Improved MIDI Message for Robotic Angklung Choir

Eko Mursito Budi, Asep Suhada, Hermawan K. Dipojono, Andrianto Handojo, Joko Sarwono
Engineering Physics Department, Institut Teknologi Bandung
Gedung Labtek VI lt 2, Ganesha 10, Bandung, Indonesia
mursito@tf.itb.ac.id

ABSTRACT
Angklung is a unique musical instrument from South East Asia that should be played in a choir, because each angklung can only produce a single note. In a decent angklung composition, at least 3 angklungs should be played together to perform a sound harmony while certain parts of the song might demand up to 10 voices. Moreover, there are two different angklung playing techniques called kurulung (shaking) and centok (snapping). This paper presents the problems of conducting such of act by an angklung robot, then proposes some improvements to MIDI protocol to overcome the challenges.

KEYWORDS
angklung, choir, MIDI, message, robot.

1 INTRODUCTION
Angklung is a musical instrument that is made from bamboo (Figure 1). Each angklung consists of two or more sound tubes mounted on a frame that has a base tube. When the angklung is shaken, the sound tubes hit the base tube and produce a unique rattled sound. Therefore, angklung is categorized as a rattled idiophones instrument [1], which is quite rare in the world.

The modern angklung was innovated in 1938 by Daeng Soetigna from Kuningan, West Java [2]. Before that, the legacy angklung in West Java had been using the Sundanese pentatonic scale (pelog, slendro or madenda). Pak Daeng created angklung that uses the diatonic scale, by defining two types of angklung [3]:

1. Melodic angklung, for playing the main notes of the song (Figure 1).
2. Accompaniment angklung, for supporting the melody by playing the background chords (Figure 2).

Melodic angklung has two different sound tubes. The big tube is tuned to a certain pitch while the small one is tuned to an octave higher (for example, if the big tube is C3, then the small tube must be C4) [4][5]. Many angklungs are needed to cover a certain note range. The standard angklung Pak Daeng comprises of 30 melody angklungs from G2 to C6.
Meanwhile, the accompaniment angklung has many tubes which are tuned according to the chord. For example, the pitch of a C-major angklung are C4, E3 and G3. The standard angklung Pak Daeng normally provides 12 mayor accompaniment angklungs and 12 minor accompaniment angklungs.

An angklung performance requires a group of people to perform a song, just like a choir. Each person holds one or more angklungs of certain note. Then they play together, shaking the angklung in turn, according to the note sequence of the song. As a variation, there are several angklung playing techniques that could be performed by a human player:

1. **Kurulung** is the most common technique where the angklung is continuously shaken as long as indicated by the note's length.
2. **Centok** is played by snapping the base tube once, producing a staccato note.
3. **Tengkep** is doing the kurulung, by holding either one of the sound tubes, thus producing a pure pitch.

Due to its uniqueness, a song must be specially arranged to be performed by an angklung choir. Appendix-A shows an example of an angklung composition for the “Que Sera Sera” song. The composition is written in doremi notation [6]. Without steeping into detail, it can be seen that the composition consists of multiple-voices (V1, V2, and so on). In each voices, the notes are expressed using the number (1 – 7). The note '1' is equal to C3, '2' is D3 and so on because the key of this song is C (written as K: C on the header). Another interesting symbol is the stacatto '^^', written above a note. Examining the arrangements reveals some interesting angklung choir characteristics:

- Several angklungs should be played together as a multi-voices system. The main melody of the song usually become the highest note, then additional lower notes are added to make a harmony.
- The normal notes are played using the *kurulung* technique, while the staccato notes are played as the *centok* technique.

- The *kurulung* and *centok* techniques can be used in any combination (all *kurulung*, all *centok*, *centok* as the main melody supported by *kurulung*, or vice versa).

With proper arrangement, an angklung choir can perform as grand as an orchestra. The biggest difference is that various instruments are involved in an orchestra, creating a rich polyphonic music. On the other hand, angklung choir is mostly homophonic, consisting only of melody angklung, accompaniment angklung, acoustic bass and some percussion. Nevertheless, the rattled sound of bamboo is exquisite in its own.

