# PROJECT #3 VERIFICATION

MUSIC BOX

COOPER BROTHERTON

INSTRUCTOR: DR. MICHAEL JO

ECE230-03

# REQUIREMENTS

- The master clock and timers shall be sourced by the 48MHz HFXT oscillator.
- All timing requirements in the program shall be accomplished with timers.
- On start, the program shall begin with the first note of the song and play the entire song in an infinite loop.
- The period for all six notes shall be accurate to within ±1µs.
- During *rest* notes, no tone should play.
- The duration of each note shall be 500ms (accurate to within ±1ms).
- The circuit implementation shall match Figure 1, and the output is clear and audible.

# ADVANCED REQUIREMENTS

- The master clock and timers shall be sourced by the 48MHz HFXT oscillator.
- All timing requirements in the program shall be accomplished with timers.
- On start, the song shall be *paused*.
- While the song is *paused*, upon the press and release of S2, the song shall begin playing at the current position in the song (the first note of the song if this is the first press of S2) and play the entire song in an infinite loop.
    - Switch S2 shall be software debounce on press and release.
    - The song shall begin playing within 10ms of the release of S2.
- While the song is *playing*, upon the press of S2, the song shall *pause* at the current position and the position is retained until the song is un-paused.
- The period for *all* notes shall be accurate to within ±0.25µs.
- During *rest* notes, no tone should play.
- The duration of each note shall be accurate to within ±1ms.
- The circuit implementation shall match Figure 1, and the output is clear and audible.
- The song must include at least three different note lengths and at least 6 different notes.

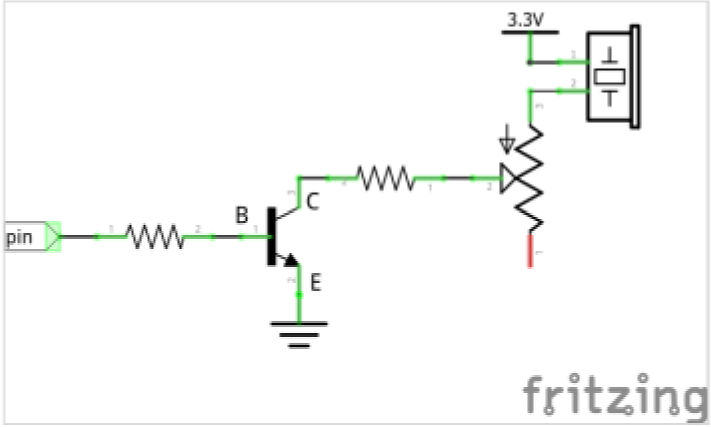


Figure 1 – Circuit Diagram

# TEST PLAN

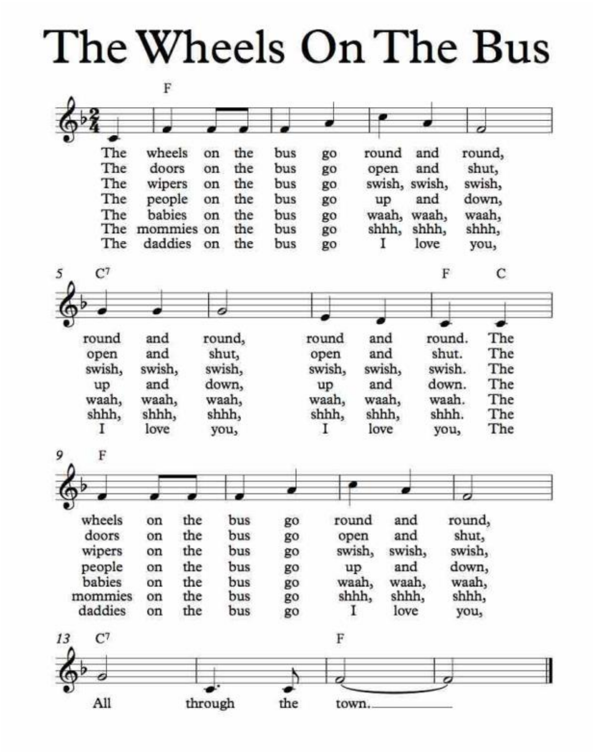
The following details a plan for testing the specifications.

