The "Thereping" is a digital musical instrument that plays sounds based on a combination of the position of your hand and some pushbutton switches.

It uses a combination of a sonar sensor and a microcontroller to allow the user to play interesting melodies without requiring any musical skills. When joined together with a "Thereclock" sync unit, multiple Therepings can all play together and create interesting music, again without requiring any musical expertise on the part of the players.

THE CREATION OF THE THEREPING

In the Beginning

The quest for a new and interesting musical instrument began when The Robot Group (an Austin, TX-based Robotics, Art, and Technology club) was selected to participate in the First Night Austin celebration with a proposal they entitled "The Robot Theremin Band." The idea was to use theremins (see sidebar: "What's a Theremin?") and robots together to create an experimental, musical spectacle. However, the Theremin is a difficult instrument to play *well* as it requires a high degree of accurate physical control (i.e., hand/body position), as well as a good knowledge of music theory. As far as expertise is concerned, it has much in common with a violin in that, if played well, it can be beautiful. If *not* played well, it can be ... unpleasant at best.

After reading a description of the proposal on The Robot Group's mailing list, I posted that "a group of technically minded folks, with no musical training, trying to play Theremins would sound roughly like someone attempting to murder a large number of cats with a mallet." Since this might be more apt to repel people than attract them, I suggested we consider crafting some sort of electronic musical instrument instead.

Ahhhhh ... FREQ OUT!

In mid November 2005, at a meeting held at my house, Don Colbath (one of the group members who actually owns a real

What's a Theremin?

■ The Theremin or Thereminvox is one of the earliest fully electronic musical instruments. Invented in 1919 by Russian Léon Theremin, the Theremin is unique in that it requires no physical contact in order to produce music and was, in fact, the first musical instrument designed to be played without being touched.

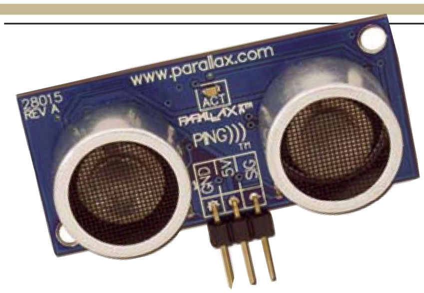
The instrument consists of a box with two projecting antennas around which the user moves his or her hands to play. To control the Theremin, the musician stands in front of the instrument and moves his or her hands in the proximity of two metal antennas — the distance from the antennas determining frequency (pitch) and amplitude (volume).

Small movements of the hands can create a tremolo or vibrato effect. Typically, the right hand controls the pitch and the left hand is used for the volume, although some play left-handed.

Based on the principle of heterodyning oscillators, the Theremin generates an audio signal by combining two different — but very high — frequency radio signals. The capacitance of the human body close to the antennas causes pitch changes in the audio signal, in much the same way that a person moving about a room can affect television or radio reception.

By changing the position of the hands relative to the vertical antenna, a performer can control the frequency of the output signal. Similarly, the amplitude of the signal can be affected by altering the hand's proximity to the looped antenna. *The information above is excerpted from "Wikipedia," located at www.wikipedia.org*

BY VERN GRANER



■ FIGURE 1. PING)))™ Sensor. The Parallax PING))) sensor uses ultrasonic soundwaves to determine the distance to an object and returns a value in milliseconds that is quite accurate. It only requires three connections: 5V+, GND, and a single TTL level pin for both sending and receiving data.

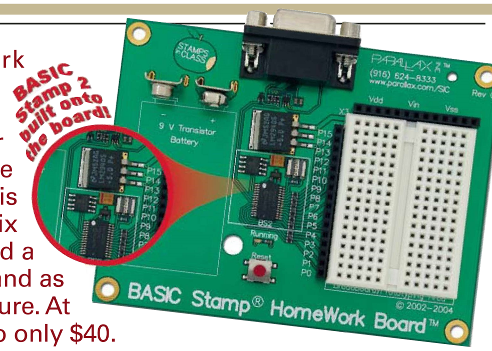
Theremin) was toying with a small test system I was using to demonstrate the Parallax BASIC Stamp II connected to a Parallax PING)))™ sonar sensor (Figure 1). The test system was running a program that would show the distance of an object from the PING))) sensor on an LCD display. Don remarked that if the distance reading could somehow be sent to an audio oscillator, it might be possible to make a sound similar to a Theremin!

Intrigued, I wrote some very simple PBASIC code that would measure the distance reported by the sonar sensor and store it in a variable. Then, rather than try to coble together an oscillator, I just stuffed that value into the PBASIC command used to create sounds (the aptly named FREQOUT) to have the BASIC Stamp itself produce sound directly as shown below.

```
PING    PIN 4    ' Parallax PING)))
SPEAKER PIN 9    ' Speaker
SAMPLE  VAR Word ' Store result

AGAIN:
  PULSOUT PING, 5
  PULSIN PING, 1, sample
  FREQOUT SPEAKER, 100, sample
GOTO AGAIN
```

When we ran the code, the unit began to play sounds at a pitch in proportion to the distance of your hand to the PING))) sensor! A new musical instrument, the "Thereping" was born!



