu	1012
1	jumper plug	for JP 2,3,4	jmp	power, D0, D1	Jameco	112432
2	Start,stop Leads	CONNECTOR-M08LOCK	CON1,2	Header 6 (only 6 positions used for ncb)	E-Salon	
1	ULN2803	ULN2803A	IC1	DRIVER ARRAY	Jameco	34315
1	18 pin socket	18 pin socket	socket	socket 18	Jameco	112231
1	short	M02LOCK	JP4	Standard 2-pin 0.1 header. Use with"	Banggood	
1	board 69x100 mm	New Coded Buzzer		Board	Seed	

## Table 1- Bill of Materials Rev 1.0

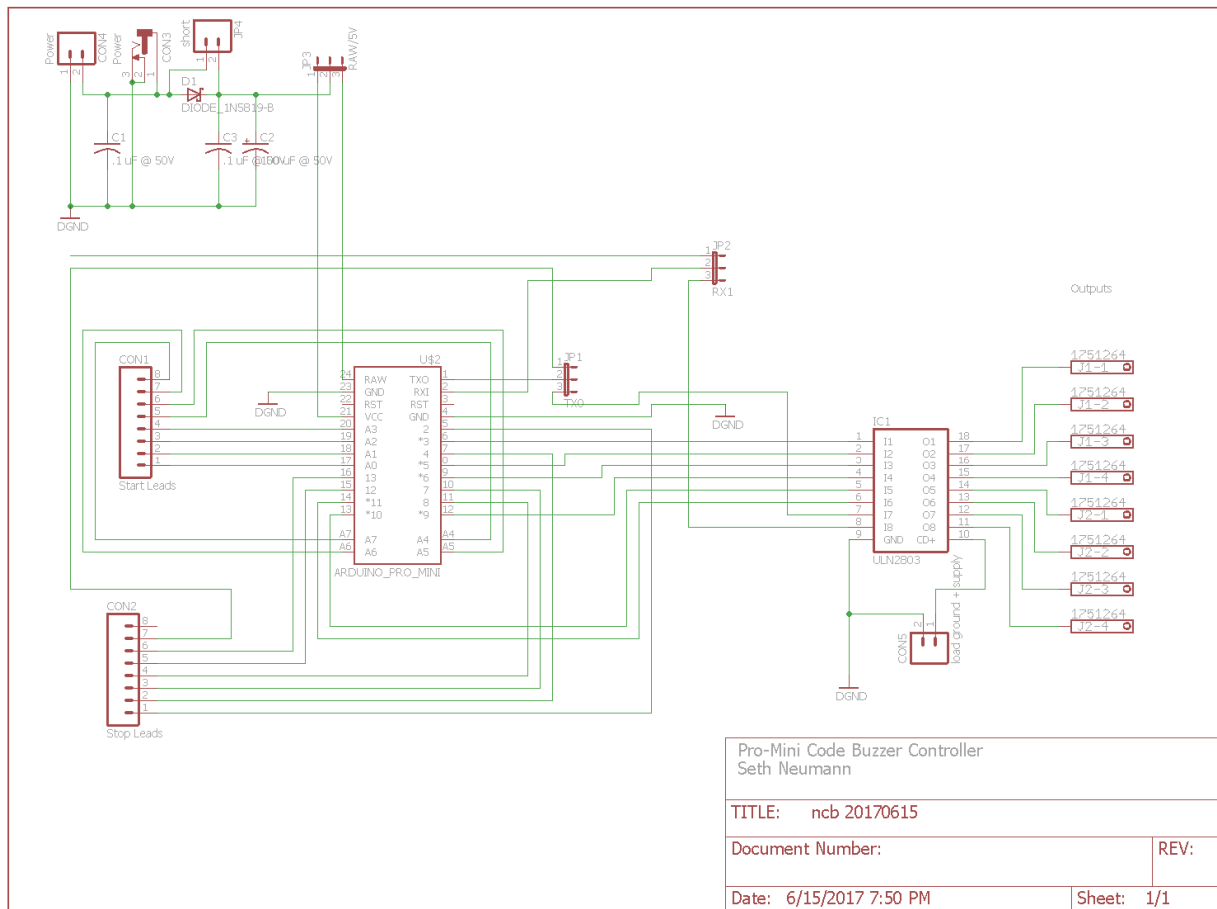
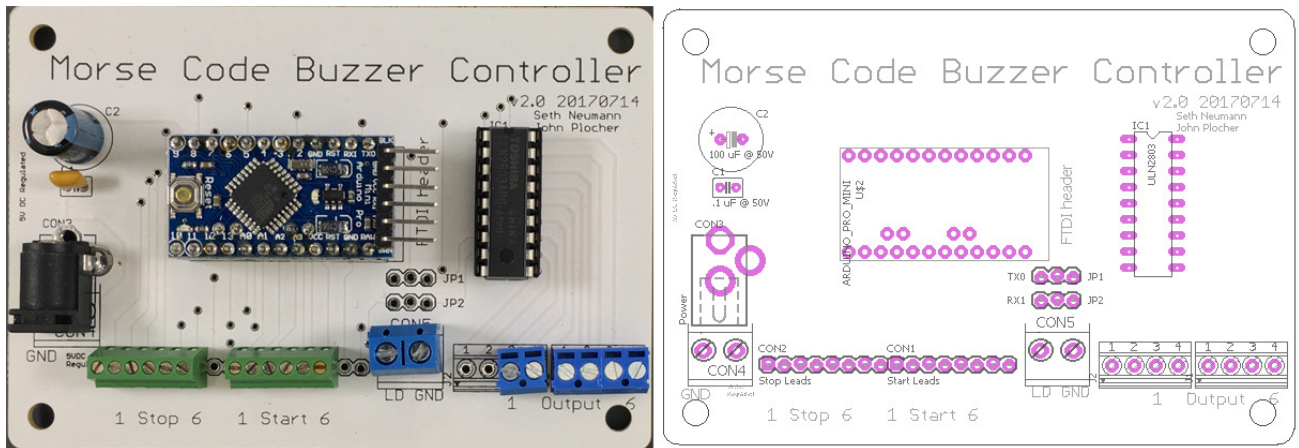


Figure 2 - Rev 1 Schematic

## 2.2. VERSION 2.0



The Revision 2.0 board was redrawn by John Plocher and the layout is optimized, and the labels and legends have been corrected. These were all on white PCBs and have 2.0 20170714 printed on the upper right. The inputs are configured the same Rev 1.0. The power connector has been rotated and placed so that you can either use the 2.1mm barrel jack or a 5mm power connector, not both. A separate 5mm jack is provided for output ground and for a connection to the load supply so the clamp diodes in the ULN2803 Darlington driver will protect the outputs. We are now shipping the Rev 3 configuration, but if you want to order this board (for an A/D project) we can make them. Using the 6<sup>th</sup> circuit, requires removing the resistor for D13 from the Arduino Pro-Mini. This board uses 5VDC regulated power.

A quantity of zero denotes an alternate component or a component that may be used in a particular configuration.