| Test | Procedure | Pass/Fail Criteria |
|---|---|---|
| 1 | Verify the song has at least 3 different note lengths and 6 different notes. | The song contains at least 3 different note lengths and 6 different notes. |
| 2 | Verify the master clock and timers are sourced by the 48MHz HFXT oscillator | MCLK and timers use HFXT to when initialized |
| 3 | Verify all timing requirements are accomplished with timers | No for loops are used for timing, only timers |
| 4 | On start, verify the song is paused | The song does not play on program launch |
| 5 | Verify the song plays when S2 is pressed and released while paused within spec | The song plays within 0<t<10ms after S2 is released. |
| 6 | Verify S2 is debounced on both press and release | S2 has software debouncing on every press and release |
| 7 | Verify the song pauses when S2 is pressed while playing within spec | The song pauses within 0<t<1ms after S2 is pressed. |
| 8 | Verify the song's position is maintained between playing and pausing. | On the first press and release of S2, the song plays from the beginning, any subsequent press and releases while paused resume play from where it was paused |
| 9 | Verify the system waits until S2 has been released before waiting for a new S2 press | The system waits until S2 is released until detecting a new S2 press and release. |
| 10 | Use math to verify the period of all notes are within spec | Each note's defined half period is within ±0.25µs of the note's actual period |
| 11 | Use an oscilloscope to verify no tones play during rest notes | The output to P2.6 is constant during rest notes |
| 12 | Use an oscilloscope to verify the duration of each note is within spec | Notes are played in staccato for their appropriate duration within ±1ms |
| 13 | Verify the circuit matches Figure 1 and the output is clear and audible | Ensure the circuit is built like the figure and the speaker is audible and not distorted while playing |

# VERIFICATION

## TEST 1

The following is the sheet music for the song “The Wheels on the Bus” used for the project.


The Wheels On The Bus
F
The wheels on the bus go round and round,
The doors on the bus go open and shut,
The wipers on the bus go swish, swish, swish,
The people on the bus go up and down,
The babies on the bus go waah, waah, waah,
The mommies on the bus go shhh, shhh, shhh,
The daddies on the bus go I love you,
5 C7
F C
round and round, round and round. The
open and shut, open and shut. The
swish, swish, swish, swish, swish, swish. The
up and down, up and down. The
waah, waah, waah, waah, waah, waah. The
shhh, shhh, shhh, shhh, shhh, shhh. The
I love you, I love you, The
9 F
wheels on the bus go round and round,
doors on the bus go open and shut,
wipers on the bus go swish, swish, swish,
people on the bus go up and down,
babies on the bus go waah, waah, waah,
mommies on the bus go shhh, shhh, shhh,
daddies on the bus go I love you,
13 C7
F
All through the town.

There are 5 different note lengths: eighth, quarter, quarter dotted, half, and whole notes.

There are 7 different notes: C4, D4, E4, F4, G4, A4, and C5.

Both are greater than the minimum required number.

**Meets criteria.**

## TEST 2

Within `Timers.c`, the function `CS_setExternalClockSourceFrequency` is called with a value of `48000000` which sets the HFXT to 48MHz. When using the function `MAP_CS_initClockSignal`, both MCLK and SMCLK use HFXT with different dividers. All initializations of Timer A use SMCLK as a source.

HFXT is set to 48MHz and the master clock and timers are sourced by it.

**Meets criteria.**

## TEST 3

The code does not use a `for` loop nor a `while` loop for timing. The only time `while` loops are used is for infinite looping or polling switch input. Timers are used in every instance of timing.