■ FIGURE 2. The Parallax HomeWork board. These are self-contained BASIC Stamp II microcontrollers with a built-in breadboard and DB9 serial interface for programming. The HomeWork boards are only sold in a 10 or 20 piece pack, but this would give us enough units to have six instruments, one master sync clock, and a few spares to use for experimentation and as backups in the event of a component failure. At quantity 10, the per-board price drops to only $40.

Parts is Parts

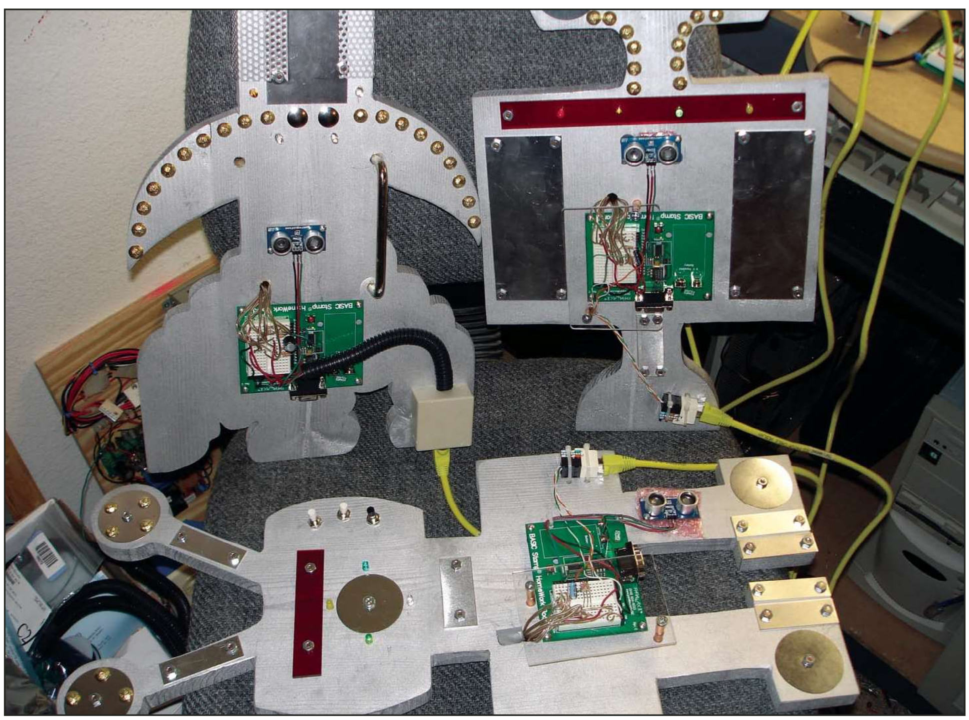
Now that we had a proof of concept, it was time to fabricate a bunch of instruments. Denise Scioli, a member of the The Robot Group, agreed to contact Parallax and order all the parts we needed. Since the prototype instrument was based around the BASIC Stamp II microcontroller from Parallax, and we planned to build six instruments, we decided on the Parallax HomeWork Board (Figure 2) to act as the base, as it's available in 10-packs, at a good price.

Denise placed the order and soon we were in possession of 10 HomeWork boards and 10 PING))) sonar sensors. In preparation for the project, Denise also created and decorated some wooden cutouts (Figure 3) in fanciful shapes of rocket ships and robots that were to be used as a base to hold the HomeWork boards and provide a familiar playing paradigm (i.e., "guitar like"). I attached one of the HomeWork boards and a PING))) sensor to one of the boards to get an idea for how the instrument would look and feel.

Refining the Idea

Though we had one prototype circuit and mount for our fledgling Thereping — and this prototype would make sound — it was not necessarily a *pleasant* sound. Also, like an actual Theremin, it required

■ FIGURE 3. Three Therepings.



some skill in music and physical control to create consistent musical notes and/or melodies. More importantly, if we built a number of these units, they would suffer the same weakness from which a group of "real" Theremins would have suffered (i.e., the "malleted felines" syndrome).

We needed the ability to place musically "unskilled" folks into the role of "musician." With just under six weeks left, the problem was broken down into pieces to solve one at a time.

What Can Go Wrong?

For our purposes, there are two fundamental things that you can do wrong when playing an instrument:

1. You could play an incorrect note.

2. You could play a note at the wrong time.

The "incorrect note" issue was attacked first. Since there are scales in which any of the notes, when played together, will sound correct, I conducted an experiment using a blues scale in the key of C.

On the prototype instrument, I created code that would instruct the PING))) sensor to fetch a distance reading, then check the returned reading against a range of values to determine what note should be played. The first code looked like that in Listing 1.

Though not very efficient, this code did "brute-force" determine the hand distance above the sonar sensor and then make it so the instrument would only play corresponding notes that would be valid in the blues scale. When this code is run, you can move your hand above the sensor and the note that corresponds with your hand position is played. However, the timing between the notes is fixed at 100 ms (roughly 16 notes at 120 bps) and your hand position is absolute (i.e., you cannot change the distance your hand must travel to create specific notes).

At the next meeting of The Robot Group, Eric Lundquist — another long-time group member (and programmer by trade) — had some great ideas to optimize the program. We refined the code in a number of ways to make the instrument more efficient and flexible.