Qty	Value	Device	Parts	Description	Vendor	Part #
1	.1 uF @ 50V	C-US025-030X050	C3	Cap, monolithic	Jameco	25523
1	100 uF @ 50V	CPOL-USE3.5-10	C2	POLARIZED CAPACITOR, American symbol	Jameco	29962
1	1751264	4 Position 3.5mm screw terminal	J1, J2	MKDS 1/ 4-3,5 Printklemme	Jameco	212411
1		2 Position 3.5mm screw terminal				
1	ARDUINO_PRO_MINI	ARDUINO_PRO_MINI	U\$2		eBay	
1	Power	CONNECTOR-DC-POWER-RA	CON3	DC POWER JACK	Jameco	101178
2	Power, load	CONNECTOR-M02508-LOCK	CON4,5	PHOENIX CONNECTOR	Jameco	2094485
0	RAW/5V	JUMPER-3PTH-LOCK	JP2,3,4		Banggood	
2	12 pos 0.100 female	12 pos 0.100 female	JP5,6	for arduino	Pololu	1030
1	2 pos 0.100 female	2 pos 0.100 female	JP7,8	for arduino	Pololu	1012
0	jumper plug	for JP 2,3,4	jmp	power, D0, D1	Jameco	112432
2	Start,stop Leads	CONNECTOR-M08LOCK	CON1,2	Header 6 (only 6 positions used for ncb)	E-Salon	
1	ULN2803	ULN2803A	IC1	DRIVER ARRAY	Jameco	34315
1	18 pin socket	18 pin socket	socket	socket 18	Jameco	112231
0	short	M02LOCK	JP4	Standard 2-pin 0.1 header. Use with"	Banggood	
1	board 69x100 mm	New Coded Buzzer		Board	Seed	

**Table 2 - Bill of Materials Rev 2.0**

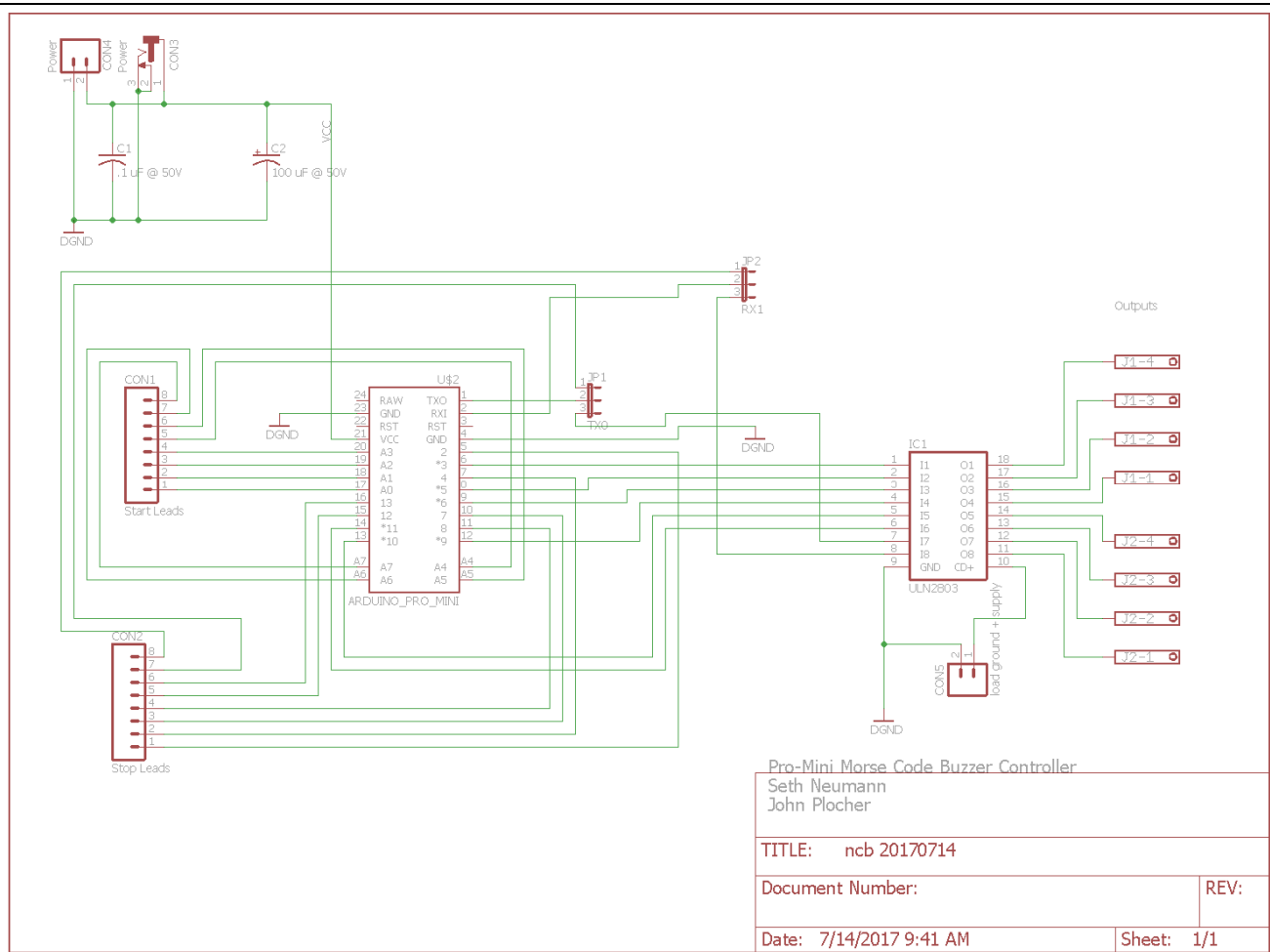
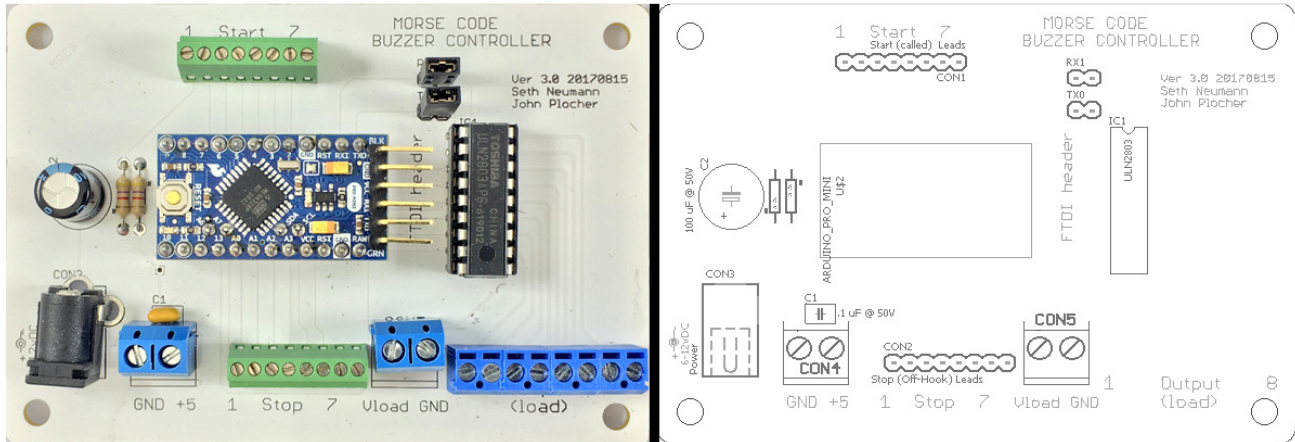


Figure 3 - Rev 2 Schematic

## 2.3. VERSION 3.0



**Figure 4 - Rev 3.0 Board with Pro-Mini and Layout**

The Revision 3.0 board (all white PCBs) labeled version 3.0 20170815 are optimized for Steve Williams’ Station Buzzers sketch. We abandoned the mapping of analog-capable inputs to the start (called) leads and the PWM capable outputs to the ULN2803. We added pull up resistors to the A6 and A7 pins, to provide more inputs. This board supports up to 7 circuits plus an eighth, “ambience” output. This board uses 6-12 volt regulated power, lower is better.

A quantity of zero denotes an alternate component or a component that may be used in a particular configuration.