**Meets criteria.**

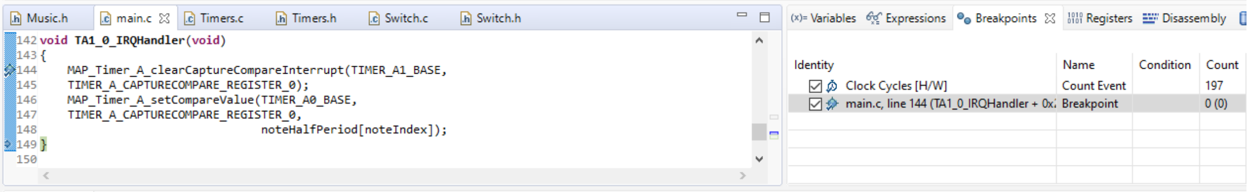
## TEST 4

When the program starts, the song does not play.

**Meets criteria.**

## TEST 5

Once S2 is released, the program goes through a debounce function which has a 5ms delay. The delay of 15000 ticks stored in TA2 CCR0, which corresponds to 5ms as seen below. With the addition of the few instructions taking 197 cycles from the clock cycle counter it totals to less than 10ms.



```
void TA1_0_IRQHandler(void)
{
    MAP_Timer_A_clearCaptureCompareInterrupt(TIMER_A1_BASE,
    TIMER_A_CAPTURECOMPARE_REGISTER_0);
    MAP_Timer_A_setCompareValue(TIMER_A0_BASE,
    TIMER_A_CAPTURECOMPARE_REGISTER_0,
                                noteHalfPeriod[noteIndex]);
}
```

| Identity | Name | Condition | Count |
|---|---|---|---|
| ☑ Clock Cycles [H/W] | Count Event | | 197 |
| ☑ main.c, line 144 (TA1_0_IRQHandler + 0x: | Breakpoint | | 0 (0) |

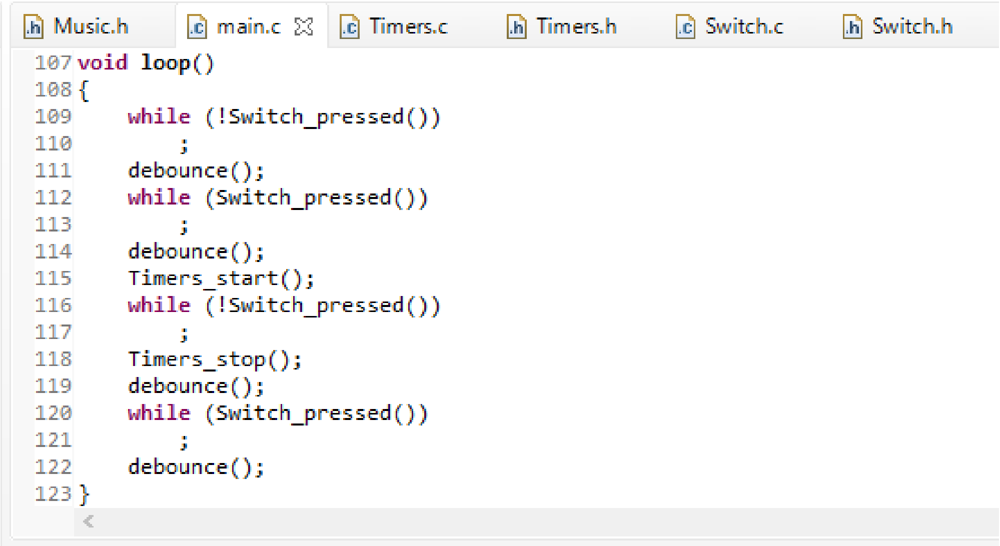
The following calculation using TA2 which sources SMCLK at 3MHz and the additional cycles for MCLK are at 12MHz:

$$\frac{15000}{3 \times 10^6} + \frac{197}{12 \times 10^6} = 5.016ms$$

**Meets criteria.**

## TEST 6

The following code depicts the loop where S2 presses are polled in the program.