First, we broke the space above the PING))) sensor into discrete "zones" that could be calibrated to make the sensing area adjustable and allow the notes to be scaled to different octaves or pitches. This was accomplished by determining the highest and lowest PING))) sensor readings in milliseconds acquired while comfortably moving our hand over the sensor (from a bit less than one inch to about eight inches) and setting those values into constants:

```
LOWPos   CON 100   ' MS Value for closest note
HIGHPos CON 800    ' MS Value for furthest note
```

Then, we set a constant to hold the number of notes into which we wanted the range divided:

```
NNotes CON 7 ' Number of notes in the scale
```

This divided the airspace above the sensor into seven one-inch zones. Now you could tell in which zone the user's hand was detected by using the simple formula:

```
Zone = (HighPos-LowPos) / NNotes
```

In order to make the code easier to read and more intuitive, I used the PBASIC CON command to create an "alias" of the Hz values for each note in a 12-note chromatic scale. This alias would reflect the note's "name," its modifier (sharp or flat), and the octave. The naming convention I chose was:

<note letter><sharp/natural><octave>

For example, the notes from C6 through C7 would look like:

```
Cn6   CON 1047
Cs6   CON 1109
Dn6   CON 1175
Ds6   CON 1245
En6   CON 1319
Fn6   CON 1397
Fs6   CON 1480
Gn6   CON 1568
Gs6   CON 1661
An6   CON 1760
As6   CON 1865
Bn6   CON 1976
Cn7   CON 2093
```

Listing 1

```
AGAIN:
  PULSOUT PING, 5
  PULSIN PING, 1, sample
' C blues scale       C - Eb - F - Gb - G - Bb - C
IF SAMPLE1 > 1000 AND SAMPLE1<1047 THEN FREQOUT SPEAKER,100,1047 ' C
IF SAMPLE1 > 1047 AND SAMPLE1<1245 THEN FREQOUT SPEAKER,100,1245 ' Eb
IF SAMPLE1 > 1245 AND SAMPLE1<1396 THEN FREQOUT SPEAKER,100,1396 ' F
IF SAMPLE1 > 1396 AND SAMPLE1<1480 THEN FREQOUT SPEAKER,100,1480 ' Gb
IF SAMPLE1 > 1480 AND SAMPLE1<1568 THEN FREQOUT SPEAKER,100,1568 ' G
IF SAMPLE1 > 1586 AND SAMPLE1<1864 THEN FREQOUT SPEAKER,100,1864 ' A#/Bb
IF SAMPLE1 > 1864 AND SAMPLE1<1976 THEN FREQOUT SPEAKER,100,1976 ' B
IF SAMPLE1 > 1976 AND SAMPLE1<2093 THEN FREQOUT SPEAKER,100,2093 ' C7
GOTO AGAIN
```

I did consider using the BASIC Stamp to calculate the Hz values for each note in real time, but it seemed like a lot more work and I was afraid doing so would require additional processing power that we might need later. Subsequently, I just pre-calculated the values for each note.

Next, the original brute force method of determining the appropriate note to play was discarded in favor of a more elegant approach using the PBASIC LOOKUP command to pick a value from a range. The new completed Thereping program worked like this:

1. Check the distance to the hand above the PING sensor.

```
PULSOUT PING, 5          ' Send a ping out
PULSIN PING, 1, sample   ' store response in "sample"
```

2. Determine in which zone the hand was located, and store it in the Note variable.

```
Note = (SAMPLE - LowPos) / Zone
```

3. Use the Note number to LOOKUP the correct note frequency and store that value in the FREQ variable.

```
' C blues scale       C - Eb - F - Gb - G - Bb - C
LOOKUP NOTE, [Cn6, Eb6, Fn6, Gb6, Gn6, Bb6, Cn7], FREQ
```

4. Play the note with the FREQOUT command.

```
FREQOUT Speaker,100,FREQ
```

After Step 4, just go around in a loop and do it all again. Using this new code, our instrument would efficiently play only notes that would be correct in a specific scale. However, we still had no control of the note's duration or of its timing in relation to other instruments or an external tempo.

Plays Well With Others

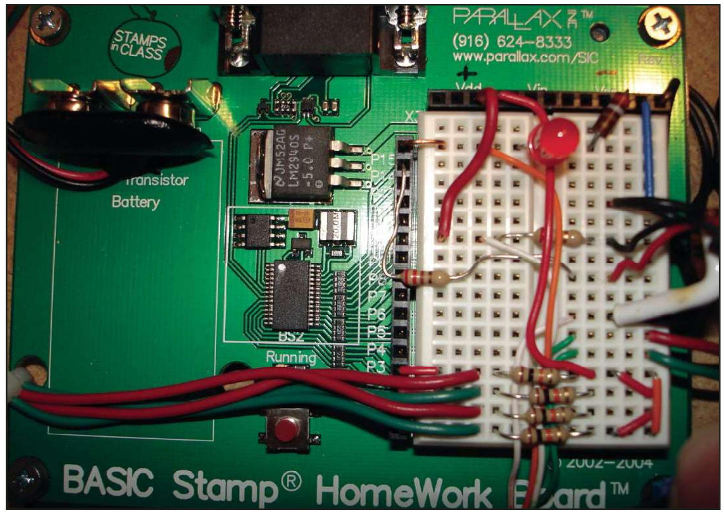
Since the point of the project was to have everyone play *together*, we needed to attack the second thing that could go wrong, namely "playing the note at the wrong time." I figured I could simply program each of the Therepings to look for a "central clock" source before playing a note, to synchronize all the instruments.