Qty	Value	Device	Parts	Description	Vendor	Part
2		JUMPER-2PTH	RX1, TX0	Jumper for TX and RX	Jameco	112432
1	.1 uF @ 50V	C-US025-030X050	C1	CAPACITOR, American symbol	Jameco	25523
1	100 uF @ 50V	CPOL-USE3.5-10	C2	POLARIZED CAPACITOR, American symbol	Jameco	29962
2	3.5mm Screw terminal	3.5mm Screw terminal	J1, J2		Jameco	212411
2	4.7K	RPTH04	R1, R2	Resistor	Jameco	691024
2	2 position 0.100 Male	2 position 0.100 Male	RX1, TX0	2 position 0.100 Male	Jameco	108338
1	ARDUINO_PRO_MINI	ARDUINO_PRO_MINI	U\$2		Sparkfun	DEV-11113
1	Power	CONNECTOR-DC-POWER-RA	CON3	DC POWER JACK	Jameco	101178
1	Power,load ground + supply	CONNECTOR-M02508-LOCK	CON4, 5	PHOENIX CONNECTOR	Jameco	2094485
2	start and stop	CONNECTOR-M08LOCK	CON1,2	8 positon (pole) 2.54mm (0.100) screw term	e-Salon	
1	ULN2803	ULN2803A	IC1	DRIVER ARRAY	Jameco	34315
1	18 pin DIP socket	for ULN 2803			Jameco	112231
2	12 position female	for Arduino Pro-Mini	JP5,6	long sides of Arduino	Pololu	1030
2	2 positon female	for Arduino Pro-Mini	JP7,8	A4-5, A6-7	Pololu	1012
1	Board 69 x 100 mm				MRCs	

**Table 3 – Bill of Materials Rev 3.0**



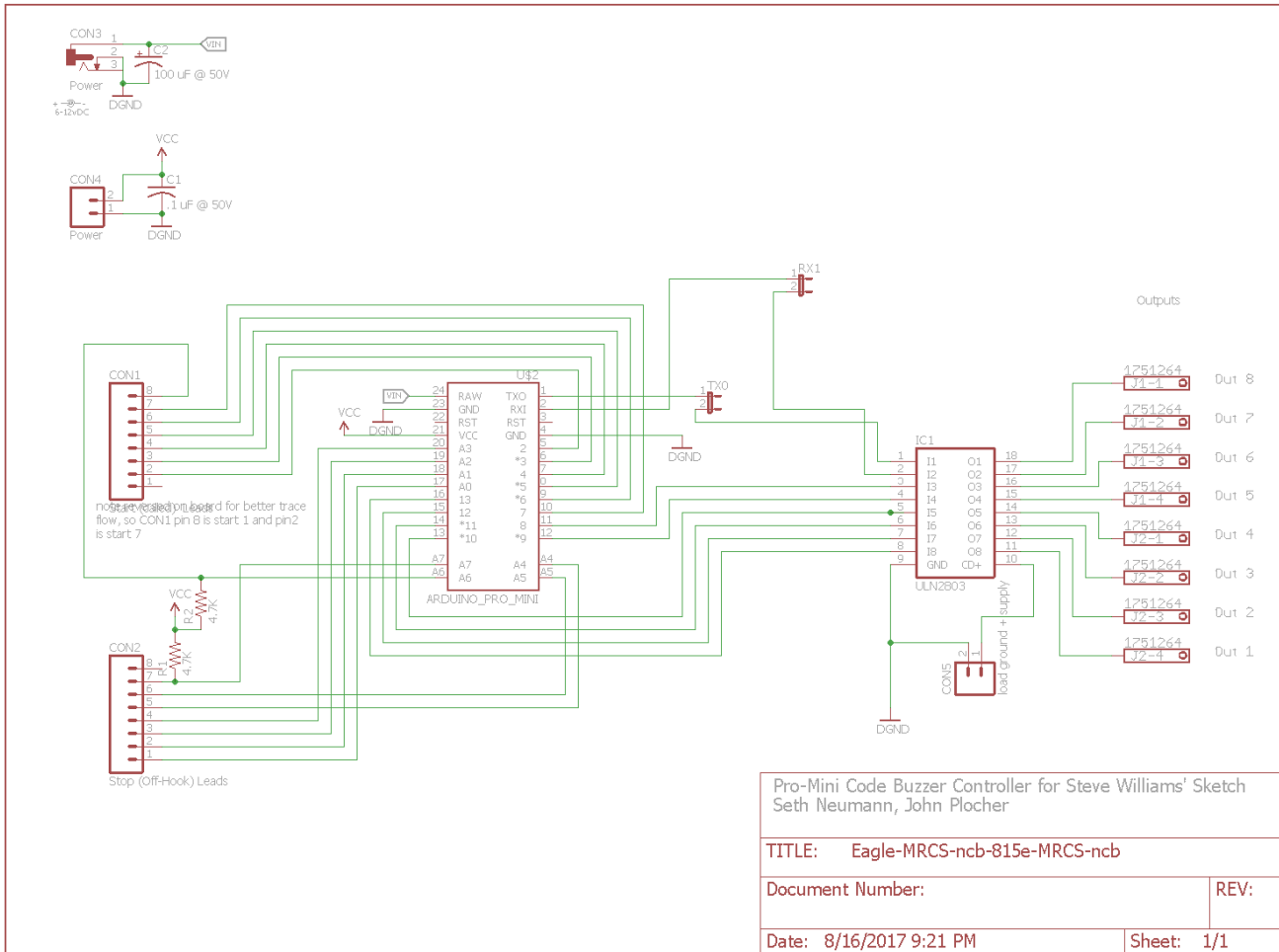


Figure 5 - Rev 3 Schematic

---

## 3 OPTIONS

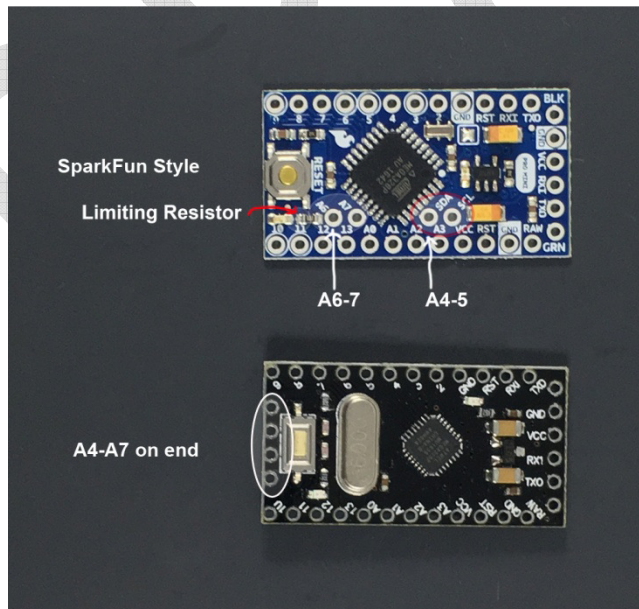
---

### 3.1. PRO-MINI VARIANTS

The Pro-Mini was developed by Sparkfun [www.sparkfun.com](http://www.sparkfun.com) as a small, inexpensive alternative to Uno and Leonardo using surface mount technology and eliminating the USB port, in favor of an FTDI interface. You will need an FTDI cable or converter to program the pro-mini. These are inexpensive and can be found at Speakfun, Adafruit and many inexpensive off-shore outlets.

There are several alternative footprints to the pro-mini. We use the revised Sparkfun version which has two rows of 12 pins along the long sides, a 6 pin FTDI header at one end and two sets of two off set pins for A4-A5 and A6-A7. If you get your Pro-Mini from Sparkfun, just go ahead and add 0.100 headers to the long sides and off set pins. One of the other designs has A4-5 in the off set position and A6-7 on the end opposite the FTDI header, the other has A4-7 on the opposite end. Be sure you've got the one you need. If you are not using all 7 positions you won't need all of the A(4-7) inputs, so check your requirements and you may be able to use the other footprints.

As noted above A0 through A5 are inputs. Since these inputs are capable of A to D conversion, I've kept them separate for use in other projects on Rev 1 and 2. The outputs D3,5,6,9,10 and 11 are PWM capable, and since PWM is an output function, I've kept them together on Rev 1 and 2. The stop inputs use the non-PWM digital pins remaining starting with D2 on Rev 1 and 2 boards. D0 and D1 are Rx and TX respectively and they are connected by three position jumpers to either outputs six and seven or stop lines 6 and seven or they can be left open if not otherwise used. The jumpers should be removed while programming. On Rev 3 boards the mapping has changed (see configuration below) to allow up to 7 circuits.