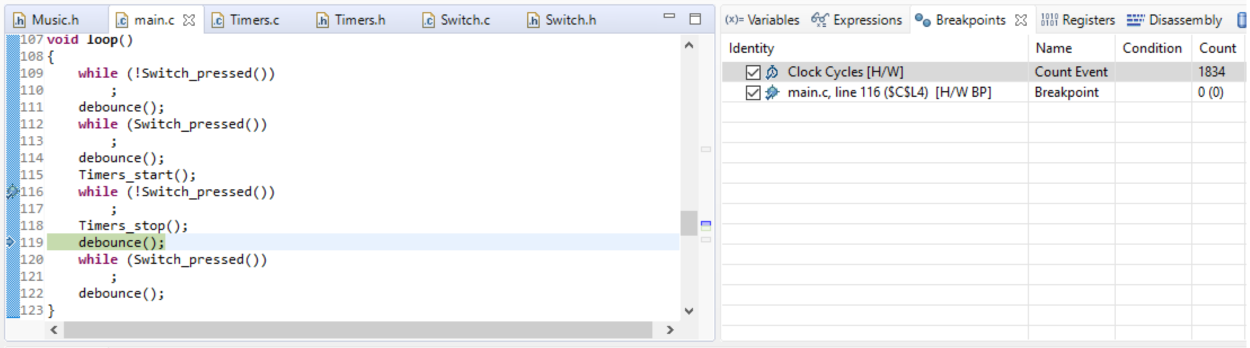
```
void loop()
{
    while (!Switch_pressed())
        ;
    debounce();
    while (Switch_pressed())
        ;
    debounce();
    Timers_start();
    while (!Switch_pressed())
        ;
    Timers_stop();
    debounce();
    while (Switch_pressed())
        ;
    debounce();
}
```

After each press or release and before the next poll, the program always goes to the debounce function.

**Meets criteria.**

## TEST 7

The following code analysis depicts the code between S2 being pressed and the song being paused. By looking at the clock cycle counter, it can be determined how long it took.



```
107 void loop()
108 {
109     while (!Switch_pressed())
110         ;
111     debounce();
112     while (Switch_pressed())
113         ;
114     debounce();
115     Timers_start();
116     while (!Switch_pressed())
117         ;
118     Timers_stop();
119     debounce();
120     while (Switch_pressed())
121         ;
122     debounce();
123 }
```

| Identity | Name | Condition | Count |
|---|---|---|---|
| ☑ Clock Cycles [H/W] | Count Event | | 1834 |
| ☑ main.c, line 116 ($C$L4) [H/W BP] | Breakpoint | | 0 (0) |

The following calculation shows how long the process took given that the MCLK clock rate is 12MHz:

$$\frac{1834}{12 \times 10^6} = 0.153ms$$

**Meets criteria.**

## TEST 8

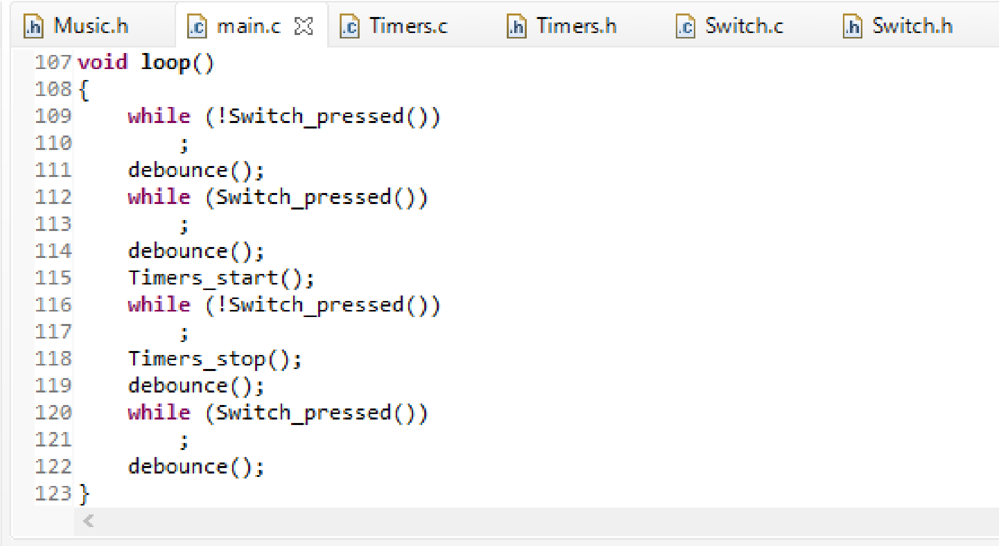
On the first press and release of S2, the song starts from the beginning. After the first press and release of S2, every time S2 is pressed and released while the music is paused, the song continues from the same note as when it was paused.