### 2 PROBLEM FORMULATION

Recently, a robot that can play up to 37 angklungs (from C2 to C6) has been developed (Figure 3). The general architecture of the system is adopted from the MIDI (Musical Instrument Digital Interface) system [5] as shown in Figure 4.
In this setup, a software in the computer runs as the main MIDI system. Meanwhile the robot is a micro-controller equipped with electromechanical arms, to shake the real angklungs. The system works as follows:

1. The Reader load the doremi file, then parsed it into MIDI sequence. It contains various music information, and most importantly a series of timed messages.
2. The sequencer, running in real-time manner, decodes the sequence then timely send the message to the appropriate channels. In this case, each channel is mapped to a certain instrument, such as a piano or a guitar. Usually, the instrument is a synthesizer which produced the artificial sound through the computer's speaker.
3. For angklung robot, the instrument is just a proxy that passes through the message to the actual robot. There are various types of message, but the most important ones are the NOTE_ON and NOTE_OFF. The first commands the robot to start playing a certain angklung, and the second to stop playing.

Digging into the robot (Figure 5), the MIDI message comes from the computer through a USB connection. The micro-controller receives the message and then turn the angklung on or off accordingly. In this particular system, there are 37 angklungs to be played. Each angklung is held by an electromechanical arms (Figure 6). To control the arms, this system uses 5x8 bits shift registers, driven by SPI (Serial Peripheral Interface), because the built-in digital I/O is not enough.

Clearly, the software in the micro-controller must receive the MIDI message then do the command. Each time the NOTE_ON or NOTE_OFF message arrive, the micro-controller needs to send 5 bytes data to the shift registers through the SPI. This operation takes time. Thus, when multiple notes arrive in succession, the micro-controller might be congested.

As happened in the earlier prototypes, the system did not work as expected, especially when playing a song with many voices. Noticeable slackness were heard between the first voice and the last voice, and thus the choir was not perfect. Problem also arose regarding the centok playing technique, which require the angklung to be played in a very short time. To overcome this problem, the improvement should be made in the MIDI message first, before fixing the robot.

3 MIDI MESSAGE

MIDI is a standard to interconnect the computer, electronic instruments, and various related devices [8]. It defines the protocol to send the musical information between devices. The most elemental data is called a MIDI message. A MIDI message consists of an array of bytes. Table 1 describes the NOTE_OFF and the NOTE_ON message that is the key of this discussion.

<table>
<thead>
<tr>
<th>Byte</th>
<th>Nibble</th>
<th>Content</th>
</tr>
</thead>
</table>
| 1    | High   | Command type:  
|      |        | 8H = NOTE_OFF  
|      |        | 9H = NOTE_ON   |
|      | Low    | Channel number (0-15) |
| 2    |        | Note number (0-127)  
|      |        | 60 = C3 (middle C) |
| 3    |        | Velocity (0-127)     |
To play a certain note into an instrument, a NOTE_ON message is sent to the instrument. Then, after waiting as long as the note length, a NOTE_OFF message is sent to stop the sound. The size of both messages are 3 bytes. If several notes must be played simultaneously, then many messages must be sent. Obviously, the total size for NOTE_N and NOTE_OFF messages will be:

\[ S = 2 \cdot M \cdot N \]  

where:
- \( S \): the total size (bytes)
- \( M \): message size.
- \( N \): number of notes

While the time required to send the message will be roughly:

\[ T = 2 \cdot N ( M \cdot T_b + T_d) \]  

where:
- \( T \): total time
- \( T_b \): time to send a byte
- \( T_d \): latency time to send a frame

In the example of “Que Sera Sera” song, the coda part has 6 kurulung voices. Thus, in each transition, 6 NOTE_OFF messages are sent to turn off the previous notes, followed quickly by 6 NOTE_ON messages to turn on the next notes. This situation may raise a significant delay between the first note and the last note, causing the slackness in the choir.

Meanwhile, some parts of the song must be played using the centok together. Using the standard MIDI message, there are two possible ways to handle this technique:

1. By sending the NOTE_ON message and quickly followed by a NOTE_OFF message.
2. By using a different channels for the kurulung and the centok channel. This way, a NOTE_ON message must be sent to start the playing, but the NOTE_OFF message does not have to follow it as quick as the first approach.

The first approach will face timing problem, while the second approach requires one extra channel. This is quite costly, because the standard MIDI provides only 16 channels.

### 4 IMPROVED MIDI MESSAGE

To tackle the multi-voices problem, a simple solution was proposed:

- Multiple notes should be turned on or off simultaneously by a single message.
- Multiple stacatto notes should be turned on by a single message, but do not require a turn off message.

To carry out the solution, the MIDI system common message will be used. This message has been provided by the standard for accommodating custom message. As described in Table 2, this message starts with system exclusive command in the first byte (F0H), then it may contain many bytes, before ended with the termination mark (F7H).