**Figure 6- Pro-Mini Identification**

### 3.2.CONNECTION INFO:

Circuit	Start	Stop	Output	SparkFun Style	A6-7 on end	A4-7 on end
1	A0	D2	D3	X	X	X
2	A1	D4	D5	X	X	X
3	A2	D7	D6	X	X	X
4	A3	D8	D9	X	X	X
5	A4 – not on some PMs	D12	D10	X	X	
6	A5	D13*	D11	X*	X	
	A6 – not digital	D0 – RXI**	D1 TXO**			
	A7 – not digital					
Circuits				6*	6*	4

**Table 4 – Rev 1,2 Pin Mapping**

\* you must remove the LED limiting resistor on D13 on the Pro-Mini to use this input, however output D11 may be used for “ambience” see software configuration without modifying D13

\*\*Pro-Mini pin D0 (TXO) may be connected via jumper JP1 to position 7 on the ULN2803 as an output or to Stop lead 7 as an input, or left unconnected (recommended while programming)

\*\* Pro-Mini pin D1 (RXI) may be connected via jumper JP2 to position 8 on the ULN2803 as an output or to Stop lead 8 as an input or left unconnected (recommended while programming)

Circuit (on stencil)	Start	Stop	Output	SparkFun Style	A6-7 on end	A4-7 on end
1	A6 – external pull up R2	A0	D13	X	X	X
2	D7	A1	D12	X	X	X
3	D6	A2	D11	X	X	X
4	D5	A3	D10	X	X	X
5	D4	A4	D9	X	X	
6	D3	A5	D8	X	X	
7	D2	A7 – external pull up R1	D0 RXI **	X		
8	n/a	n/a	D1 TXO ***			
Circuits				7	6	4

**Table 5 - Rev 3 Pin Mapping**

---

\*\*Pro-Mini pin D0 (TXO) may be connected via jumper JP1 to position 7 on the ULN2803 as an output or to Stop lead 7 as an input, or left unconnected (recommended while programming)

\*\*\* Pro-Mini pin D1 (RXI) may be connected via jumper JP2 to position 8 on the ULN2803 as an output or to Stop lead 8 as an input or left unconnected (recommended while programming). It is recommended that you use TXO if you want to employ the “ambience” function.

### 3.3. ULN2803

The ULN2803 is a high current driver that can sink up to 0.5A at up to 50V on each output. We included it because the lead users wanted to drive telegraph sounders, which operate at 24-48 volts, and the Pro-Mini is limited to 5V (and 25 mA). If you will only be driving logic level loads, you can omit the ULN2803 and wire across the pins for the socket. If you are using a Rev 1 or 2 board for motor or high current LED control, you’ll want to use the ULN2803. We recommend socketing the ULN2803 because it lives in a rough neighborhood and may need replacement.

### 3.4. CONNECTORS

- The input connectors are on 0.100 centers (staggered slightly to hold the connectors in place during assembly). While our standard connector is the 0.100 screw terminal, you may substitute any other 0.100 connector you prefer. If ordering an assembled and tested unit from MRCS and you would prefer a different connector, please contact us at [sales@modelrailroadcontrolsystems.com](mailto:sales@modelrailroadcontrolsystems.com) and indicate your preference and we’ll provide a quotation.
- Output connectors are 3.5mm screw terminals. Again you can substitute any 3.5mm pitch connector.
- We provided a 2.1mm center positive barrel jack for power, Rev 1 and 2 use 5V, Rev 3 uses 6-12 volts. **Be careful and check the output of your power supply, overvoltage will kill the Pro-Mini.** The 5mm pitch screw terminal block parallels the barrel jack (Rev1,2) but in Rev 3 the 5V is the output of the Pro-Mini’s regulator.
- The last connector is a 5mm screw terminal for referencing the output loads (buzzer, sounder etc.) Connect the ground to the ground of the supply powering the outputs (don’t worry about this if running low current/low voltage buzzers off the Morse Code Buzzer Controller’s supply). Connect the positive side of the load supply to the screw marked “LD” or “LOAD” This provides a reference for the clamp diodes in the ULN2803 that protects it from inductive spikes.

---

## 3.5.DETAILED ASSEMBLY

All of the components are through-hole technology with wire leads. A useful tool is a lead bender for forming the leads at 90 degrees for easy insertion into the pad holes. Start with inserting the lower height components. The general rule is install the lowest components first, working towards components that are higher off the board. This enables you to support the low components as you solder them.

### Resistors, Diodes

Install Resistors R1 and R2 (Rev 3 only) on the center left of the boards. Try to keep the gold tolerance band on the same side for easier reading of values.

Install diodes D1 (Rev 1.0 only). Observe polarity – band towards edge of board.

### Monolithic Capacitors

Install C1,C3 (C3 on Rev 1.0 only) on left edge of board

### IC Socket (if you are using the ULN2803)

Socket for IC1 – notch in socket pointing up

### Electrolytic Capacitors

Install C2 last as it is the highest component on the board. Observe polarity: stripe to the right.

Arduino Sockets – these come in various heights, so compare height off board with other components.

### Option Headers

Install JP1,2,3,4 (Rev 1.0), JP1,2 (Rev 2.0, Rev 3.0)

### Screw Terminals and Power Jack

Install Con3 (2.1 mm barrel jack) unless you will not be using a 2.1mm plug to power your Morse Code Buzzer Controller

Install Con1 and Con2

Install Con4 unless you will be using Con3

Install Con5

### Electronics

Install the ULN2803 if you are planning to use high current or high voltage (>5V) loads

Install the Pro-Mini

[Pro-Mini]

Most pro-Minis come with male headers, if not you will need to supply them. I use the board as a fixture to hold the headers in place. Just put the headers into the socket and then gently wiggle the Pro-Mini so it sits down tight and solder the end pins on each header. Then check for alignment, it's easy to adjust with just one pin on each end soldered, a quick re-melt will do.

---

## 4 SOFTWARE (FROM STEVE WILLIAMS)

---

### 4.1. OVERVIEW

On many operations-oriented model railroads, particularly those which operate under a timetable and train-order (TT&TO) or tower-based dispatching system, there is a need to provide telephone communications between the dispatcher and the agent/operators. However, even on fairly large model railroads with multiple agent/operators, the physical distances between the operator positions are often insufficient to allow the operators to reliably distinguish that “hey that’s **my** phone ringing.”

This document describes the design of the “station\_buzzers” Arduino script which controls the buzzers on a model railroad telephone system to play the called station’s timetable station call-sign in American Railroad Morse code, and to cycle among the called stations so that only one buzzer is sounding at a time. Model Railroad Control Systems has developed a companion “Morse Code Buzzer Board” which provides convenient connections to an Arduino Pro-Mini running the station\_buzzers sketch.

### 4.2. GOALS

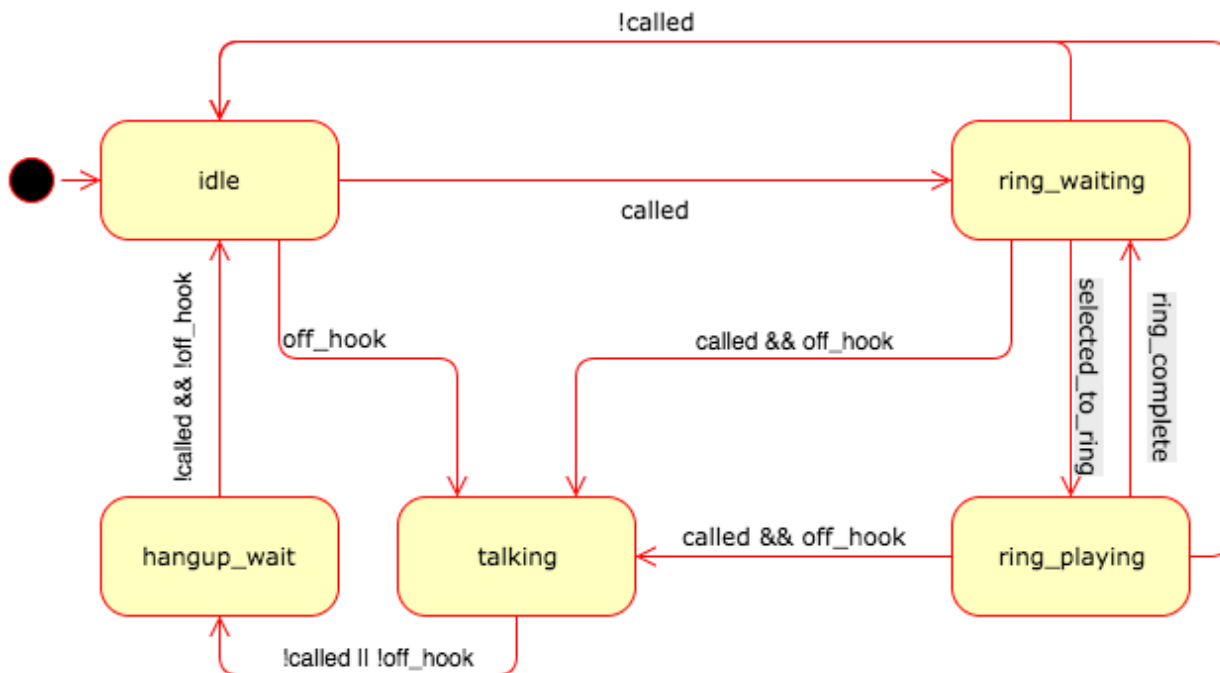
1. Provide separate “buzzer pins” for each station. Each pin can be made active HIGH or active LOW as needed depending on the external buzzer driver circuits used. The Morse Code Buzzer Controller uses an active high output to activate it’s Darlington output which provides a ground to operate a load of up to 0.5A at up to 50V.
2. Provide support for “ambience” buzzers that will randomly play one of several canned messages. These could be train orders, news reports, or “inside jokes”.
3. Provide separate “answered” inputs for each station with individual control of the active HIGH or LOW level for the input. The Morse Code Buzzer Controller assumes a low input, but you can change it in this sketch.
4. Provide separate “called” input pins for each with individual control of the HIGH or LOW.
  - a. Each “called” input should also have individual selection between “momentary” versus “continuous” mode. For “continuous” mode, the station’s call-sign will continue to play until either the station is answered or the caller hangs up. For “momentary” stations, the station will start ringing after a momentary button press and continue ringing until answered or after a timeout. The Morse Code Buzzer Controller assumes a low input, but you can change it in this sketch.
5. Only allow one station buzzer to play at a time. When multiple stations are being called simultaneously, cycle through them playing each station’s call-sign with a couple of seconds delay between the end of one call-sign and the beginning of the next. This provides temporal as well as spatial separation of the station call-signs.