This is because a global variable `noteIndex` is only updated when the TA0 and TA1 are running and when the music is paused, so are the timers.

**Meets criteria.**

## TEST 9

The following code analysis shows the loop where S2 presses control the state of the program.



```
void loop()
{
    while (!Switch_pressed())
        ;
    debounce();
    while (Switch_pressed())
        ;
    debounce();
    Timers_start();
    while (!Switch_pressed())
        ;
    Timers_stop();
    debounce();
    while (Switch_pressed())
        ;
    debounce();
}
```

After the music has been stopped (`Timers_stop()`) and debouncing, the program is held with a `while` loop as long as S2 is being pressed. Additionally, running the program verifies this interaction.

**Meets criteria.**

## TEST 10

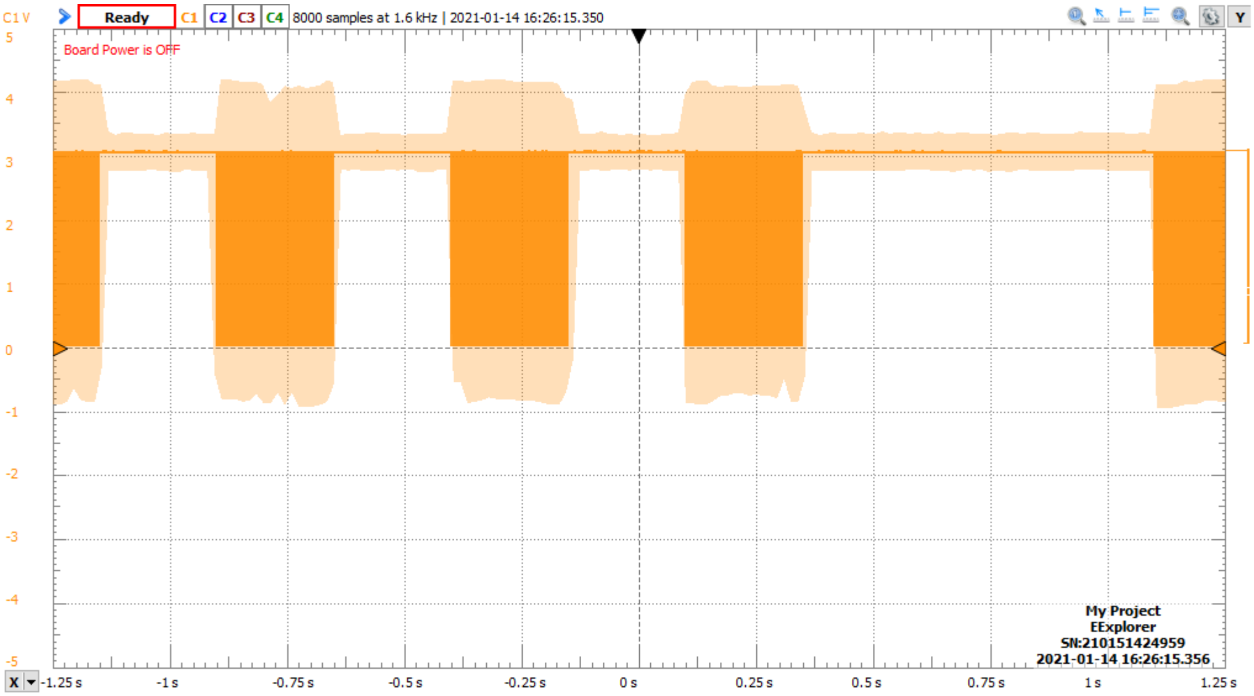
The following math will compare each note's half-period in `Music.h` with the note's actual period based on its frequency and ensure the period is within spec. The value from `Music.h` is loaded into TA0CCR0, which uses SMCLK, which has a frequency of 1.5MHz. ±0.25µs