Originally, I considered using an LED IR beacon as a sync source. The design consisted of a tall tower (dubbed a "tempo tower") that would allow IR LEDs to be pointed downward to the stage where the instruments would be during a performance. Since our stage was to be a trailer in a parade and a small outdoor pavilion, this solution at first appeared feasible. However, we were slated to play on the pavilion in the daytime when natural UV/IR light would drown any sync signal from the IR LED sync source.

Also, the musicians would need an uninterrupted line-of-sight to the tempo tower if continuous sound was to be maintained. If the players moved about, the Therepings "view" of the tower could be blocked. It started to become clear we needed to re-think our sync.

Where a Clock? Thereclock!

Though the benefits of a wireless form of sync signal would be large (i.e., no wires to tangle, greater mobility for the musicians, etc.), the downside would be that each instrument would need to contain an on-board power source, speaker system, and associated audio amplifier (or even more complicated, their own radio transmitter!).



■ FIGURE 4. Thereclock Prototype.

Also, in order to be heard in a parade atmosphere, the Therepings would have to be equipped with some pretty beefy amps/speakers that would require some equally hefty batteries. In the end, the additional costs and associated construction time pretty much ruled out going wireless.

Since we had planned to have a central P.A. system, it made sense to simply fall back to a hard-wired approach. I created an experimental master clock system (dubbed the "Thereclock") using one of the Parallax HomeWork boards and some very simple code to toggle pin 15 of the board HIGH then LOW.

```
AGAIN:
  HIGH 15
  PAUSE 250
  LOW 15
  PAUSE 250
GOTO AGAIN
```

I added an LED to pin 15 on the Thereclock board, so I could verify the code was indeed toggling the output, and a jumper from pin 15 on the Thereclock Stamp to pin 15 on the Thereping prototype (Figure 4).

Note that they were sharing the same power supply, so they already had a common ground. Once complete, I modified the Thereping code to include a pin definition for SYNC and added this new line just above the existing sound-producing command as shown.

```
SyncWait:
IF SYNC = 0 THEN SyncWait

FREQOUT Speaker,100,FREQ
```

I placed my hand over the PING))) sensor, and the notes began to fire off in precise accordance with the blinking of the LED on the Thereclock! Houston, we have SYNC!

Play it Again ... and Again ... and ...

It was now mid-December, and I had experimented with the prototype Thereclock and Thereping circuits for a while now. It began to get a bit fatiguing to hear the same exact note durations without any variation. It was interesting to hear for a short time, but a group of these instruments would sound very similar and it might be impossible to determine which instrument was playing. Even if you could, the sound produced would tend to be mostly redundant.

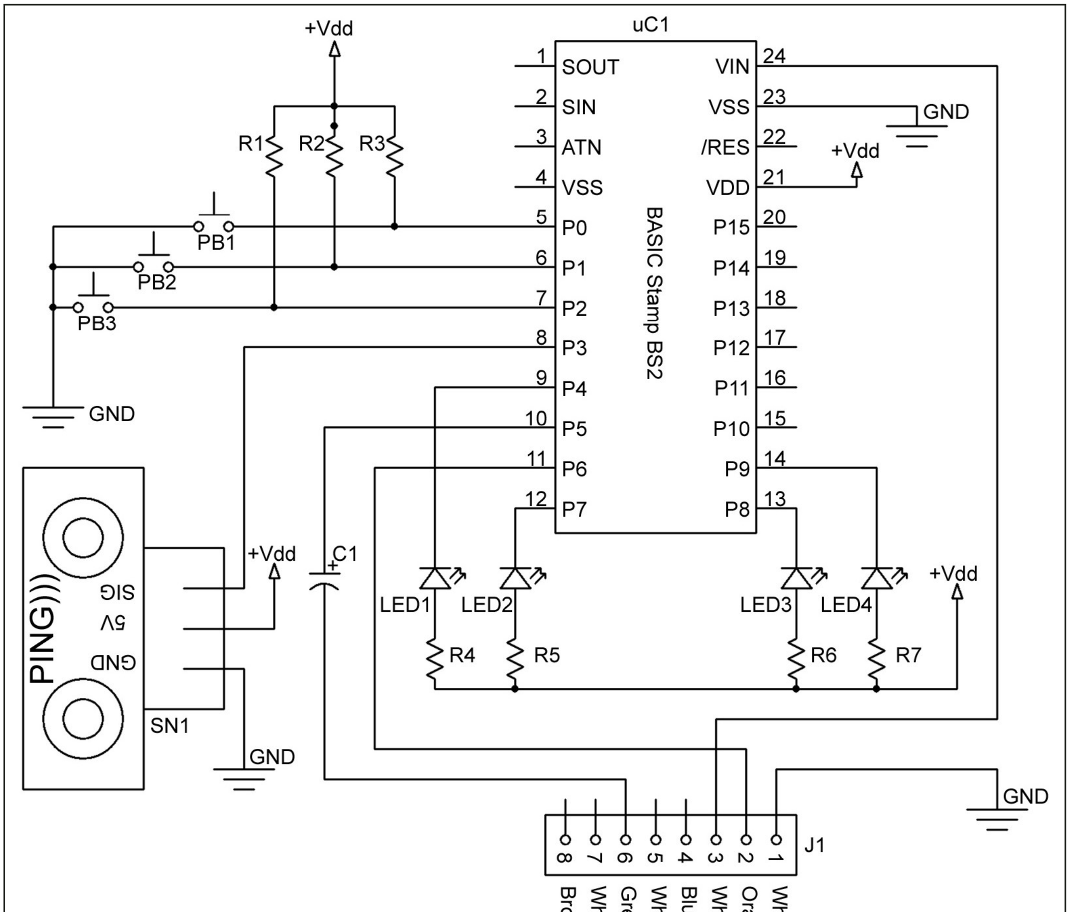
It would be more interesting and make for a more expressive instrument if the performer could choose to play 1/8th notes for a while, then play 1/4 notes, then switch to 1/16th notes whenever they wanted. To check for the selection of the performer, I added some buttons (PB1 through PB3) to the Thereping circuit (Figure 5).