## 4.3. SPECIFICATIONS

Configuration of the list of station's, their call-signs, and operating modes will be done through a C++ structure array.

### 4.3.1. Call Behavior

The basic call flow for a single station is shown in the flow chart below.



**Figure 7 - Station State Diagram**

The states that each station runs through are:

**IDLE** -- both called and off\_hook are deasserted

**RING\_WAITING** -- the station has been called, but not yet answered. Stations enter this state before they are selected to ring. Stations entering RING\_WAITING state set their last\_ring\_millis value to the current millis(), but stations entering from IDLE subtract 60 seconds so that they will look artificially old to the “selected\_to\_ring” logic

**RING\_PLAYING** -- the station has been called and not answered and has been selected to ring. Only one station can be in the RING\_PLAYING state at a time, and typically it will stay there until the Morse code for the station call-sign finishes playing one time. The selection of which station should ring is done by choosing the station which has been in the RING\_WAITING state the longest.

---

**TALKING** -- the station is off-hook. We don't make a distinction between whether we got here directly from IDLE (station-initiated call) or from one of the RING\_WAITING or RING\_PLAYING states. For “momentary” stations, immediately upon entering TALKING, the called input is cleared and the station will transition to HANGUP\_WAIT state.

**HANGUP\_WAIT** -- one or the other sides has hung up and we are waiting for both sides to hang up.

The same basic flow applies to stations that use either “momentary” or “continuous” called inputs. All that is different is how the various “called” and “off\_hook” methods behave.

### 4.3.2. Ambience stations

The sketch will also support defining one or more “ambience” stations. These are intended to be used for extra “audio candy” in the layout room. Each “ambience” station has a pointer to a table of message strings, and will randomly play one these messages each time it wakes up. The strings themselves will be stored in the Arduino’s flash (program) memory to allow a larger number of longer messages than just the typical one or two character station call-signs. The wake up interval will be randomly selected between a minimum and maximum interval.

Upon entry to IDLE state, an ambience station will choose the next string to play and the time in millis() at which to play it. When the designated time comes, the ambience station will transition to RING\_WAITING. The choose\_next\_ringer selection logic will only allow an ambience station to transition RING\_WAITING -> RING\_PLAYING if all “regular” stations are in state IDLE.

If the sketch defines more than one ambience station, each ambience station will individually choose its random start time and which message to play next. If a another ambience station is playing when this station becomes ready to play, the sketch will let the first station’s message complete before starting the second. Internally, this is done through the normal IDLE -> RING\_WAITING -> RING\_PLAYING -> IDLE state sequence.

Any regular station leaving state IDLE (being called or going off-hook) will immediately force any ambience station in state RING\_PLAYING back to state IDLE, terminating the current ambience message, resetting the random timers and making a new message choice. The thinking here is that the ambience buzzer may be playing a longer message that could be distracting, so it is better to shut it off immediately.

Similarly, if any of the regular stations leaves state IDLE while any ambience station(s) are in RING\_WAITING state, the ambience stations will stay RING\_WAITING until all of the regular stations return to state IDLE (i.e. all calls are ended) before becoming eligible to play. It seems appropriate to give force the next ringer selection logic to wait a bit longer before allowing the ambience to resume, so there will be a separate timer to keep “ambience” messages farther apart.



---

### 4.3.3. Buzzers

We have used this sketch to drive a variety of sound-generating hardware from piezo-electric buzzers to relay-based telegraph sounders. In a quiet room, it may be possible to drive a piezo transducer directly from an active-low Arduino output. But in most cases an external driver circuit is needed to protect the Arduino from the speaker power supply voltage or to provide more “oomph” for the sound. For this reason, we want individual selection of active HIGH or LOW on each buzzer circuit. The MRCS “Morse Code Buzzer Board” outputs provide up to 500mA at up to 50 Volts via Darlington drivers, which is more than adequate for most vintage sounders, and it works with piezo-electric and mechanical buzzers as well.

### 4.3.4. Off-hook inputs

The local hook switch in most telephones is more suited being wired so that the Arduino input is floating when on-hook and grounded when the handset is off-hook. But given the variety of cannibalized phone gear which might be deployed on a model railroad phone system, there is no reason not to support individual selection of active HIGH or LOW on each “off-hook” input.

### 4.3.5. Called inputs

Most commonly, the “called” inputs will be active LOW normally open connections -- the calling station will pull the “called” input to ground, and the Arduino’s internal pull-up resistor will float the input HIGH when not called. But again, for flexibility, each input will have individual control of the “called” state HIGH or LOW.

In addition, there are two main styles of telephone systems in use for model railroads: multi-line systems based on office style PBX gear or simpler single party-line systems. For multi-line systems, it is common to have the called station ring either until it is answered or until the calling party hangs up.

On party line systems, it is often simpler to have a set of momentary station call buttons on the dispatcher’s desk. When the button is pressed, the corresponding station begins to ring, and continues to ring until it is either answered or a time limit (say 2 minutes) is reached.

We will support selection of the “momentary” or “continuous” behavior on a per-line basis. Each station will have an individual timeout value. Zero will indicate a “continuous” style station that rings until the “called” input goes inactive or the “answered” goes active. Non-zero will give a programmable ring duration for “momentary” stations.

## 4.4. SOFTWARE CONFIGURATION

Open the file `station_buzzers.ino` in Arduino IDE (Integrated Development Environment) and the tab `STATION_BUZZERS`. This file is available on Steve’s GitHub page: [https://github.com/stevew1154/station\\_buzzers](https://github.com/stevew1154/station_buzzers)

The following is part of the sketch from lines 40 – 157, the Rev configuration starts at line 76. The active entry for A6, A7 must be `ANALOG_LOW`, Note `ANALOG_LOW` has a un under-bar in it.