| Note | Frequency (Hz) | Period (ms) | `Music.h` half period | `Music.h` period (ms) | ±0.25µs? |
|---|---|---|---|---|---|
| C4 | 261.63 | 3.82222 | 5733 | 3.82200 | ✓ |
| D4 | 293.66 | 3.40530 | 5108 | 3.40533 | ✓ |
| E4 | 329.63 | 3.03375 | 4551 | 3.03400 | ✓ |
| F4 | 349.23 | 2.86344 | 4295 | 2.86333 | ✓ |
| G4 | 392.00 | 2.55102 | 3827 | 2.55133 | ✓ |
| A4 | 440.00 | 2.27273 | 3409 | 2.27267 | ✓ |
| C5 | 523.25 | 1.91113 | 2867 | 1.91133 | ✓ |

**Meets criteria.**

## TEST 11

Because "The Wheels on the Bus" does not have any rest notes, the following oscilloscope capture depicts P2.6 during a part of "Twinkle Twinkle Little Star" where a rest is present.



8000 samples at 1.6 kHz | 2021-01-14 16:26:15.350

Board Power is OFF

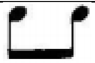
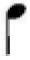
My Project
EExplorer
SN:210151424959
2021-01-14 16:26:15.356

The output of P2.6 is a constant 3.06221V during a rest note.

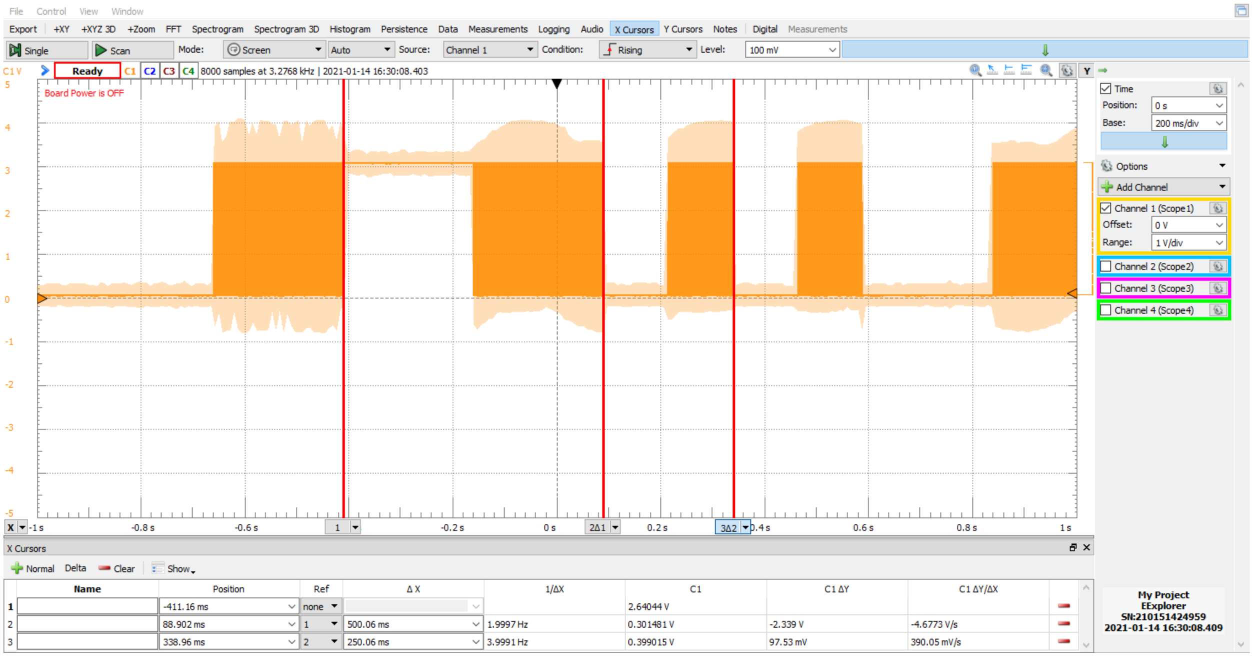
**Meets criteria.**

## TEST 12

The following oscilloscope shots are for showing that the note duration is correct. Notes are played staccato where the note plays for half the duration. The song is played at 120 BPM (500ms per beat). The expected and measured values are tabulated below.