I modified the code to check the buttons and alter the number of milliseconds the note would be played. But, what resulted was only the ability to change the note from filling the entire 1/4 note interval to playing an 1/8th note followed by an 1/8th rest! The note duration was shorter, but it was no longer filling the entire clock cycle.

For those of you familiar with musical terminology, the notes just started to sound staccato. So, it was clear that in order to play synchronized 1/8th notes, I would have to increase the clock frequency to at least 1/8th notes. To play 16th notes, I would have to increase it to 16th notes!

To do this, I needed a mechanism to determine where I was in the measure, so all the instruments could start and stop together on the first beat. I would also need to determine the first clock cycle to accurately divide the clock cycles back down to 1/8th and 1/4 notes when they were selected.

To illustrate this problem, imagine the clock is pulsing 16 times per 1/4 note. How do we determine which



■ FIGURE 5. Thereping Instrument Schematic.

Listing 2

```
DO WHILE BTN1=0    ' Check for "drone" button and play drone sound if pressed
  SyncWait1:
  BUTTON SYNC, 1, 255, 0, BtnWrk, 0, SyncWait1 ' Wait for a sync pulse

  Duration = Clock
  FREQOUT Speaker,Duration,Cn4/2, Cn4 'Play the note!
  PAUSE offset
  Duration = Clock
  FREQOUT Speaker,Duration,Cn4/2, Cn4 'Play the note!
  PAUSE offset
  Duration = Clock
  FREQOUT Speaker,Duration,Cn4/2, Cn4 'Play the note!
  PAUSE offset
  Duration = Clock
  FREQOUT Speaker,Duration,Cn4/2, Cn4 'Play the note!
  PAUSE offset
  Duration = Clock
  FREQOUT Speaker,Duration,Cn4/2, Cn4 'Play the note!
  PAUSE offset
  Duration = Clock
  FREQOUT Speaker,Duration,Cn4/2, Cn4 'Play the note!
  PAUSE offset
  Duration = Clock
  FREQOUT Speaker,Duration,Cn4/2, Cn4 'Play the note!
  PAUSE offset
  Duration = Clock
  FREQOUT Speaker,Duration,As4/4, As4/2 'Play the note!
  LOOP
```

of those pulses is the beginning of the 1/4 note? I decided that since I knew the interval of the clock pulses coming from the Thereclock, I would be able to trigger multiple notes on the Thereping at the onset of a clock pulse before looking to the sync input for a new clock pulse.

For an 1/8th note, I altered the code to play a note two times with a duration that was half the clock frequency before returning to wait for a new clock sync pulse. For the 16th note, it would play four notes at one-quarter the clock frequency (see flow chart in Figure 6).

Though not a perfect solution, it did allow the performer to decide on the duration of the notes and let the Therepings play distinctly different sounding musical themes.

Droning On and On ...

Now that we could play a fairly intricate solo on the instrument, we were again faced with the problem that many instruments playing similar sounds would tend to blend into just so much noise. Most people expect certain things from music, such as a rhythm section (i.e., bass and drums), an accompaniment of some type (i.e., piano, guitar, etc.), and a discernible lead melody. All I had, so far, was a group of instruments that could play the lead part.

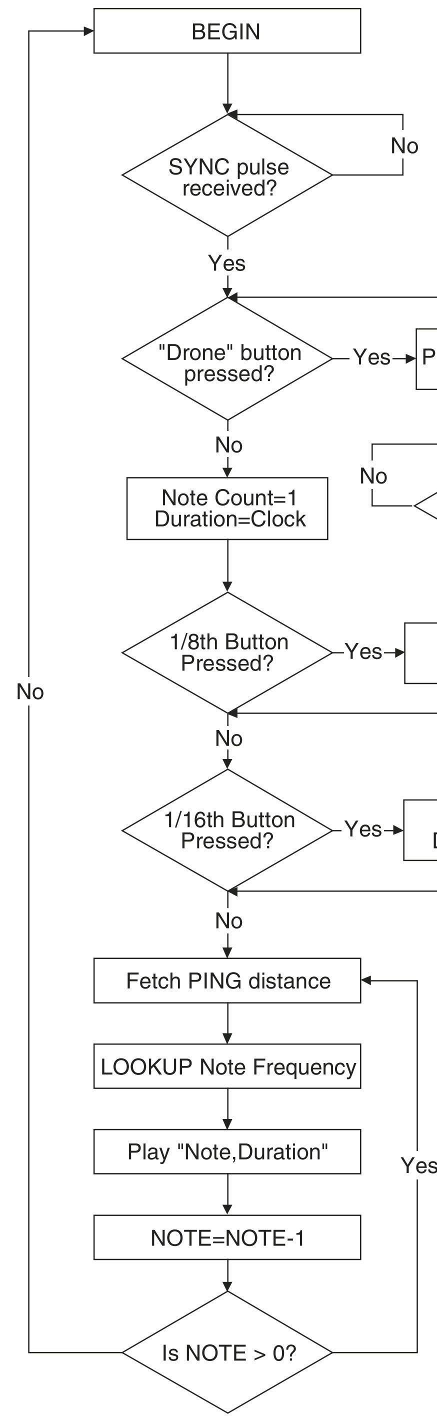
To create a more appealing sound, when the player of the solo part is done, they should be able to drop into the background and accompany or just enjoy what others were playing. Since we were in the same key throughout the song, certain notes could be played to act as a foundation to the music.