**Table 2. System Common Message**

<table>
<thead>
<tr>
<th>Byte</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>System common message type</td>
</tr>
<tr>
<td></td>
<td>F0H: system exclusive</td>
</tr>
<tr>
<td>2 .. n-1</td>
<td>Message content (free length).</td>
</tr>
<tr>
<td></td>
<td>The value must be between 0 – 7FH.</td>
</tr>
<tr>
<td>n</td>
<td>End of common message (F7H)</td>
</tr>
</tbody>
</table>

**Table 3. New System Common Message**

<table>
<thead>
<tr>
<th>Byte</th>
<th>Nibble</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>System exclusive (F0H)</td>
</tr>
<tr>
<td>2</td>
<td>High</td>
<td>Multi-notes command</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0: MNOTES_OFF</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1: MNOTES_ON</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2: MSTACCATOS_ON</td>
</tr>
<tr>
<td></td>
<td>Low</td>
<td>Channel number (0-15)</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>Velocity (0-127)</td>
</tr>
<tr>
<td>4 .. n-1</td>
<td>Multiple Notes (0-127)</td>
<td></td>
</tr>
<tr>
<td>n</td>
<td></td>
<td>End of common message (F7H)</td>
</tr>
</tbody>
</table>

The custom message is described in Table 3. The second byte contains the command and the channel, while the third byte is the velocity of all notes. Then, the next bytes carry the notes. With
this new message, the total size of message to turn on then turn off multiple notes is :
\[ S_m = 2 (N + 4) \]  \hspace{1cm} (3)

where :
- \( S_m \): the total message size (bytes)
- \( N \): number of notes

And the time to transmit it became :
\[ T_m = 2 \left( (N + 4) \cdot T_b + T_d \right) \]  \hspace{1cm} (4)

where:
- \( T_m \): total time for multinotes message

5 CALCULATION

Comparing the original MIDI protocol against the improved one is quite straight. For the total message size, equation (1) and (3) can be reduced into:
- \[ 2 \cdot 3 \cdot N = 2 (N+4) \]
- \[ 6N - 2N = 8 \]
- \[ N = 2 \]

Which means, that the proposed message is better than the original if there are 3 or more notes to be played together.

Meanwhile, the proposed message has really outperformed the original one in total transmission time as the number of notes increases. Equation (4) suggest that the total time \( (T_m) \) is proportional to byte transmission time \( (T_b) \) and number of notes \( (N) \), while relatively fixed against the latency time \( (T_d) \). On the other hand, equation (2) states that the total time \( (T) \) is proportional to \( N \) against both \( T_b \) and \( T_d \). As analyzed by [9], \( T_d \) is much bigger than \( T_b \), thus the difference would be huge.

6 IMPROVED IMPLEMENTATION

Adapting the new multi-notes messages is straight forward. Listing-1 shows the basic algorithm to process the new MIDI message, along with the standard ones.

It can be seen that the NOTE_ON or NOTE_OFF messages must be mapped to the 5 bytes buffer, then the buffer is sent to the SPI. The process is repeated for each message.

On the contrary, when the MNOTES_ON or MNOTES_OFF messages, multiple notes are mapped to the buffer before the buffer is written to the SPI. Thus, the arms will be turned on or off at once, which is perfect for the angklung choir.

Listing 1. Micro-controller algorithm

```c
/// Improved MIDI processing skeleton
/// Eko M. Budi, 2013

#define FIRST NOTE 48

byte buffer[5];

void setOn(int note) {
    int n = note - FIRST_NOTE;
    int x = n % 8;
    int y = n / 8;
    buffer[y] |= (0x01 << x);
}

void setOff(int note) {
    int n = note - FIRST_NOTE;
    int x = n % 8;
    int y = n / 8;
    buffer[y] &= ~(0x01 << x);
}

void sendBuffer() {
    for (i=0; i<5; i++) {
        spiWrite(buffer[i]);
    }
}

void noteOn(byte frame[]) {
    setOn(getNote(frame));
    sendBuffer();
}

void noteOff(byte frame[]) {
    setOff(getNote(frame));
    sendBuffer();
}

void mNotesOn(frame) {
    for (i=0; i<count(frame); i++) {
        setOn(getMNNote(frame, i));
    }
    sendBuffer();
}
```
// handle the MNOTES_OFF command
void mNotesOff(frame) {
    for (i=0; i<getCount(frame); i++) {
        setOff(getMNote(frame, i));
    }
    sendBuffer();
}

// the main
main() {
    byte frame[128];
    // do the initialisation
    // then loop forever
    for (;;) {
        frame = readFrame();
        command = getCommand(frame);
        switch (command) {
            case NOTE_ON :
                noteOn(frame); break;
            case MNOTES_ON :
                mNotesOn(frame); break;
            case NOTE_OFF :
                noteOff(frame); break;
            case MNOTES_OFF :
                mNotesOff(frame); break;
        }
    }
}

7 CONCLUSION
In this paper we have discussed the improvement to the standard MIDI protocol that was specially designed for playing a choir on an angklung robot. The protocol theoretically reduces the message size as well as the transmission time. Moreover, the software in the micro-controller can avoid repeating a costly operation, while at the same time can turn on and off the multiple angklung simultaneously.