---

```

// stations
//
// This table defines the stations on the layout.
//
// Normal stations have level-sensitive "called" inputs, and will continue ringing until
either
// answered or until the caller hangs up. To define a "normal" station, set the
station_type to
// STATION_NORMAL, and define the "buzzer", "called" and "off_hook" pins. The timeout
value is
// ignored.
//
// Momentary stations are defined by setting station_type to STATION_MOMENTARY, set the
buzzer,
// called, and answered pins. The timeout value should specify the timeout in seconds. If
set to
// zero, the station will ring forever or until answered.
//
// To define an "ambience" station, set the "station_type" to STATION_AMBIENCE, and
the "called" and
// "off_hook" pins to -1. The timeout will then define a "base" time between random
ambience
// messages. The sketch will pick a random interval between 2/3 and 4/3 of this base
time. It really
// doesn't matter what you set the "station_code" to, as that field will only show up in
debug
// messages.
//
// Use of Arduino "analog" pins
// =====
//
// On most Arduino variants, the analog input pins A0-A5 are also fully capable of being
digital
// input or output pins. However, the A6 and A7 pins are a mixed bag. On the Leonardo and
other
// ATMEGA 32u4 based boards, A6 and A7 are muxed with other digital input pins so you
probably
// don't want to use them by the names A6/A7.
//
// On the Uno, Mini, Nano and probably most other ATMEGA-328P variants A6 and A7 are
distinct
// pins, but lack any sort of digital hardware, and therefore can only be used as analog
// inputs. This sketch will allow you to use A6 and A7 of the 328-based Arduinos as

```

```

either "called"

// or "off_hook" inputs, but you must specify "active" as ANALOG_LOW or ANALOG_HIGH
instead of LOW

// or HIGH. Also, your circuit must provide an external pull-up (ANALOG_LOW) or pull-down
// (ANALOG_HIGH) resistor. I.e. if your called switch connects pin A6 to ground, specify
it as

// A6,ANALOG_LOW in the table and provide an external pull-up resistor (1K Ohm should
work well).

// MRCS Buzzer Board rev 2
//
// The second-generation buzzer board supports 7 stations plus an ambience buzzer
#undef MRCS_REV2_TABLE
[This is the standard Rev 3 table, make this line above "#DEFINE", note you will need to
change the pin mapping and use your station codes instead of AA, BB etc.]
#ifdef MRCS_REV2_TABLE

const struct Station_Info stations[] = {
// buzzer called off_hook timeout station
// station_type, pin,active, pin,active, pin,active, seconds, code
{ STATION_MOMENTARY, 13, HIGH, A0, LOW, A6, ANALOG_LOW, 0, "AA" },
{ STATION_MOMENTARY, 12, HIGH, A1, LOW, 7, LOW, 0, "BB" },
{ STATION_MOMENTARY, 11, HIGH, A2, LOW, 6, LOW, 0, "CC" },
{ STATION_MOMENTARY, 10, HIGH, A3, LOW, 5, LOW, 0, "DD" },
{ STATION_MOMENTARY, 9, HIGH, A4, LOW, 4, LOW, 0, "EE" },
{ STATION_MOMENTARY, 8, HIGH, A5, LOW, 3, LOW, 0, "FF" },
{ STATION_MOMENTARY, 0, HIGH, A7, ANALOG_LOW, 2, LOW, 0, "GG" },

// This demonstrates an "ambience" station which will buzz one of the random ambience
messages

// at a random time between 2/3 and 4/3 of the "timeout_sec". This station doesn't need
// "answered" or "called" pins so they are set to -1. Also, the "station code" is
ignored.
{ STATION_AMBIENCE, 1, HIGH, -1, LOW, -1, LOW, 60, "MM" },
};

#endif

// David Parks Cumberland West
//
// This layout has 5 "normal" stations that ring until answered or until the caller hangs
up
#define DAVID_PARKS_TABLE
#ifdef DAVID_PARKS_TABLE

```

---

```

const struct Station_Info stations[] = {
// buzzer called off_hook timeout station
// station_type, pin,active, pin,active, pin,active, seconds, code
{ STATION_NORMAL, 8, HIGH, A0, LOW, 2, LOW, 0, "ND" }, // Viaduct
{ STATION_NORMAL, 9, HIGH, A1, LOW, 3, LOW, 0, "GE" }, // Evitts
{ STATION_NORMAL, 10, HIGH, A2, LOW, 4, LOW, 0, "KY" }, // Keyser
{ STATION_NORMAL, 11, HIGH, A3, LOW, 5, LOW, 0, "CO" }, // McKenzie
{ STATION_NORMAL, 12, HIGH, A4, LOW, 6, LOW, 0, "P" }, // Piedmont

// This demonstrates an "ambience" station which will buzz one of the random ambience
messages
// at a random time between 2/3 and 4/3 of the "timeout_sec". This station doesn't need
// "answered" or "called" pins so they are set to -1. Also, the "station code" is
ignored.
{ STATION_AMBIENCE, 13, HIGH, -1, LOW, -1, LOW, 60, "DS" }, // Dispatcher
};
#endif

// Dave Adams D&RGW
//
// This layout has 6 momentary stations, and no ambience buzzer. It is built on the MRCS
Morse Buzzer
// board v1.0, using an Arduino Pro Mini (Atmel 328P), and in order to use pin 13 as an
input for "RO",
// Seth Neumann removed the built-in LED on D13.
//
// This sketch should work equally well on a Nano or a Leonardo with the same caveat: D13
is hard to
// use as an input without either removing the on-board LED circuit or adding a pull-up
resistor.
#undef DAVE_ADAMS_TABLE
#ifdef DAVE_ADAMS_TABLE
const struct Station_Info stations[] = {
// buzzer called off_hook timeout station
// station_type, pin,active, pin,active, pin,active, seconds, code
{ STATION_MOMENTARY, 11, HIGH, A0, LOW, 2, LOW, 30, "DW" }, // West Durango
{ STATION_MOMENTARY, 10, HIGH, A1, LOW, 4, LOW, 30, "HF" }, // Hesperus
{ STATION_MOMENTARY, 9, HIGH, A2, LOW, 7, LOW, 30, "CA" }, // Cima Summit
{ STATION_MOMENTARY, 6, HIGH, A3, LOW, 8, LOW, 30, "MX" }, // Mancos
{ STATION_MOMENTARY, 5, HIGH, A4, LOW, 12, LOW, 30, "DJ" }, // Dolores

```

---

```
{ STATION_MOMENTARY, 3, HIGH, A5, LOW, 13, LOW, 30, "RO" }, // Rico
};
#endif
const int num_stations = sizeof(stations) / sizeof(stations[0]);

// The messages played by the ambience sations are defined here. We are playing Arduino
AVR tricks

// here to place the strings themselves in the Arduino's larger program memory.
//
// You can add as many messages as you need, limited only by the code space in your
Arduino. Adding
// a message is a two-step process -- first, you need to add a declaration for the string
itself
// like the declaration for "ambience_00" below. Simply copy/paste the last string and
bump the
// number. Then you have to add the new string to the "ambience_messages" array.
const char ambience_00[] PROGMEM = "OS No 2 by ND at 14:45";
const char ambience_01[] PROGMEM = "DS GE Copy 3 West";
const char ambience_02[] PROGMEM = "Check box 13, the dispatcher is an idiot.";

const char * const ambience_messages[] PROGMEM = {
ambience_00, ambience_01, ambience_02,
};

const int num_ambience_messages = sizeof(ambience_messages) /
sizeof(ambience_messages[0]);
```

---

## 5 TESTING

---

Testing your Morse Code Buzzer Controller is quick and simple:

1. Mount the board securely on standoffs or in a piece of 2.75" DINRail
2. Apply 5VDC (Rev 1.0, 2.0) or a 6-12 V regulated supply (Rev 3.0) regulated power with a 2.1mm center positive plug to Con3
3. Connect an LED with 470 ohms in series from your supply voltage to output 1 (cathode – the short lead) to output 1.
4. Apply a ground momentarily to input 1.
5. LED should flash with station 1's code. Ground Stop 1 to stop flashing.
6. Repeat for all defined stations.