| Note type | Symbol | Expected duration (ms) | Measured duration (ms) | ±1 ms? |
|---|---|---|---|---|
| Eighth |  | 250 | 250.06 | ✓ |
| Quarter |  | 500 | 500.6 | ✓ |
| Quarter dot | | 750 | 449.95 | ✓ |
| Half | | 1000 | 1000.2 | ✓ |
| Whole | | 2000 | 1999.9 | ✓ |

The oscilloscope captures are the following:



Mode: Screen Auto Source: Channel 1 Condition: Rising Level: 100 mV

8000 samples at 3.2768 kHz | 2021-01-14 16:30:08.403

Board Power is OFF

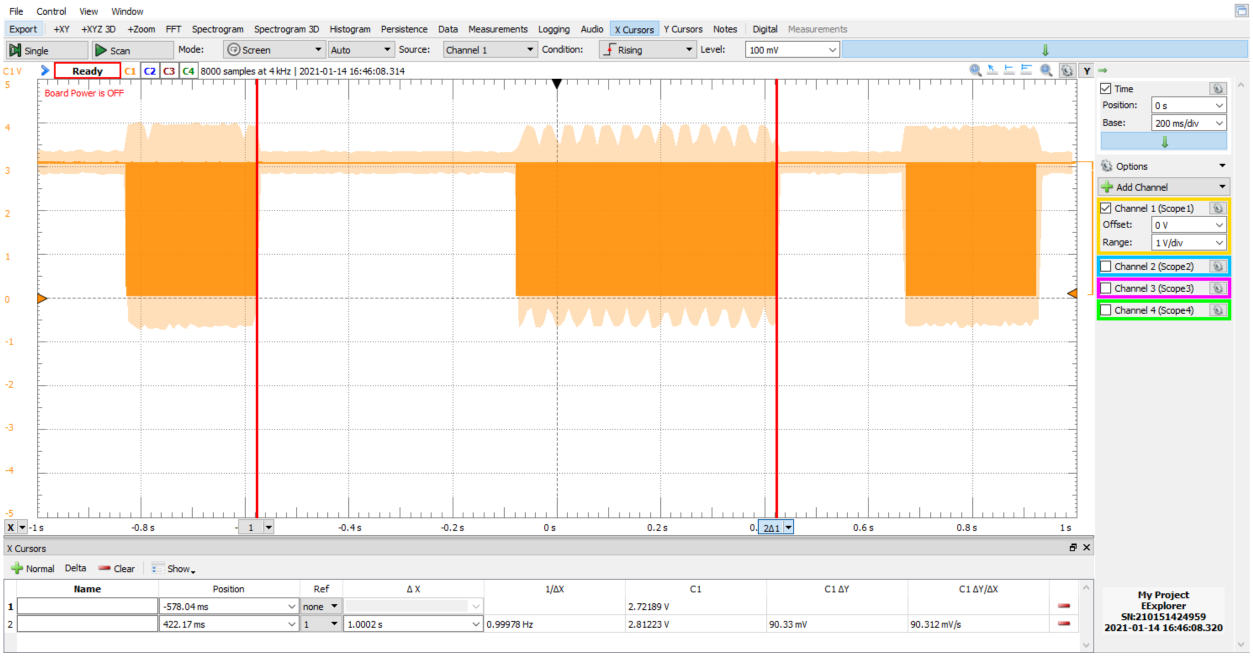
Time Position: 0 s Base: 200 ms/div

Channel 1 (Scope1) Offset: 0 V Range: 1 V/div

| | Name | Position | Ref | Δ X | 1/ΔX | C1 | C1 ΔY | C1 ΔY/ΔX |
|---|---|---|---|---|---|---|---|---|
| 1 | | -411.16 ms | none | | | 2.64044 V | | |
| 2 | | 88.902 ms | 1 | 500.06 ms | 1.9997 Hz | 0.301481 V | -2.339 V | -4.6773 V/s |
| 3 | | 338.96 ms | 2 | 250.06 ms | 3.9991 Hz | 0.399015 V | 97.53 mV | 390.05 mV/s |