The bagpipe is an interesting instrument in that it has three horns (referred to as drones) that play a consistent chord of notes throughout every song. I thought I could create a sort of drone for the instrument using one of the buttons to tell the microcontroller that it should simply play a preset pattern of notes and ignore the sonar sensor altogether. Then, a player could rest their hand, dance, look around, or listen to others playing.

The instruments would also play something closer to what most people expect to hear in modern music. I added code that would detect the drone button and branch to a routine that simply repeats a series of notes in a bass line. The drone addition to the code is shown in Listing 2.

One thing to notice is that the note frequency values are divided by two in order to drop the frequency by an octave (i.e., Cn4/2). This alters the voice of the instrument to more closely resemble a bass guitar, creating a driving bass line to accompany other players who would be soloing.

I created a couple of variations on the drone part in order to change up the sound and loaded the alternate code into two of the five Therepings. I wanted as much variety in the final song as we could get. You may also notice this is a pretty sloppy way to accomplish this task. From a programming perspective, it would be much tidier to place the FREQOUT commands in a loop.



■ FIGURE 6. Logic Flowchart for the Thereping.

However, we were running out of time and most of the coding was done with an eye towards illustrating the function and allowing quick changes rather than optimizing form. Remember this if you decide to build your own Thereping — there are *lots* of opportunities for improvement!

Getting Wired

Now we had a proof of concept for syncing multiple

Thereclock and Therepings with Drum Set.

Pin	Color	Use
1	White/Orange	GND
2	Orange	SYNC
3	White/Green	V+
4	Blue	NC
5	White/Blue	NC
6	Green	AUDIO
7	White/Brown	NC
8	Brown	NC

TABLE 1

instruments using wire. We just needed a wiring method to easily and robustly link all the instruments to the central clock. I had quite a few Ethernet cables and RJ45 keystone jacks laying about, so I decided to try using regular CAT-5 cable to connect the Thereclock unit to the Thereping instruments. The four pairs in the CAT-5 cable would be plenty for all the needed signals and allow for future expansion if the need arose. See Table 1.

The final schematic for the Thereping shows the CAT-5 connector as the sole I/O point for the instrument providing everything that is needed to operate it.

Since part of our objective was to provide a spectacle, we added some LEDs to each Thereping. This would make interesting visual effects, act as an indicator that the unit was successfully receiving SYNC from the Thereclock, and could be used for diagnostics and development. The final schematic for the Thereping is shown in Figure 5.

Gimme a BEAT!

Okay, it's now near the end of December. The Thereping has the ability to play lead or solo parts, an accompaniment via a bass line, and we have a way to keep all the instruments in sync. The only thing missing in my mind was a BEAT. Since this was designed to be used in a parade, it seemed fitting to have a rhythm playing that would provide the background for the performance.

I researched sending MIDI from the BASIC Stamp and found it was trivial to create the MIDI output hardware connection. All that's required are two resistors and one I/O pin from the Stamp. I took the prototype Thereclock and added the resistors and then cut up an old MIDI cable to connect to the breadboard. I plugged the MIDI cable into the MIDI IN of the Yamaha DTXpressII sound module on my drum kit, and then went in search of some example code. It turns out that to make a BASIC Stamp send a simple MIDI command is surprisingly straightforward. The code in Listing 3 plays a constant 1/4 note at about 120 bps using the kick drum sound.

Since the MIDI output of the Thereclock would be sent to the drum machine, it seemed simplest to mount the Thereping directly onto the electronic drum set that held the MIDI module. This would allow me to play the drums to add fills over the straight MIDI beat coming from the Thereclock box. This central location would also make it as easy as possible to connect all the instruments to the Thereclock.

I had a 12" round wooden disk that would make a good base for the BASIC Stamp HomeWork board, as well as the RJ-45 jacks in the wall plate. The circular shape would make it fit in with the cymbals and allow attachment to a cymbal holder by simply drilling a single hole in the edge of the board. I built the prototype and tested it with the Therepings. It was successful in syncing up all the Thereping units!

However, it would be necessary to find a way to STOP and START the unit to designate the beginning and ending of songs. I took some PC case back planes and bent them to fit on the side of the unit facing the drummer, and then installed and labeled four pushbuttons to use for controls. Now I had the ability to START and STOP the song, and I also added the traditional whistle eight beat intro and outro that would normally be used by a drum major to start and stop a marching band.