DRAFT

## 6 INSTALLATION AND CONNECTIONS

### Morse Code Buzzer Controller Connections

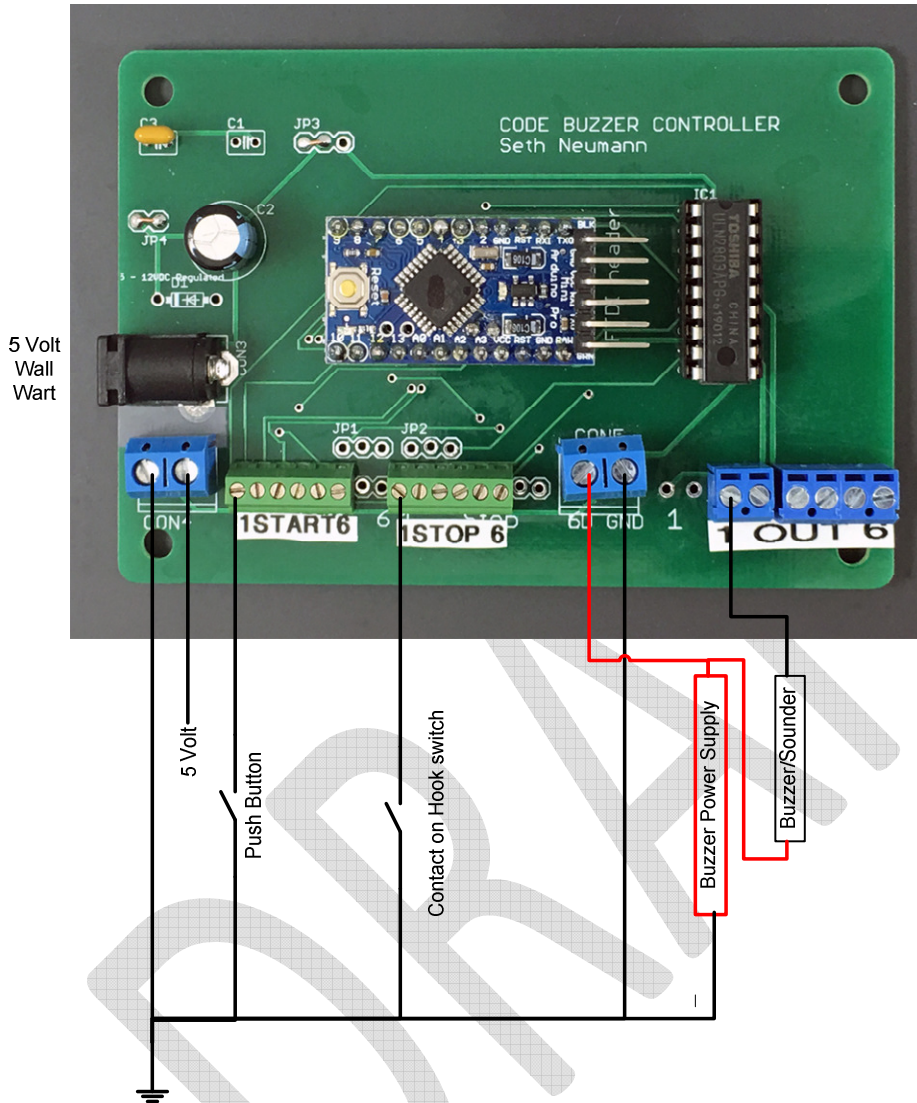
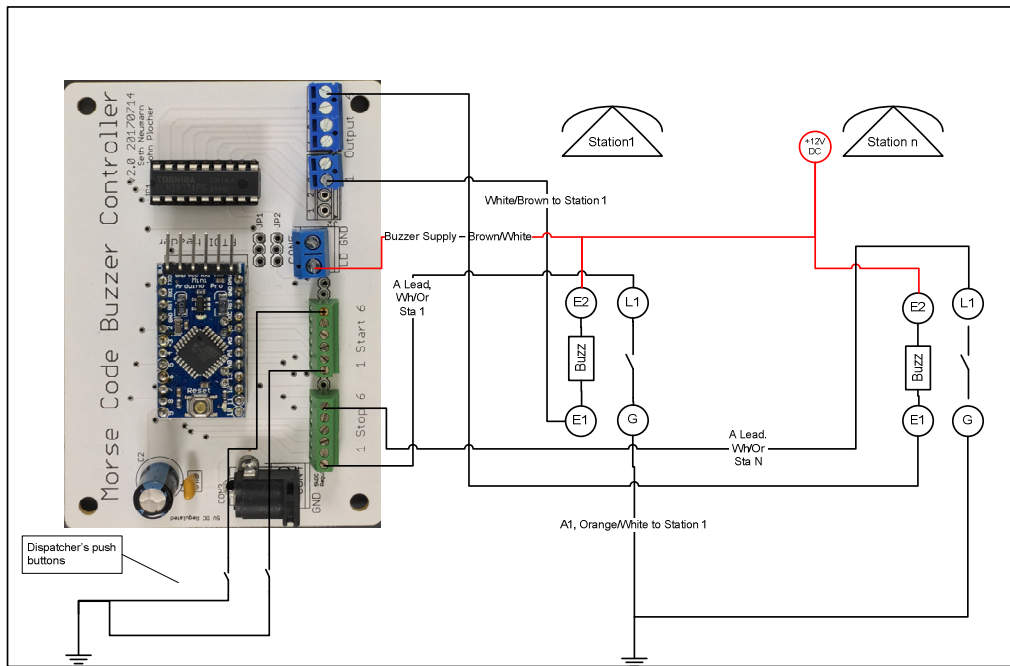
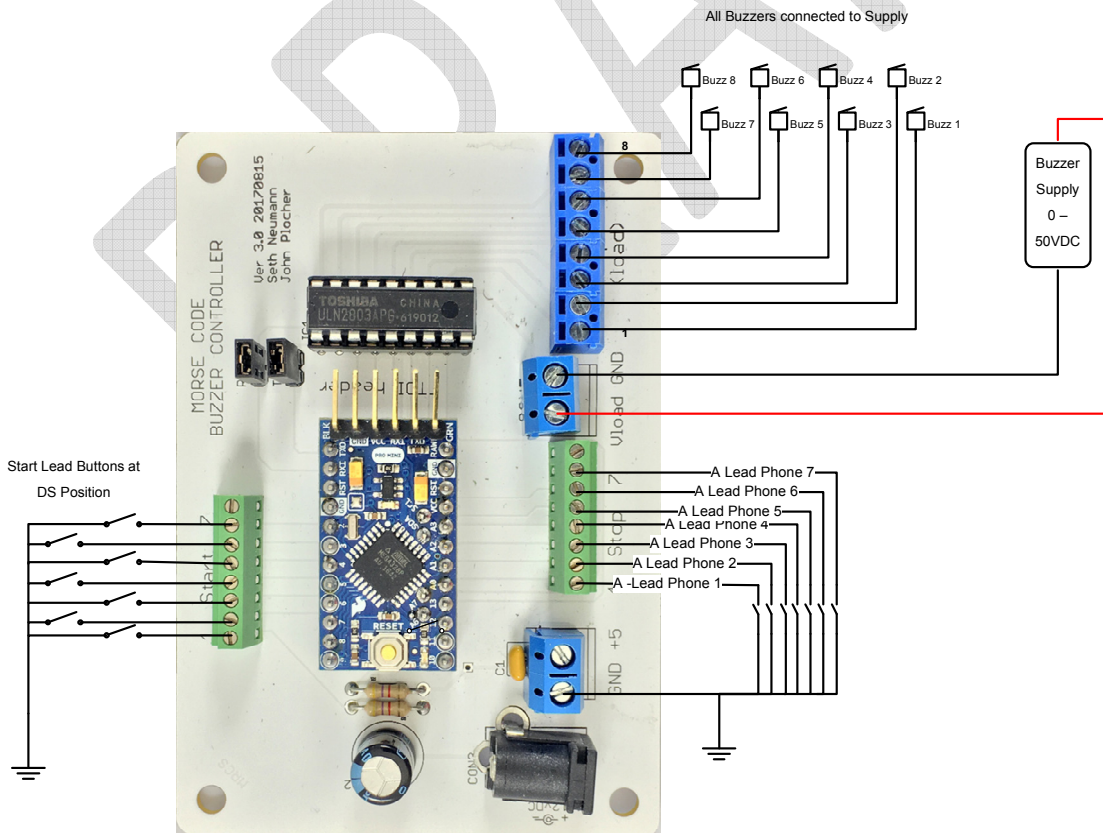