As a note however, the improvement were utilizing the reserved system exclusive message. Thus, for wide acceptance, it still needs an approval from the MIDI consortium.

8 REFERENCES
Appendix A – Example of an angklung composition (doremi notation)

T: Que Sera Sera  
C: Jay Livingston; Ray Evans  
A: Asep Suhada  
M: 3/4  
Q: 125  
K: C

$ INTRO
# 3 voices of kurulung, followed by centok, then kurulung again

V1: \[7 \hat{1} \hat{2} \hat{1} \hat{7} . 0 \hat{2} \hat{3} 4 \hat{0} 7 \hat{1} \hat{.} .0 0\]
V2: \[2 3 \hat{4} 3 \hat{2} . 0 \hat{7} 1 2 \hat{0} \hat{5} \hat{5} \hat{.} .0 0\]
V3: \[7 . . . . .0 0 0 5 \hat{0} \hat{2} \hat{3} . . .0 0\]

$ VERSE-1
# Centok and kurulung background

V1: \[1 \hat{2} 3 \hat{5} 3 \hat{5} 3 \hat{5} \hat{.} \hat{3} \hat{5} \hat{3} \hat{6} 5 \hat{.} 6 \hat{5} 3 4 0 0 0 0 0\]
V2: \[0 . . 1 . \hat{7} . \hat{6} . \hat{5} . \hat{6} . \hat{7} . 1 . \hat{7} . .\]
W: When I was just a little girl, I asked my mother, What will I be

$ VERSE-2
# 3 voices, all centok

V1: \[\hat{7} \hat{1} \hat{2} \hat{1} \hat{7} . 0 \hat{7} \hat{1} \hat{7} . 4 \hat{5} 6 \hat{5} . \hat{7} \hat{1} \hat{.} .\]
V2: \[2 \hat{3} \hat{4} 3 \hat{2} . 2 \hat{4} \hat{5} 2 . 2 \hat{3} \hat{4} 3 . \hat{5} \hat{3} \hat{.} .\]
V3: \[5 \hat{6} \hat{7} 6 \hat{5} . 4 \hat{5} 6 \hat{5} . \hat{5} . \hat{5} . \hat{4} \hat{1} \hat{.} .\]
W: Will I be pretty, Will I be rich, Here's what she said to me

$ CHORUS
# 3 voices, all kurulung

V1: \[1 . \hat{7} 6 . \hat{4} \hat{6} . . . 7 \hat{2} \hat{1} \hat{6} 5 . \hat{1} \hat{3} \hat{.} .\]
V2: \[. . . \hat{4} . \hat{6} 4 . . . 4 \hat{7} \hat{5} 4 3 . \hat{5} \hat{1} \hat{.} .\]
V3: \[\hat{8} . . \hat{4} . 1 . \hat{4} . \hat{6} . \hat{1} . \hat{5} . \hat{.} .\]
W: Que se-ra se-ra What-e-ver will be will be

V1: \[. . \hat{4} 6 5 3 \hat{5} 2 \hat{5} . . . 2 \hat{3} \hat{4} \hat{7} \hat{1} \hat{.} . .\]
V2: \[. . \hat{1} \hat{3} \hat{2} \hat{1} \hat{2} . . 1 . \hat{7} . 7 . \hat{5} \hat{3} \hat{.} . .\]
V3: \[. . \hat{5} . \hat{5} . \hat{5} . \hat{5} . \hat{5} . \hat{2} . \hat{1} \hat{.} . .\]
W: The future's not ours to see, Que sera, sera
$ CODA

# 3 kurulung voices

V1: 7 2 3 4 . 7 1 . . . 7 2 3
V2: 5 4 6 5 . 5 3 . . . 5 4 6
V3: 2 . . 2 . . 1 . . . 2 . .
W:  What will be will be, Que Sera

# soaring to 6 voices

V1: 6 . 7 1 . . . 4 . . . 3 . . .
V2: 2 . 2 3 . . . 1 . . . 1 . . .
V3: 7 . 5 5 . . . 4 . . . 3 . . .
V4: 0 . 2 1 . . . 6 . . . 5 . . .
V5: 0 0 0 0 0 0 0 0 0 1 . . . 1 . . .
V6: 0 0 0 0 0 0 0 0 0 0 4 . . . 1 . . .
W:  Sera ...