Since the button's use is determined by software, this would also allow me to re-purpose the functions in the future (i.e., song select, tempo adjustment, etc.). I also removed the 1/4" right and left audio jacks and replaced them with a single 1/8" female jack so the

Listing 3

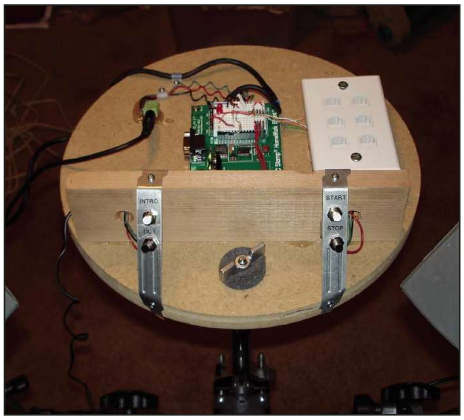
```
MidiOut   PIN 15                  ‘ MIDI Output
MidiBaud  CON $8000 + 12          ‘ 31.25 kBaud — open
Channel   CON 8                   ‘ MIDI Channel
NN        CON $90 | Channel       ‘ note on
NX        CON $80 | Channel       ‘ note off

AGAIN:
SEROUT MidiOut, MidiBaud, [NN,$24,$7f] ‘NoteOn  kick note# 36 at full velocity
PAUSE 300
SEROUT MidiOut, MidiBaud, [NX,$24,$7f] ‘NoteOff kick note# 36 at full velocity
PAUSE 300
GOTO AGAIN
```

audio could be sent directly to either a small computer speaker type amplifier or, using a 1/8" to RCA adapter, sent to the line input of a mixing console.

On the mixer, I panned the first three Therepings inputs to the left and the second set to the right, then added some echo and a touch of reverb. I didn't implement the audio out LPF circuit shown in the BASIC Stamp manual because the consensus from people who had heard the prototype is that the buzzy sound was more appealing than the duller sound that came out after the LPF was applied.

The finished schematic for the Thereclock is shown in Figure 7. I invited the The Robot Group members over for a rehearsal (Figure 8) and then we were ready for the debut performance before an expected crowd of well over 60,000 people! Yikes!