Figure 8 - Connections for Rev 1 Boards



**Figure 9 - Rev 2 Board with stations 1 and 6 connected showing internal phone connections to speech network**



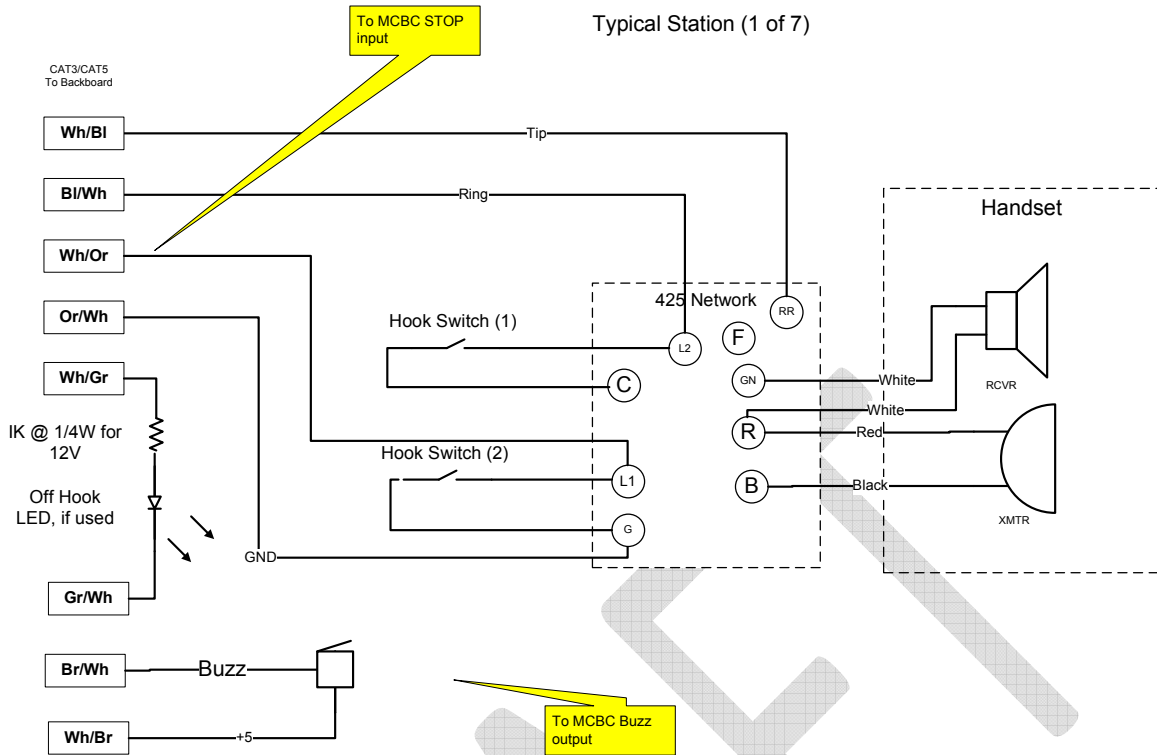
**Figure 10 - Rev 3 board with connections**



---

General Considerations for all boards and connections:

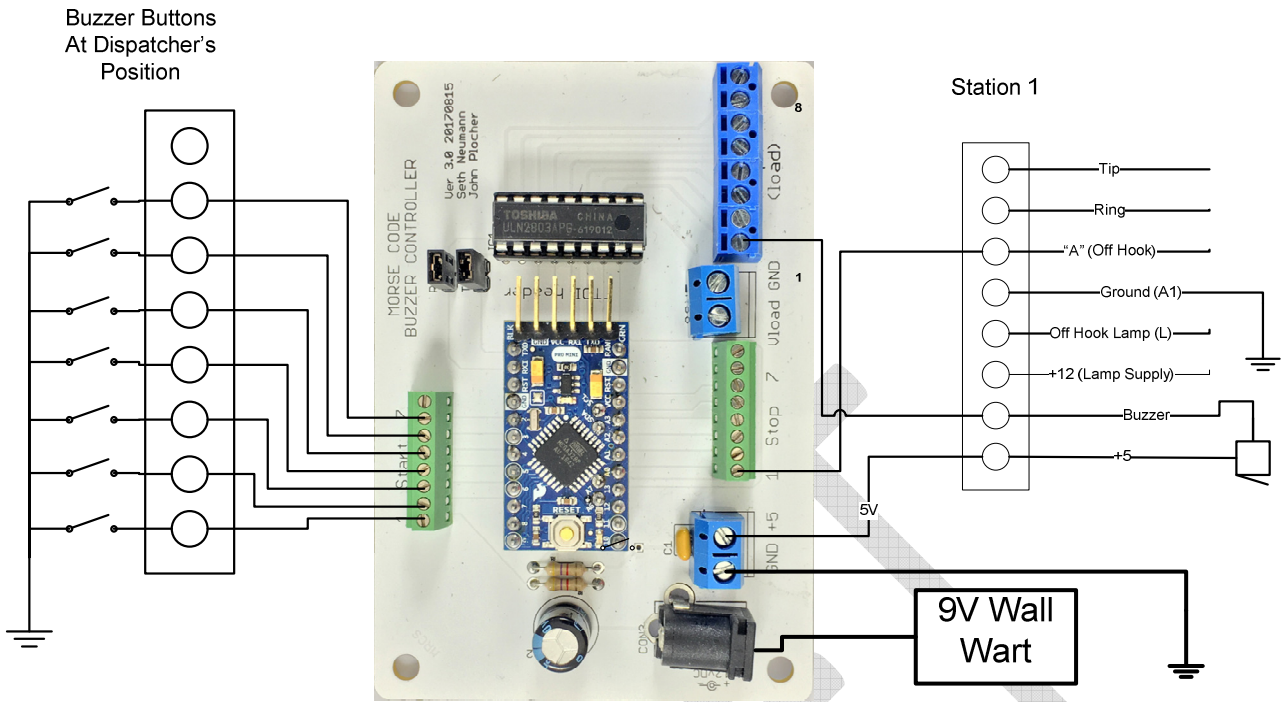
- Input voltages must be no more than 5 volts, so try to use “dry” contacts, that is contacts with no other connections. When using CAT5 wire, I use the telephone convention of WHITE/ORANGE is the switched input lead (ground when phone is off hook), and Orange/White is connected to system ground.
- If you are sharing the contact with some other function, ground must be tied to the board ground and I recommend using a diode (1N4001 or equivalent) to isolate the legs. If your existing circuit has off hook showing 5V, you can reverse the sense of the input (except for A6 and A7) in the configuration table in the sketch. A6 and A7 have pull up resistors, so this isn't a good trick for circuits 6 and 7.
- The outputs provide a ground (Sink) when the output is on. Each output can ground a load of up to 0.5A at up to 50V. The ground must be connected to the ground terminal on CON5 and the Supply output must be connected to Load (LD) on CON5 in order to use the internal clamp diodes on the ULN2803. This critical when driving inductive loads such as sounders, relays and motors. I use the telephone convention of BROWN/WHITE to Buzzer Supply and WHITE/BROWN to the ground side of the buzzer (the side connected to the Morse Code Buzzer Controller).
- Interfaces for non-compliant inputs. If you need to use a start (called) or stop (off-hook) input that is either greater than 5V with respect to the Morse Code Buzzer Controller's ground or negative (traditional Key Telephone System), the you can use either a relay or an opto-isolator to provide an isolated input to the MCBC. There is a wide selection of both available For relays, we like the OMRON G5V-2-H-DC24 or equivalent, you should be able to find this from inexpensive off-shore sources for around \$1.00, connect one side of the coil to the -24V supply and the other to the “A” lead. If you want use an opto-isolater, the venerable 4N25 is a good start, for about \$0.25, put it in series with a 2.2 K ohm resistor from -24 to the “A” lead and connect the output from ground to the MCBC input.



Notes:

- 1 – Hook switch has 2 contacts, (1) takes the phone off hook, (2) Stops the Morse Code Buzzer Controller when off hook
- 3 – Some Networks have extra tie points, E1,E2 which can be used to make connections. Also A and K can be used for miscellaneous connections (they are 2 sides of a capacitor normally part of the ringer circuit which we're not using)

**Figure 11 - Typical Telephone Connections**



**Figure 12 - MCBC wiring to station**

DRAFT