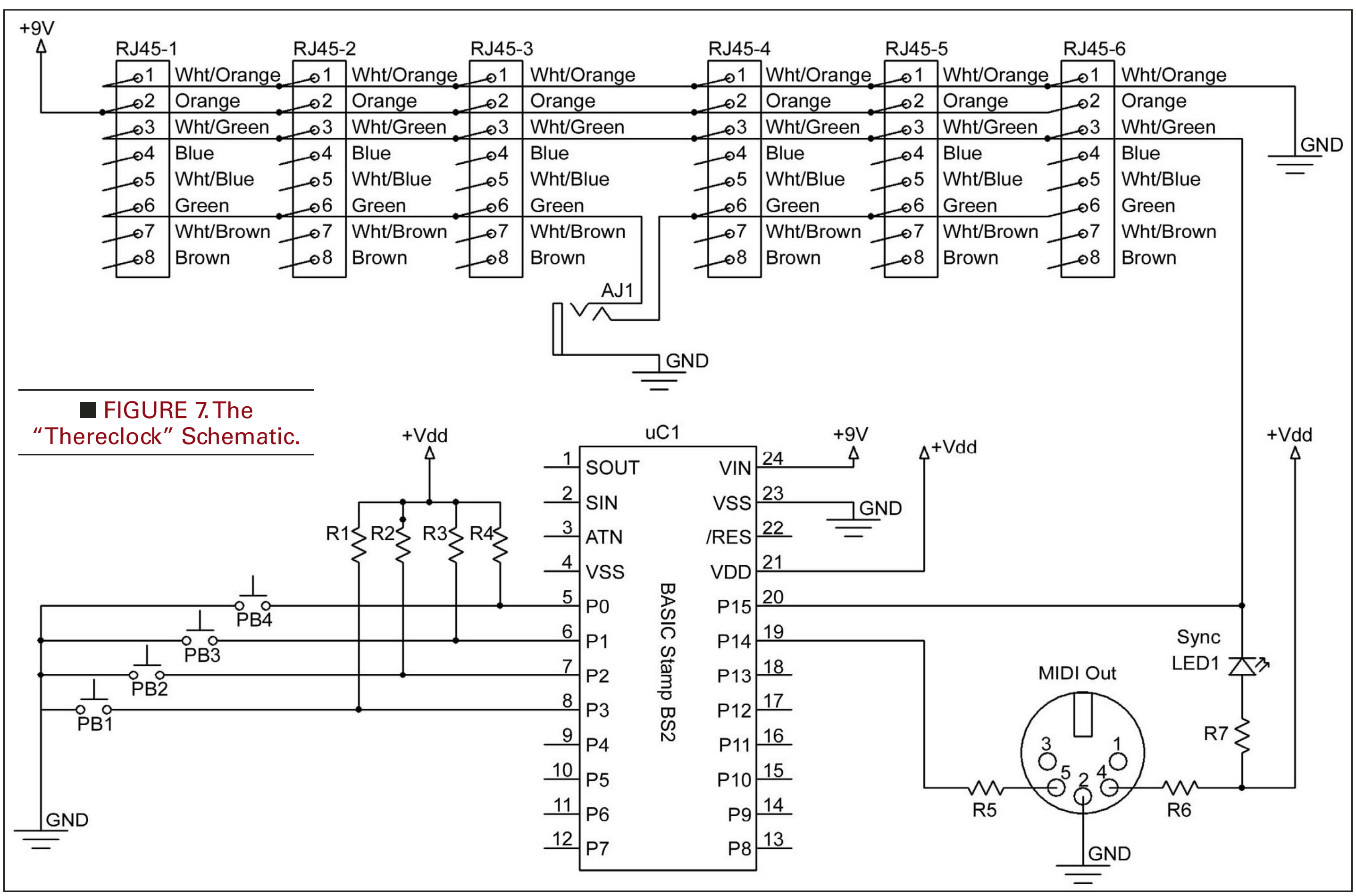


Thereclock Finished and Mounted.

■ FIGURE 8. The Thereping Rehearsal.

The Debut

On the morning of December 31, 2005, we loaded all the equipment onto a 20-foot trailer and hauled it to the pavilion where we set up the P.A. system and the rest of the Thereping equipment. We immediately drew a crowd by playing for a short while (Figure 9), before offering the Thereping units to people in the crowd so they could exper-



■ FIGURE 7. The "Thereclock" Schematic.

■ FIGURE 9. Nic Graner playing a Thereping on the Pavilion.

■ FIGURE 10. Thereping Band sporting their hats while waiting to play in the parade!

iment with the music. We had a crowd of people that watched the jam session for a good, solid three hours!

There were TV cameras from both local and national news organizations present. The most exciting part was the people participating. After just a few moments of instruction on what the parts of the instrument were, folks were able to pick it up and start playing music together instantly! This part of the show was a major success, but we still had to tear down the entire system and reassemble it on the trailer for the parade.

Showtime!

Once everything was hooked back up (and amazingly all the systems survived the relocation), we carefully drove the trailer to the start of the parade route. We powered the entire show using a 2.5 kW gasoline generator on the bed of the pickup truck and put the 400 watt stereo P.A. system on the trailer with the musicians. Each of us wore a fanciful digital hat crafted by Denise Sciloi and others of The Robot Group (Figure 10).

We played music for over an hour and had people clapping and dancing alongside the trailer as we went down Congress Ave. in Austin, TX. First

Acknowledgments

■ I would like to thank the following people who were critical in making the Thereping a reality:

Rick Abbott, Paul Atkinson, Derek Bridges, Don Colbath, Bob Comer, Kym Graner, Nic Graner, Walt Graner, PY Hung, Eric Lundquist, Tom Morin, Gray Mack, Denise Scioli, Mike Scioli, and Sharon Sudduth.

PING)))®™, BASIC Stamp®, and HomeWork Board™ are registered trademarks of Parallax, Inc.

Resources

■ Vern Graner's "Thereping" website
www.thereping.com

■ The Robot Group
www.robotgroup.net

■ First Night Austin
www.FirstNightAustin.org

■ Parallax, Inc.
www.Parallax.com

■ Yamaha DTXpress User Group
www.DTXpressions.com

Night Austin officials estimated that over 100,000 people viewed the parade that night. It was an amazing experience! I think we fulfilled our mission to create a fun and interesting spectacle.

The Future

Though I consider this project a success as it stands, I also feel there is plenty of room for improvement for the equipment. For example, we've barely scratched the surface of the capabilities of the Thereclock to produce MIDI. With a bit more programming, it should be possible to create a number of different types of beats or even songs for playback.

The extra lines in the Ethernet cable could be used for sending song selection information to the Therepings or for allowing the Thereping to send a stereo signal to the Thereclock, thereby enabling the player to select which channel their signal is sent to (i.e., clean/distorted/echo or not, etc.).

The patterns that can be chosen on the Therepings could also be expanded to include swing beats or triplets, for example. The drone setting could be altered to include MIDI output sent back via the CAT-5 cable or to produce chords or blues-style walking bass lines. In short, this instrument has barely been touched in capability! If you do decide to build one, I would love to hear how your project proceeds. All the source code, schematics, and associated material for helping build your own Thereping and Thereclock, along with audio and video clips, are available at **www.nutsvolts.com** or at my website listed in the Resources sidebar. If you have any questions, feel free to contact me at **vern@thereping.com** **NV**

THEREPING INSTRUMENT PARTS LIST

- ❑ LED1 — Red LED
- ❑ LED2 — Amber LED
- ❑ LED3 — Blue LED
- ❑ LED4 — Green LED
- ❑ J1 — RJ45 Keystone Jack
- ❑ C1 — 10µF Capacitor
- ❑ SN1 — Sonar Sensor
- ❑ PB1-3 — N.O. Momentary
- ❑ R1-R3 — 100 Kohm
- ❑ R4-R7 — 330 ohm
- ❑ uC1 — BS2 HomeWork Board

THERECLOCK PARTS LIST

- ❑ LED1 — Red LED
- ❑ J1-6 — RJ45 Keystone Jacks
- ❑ AJ1 — 1/8" Stereo Audio Jack
- ❑ MJ1 — 5 pin DIN MIDI Jack
- ❑ PB1-4 — N.O. Momentary
- ❑ R1-R4 — 100 Kohm
- ❑ R5-R7 — 330 ohm
- ❑ uC1 — BS2 HomeWork Board