My Project
EExplorer
SN:210151424959
2021-01-14 16:30:08.409

Quarter note and Eighth notes



Mode: Screen Auto Source: Channel 1 Condition: Rising Level: 100 mV

8000 samples at 4 kHz | 2021-01-14 16:46:08.314

Board Power is OFF

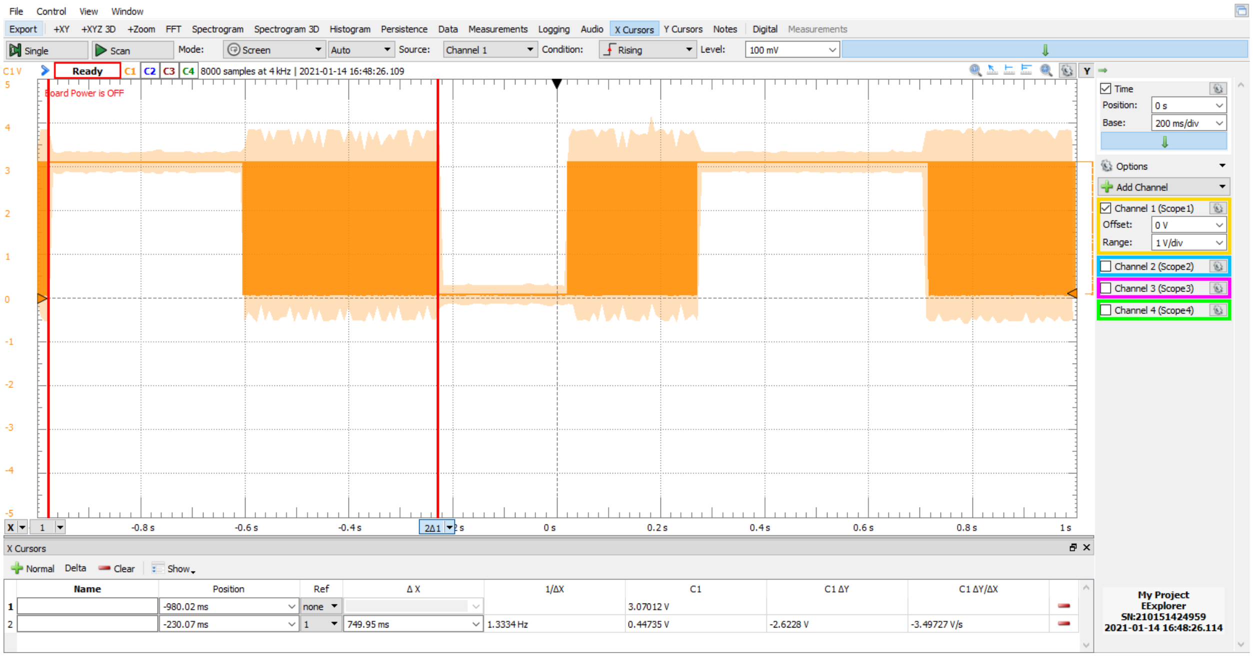
Time Position: 0 s Base: 200 ms/div

Channel 1 (Scope1) Offset: 0 V Range: 1 V/div

| | Name | Position | Ref | Δ X | 1/ΔX | C1 | C1 ΔY | C1 ΔY/ΔX |
|---|---|---|---|---|---|---|---|---|
| 1 | | -578.04 ms | none | | | 2.72189 V | | |
| 2 | | 422.17 ms | 1 | 1.0002 s | 0.99978 Hz | 2.81223 V | 90.33 mV | 90.312 mV/s |

My Project
EExplorer
SN:210151424959
2021-01-14 16:46:08.320

Half Note



Mode: Screen Auto Source: Channel 1 Condition: Rising Level: 100 mV

8000 samples at 4 kHz | 2021-01-14 16:48:26.109

☑ Time
Position: 0 s
Base: 200 ms/div

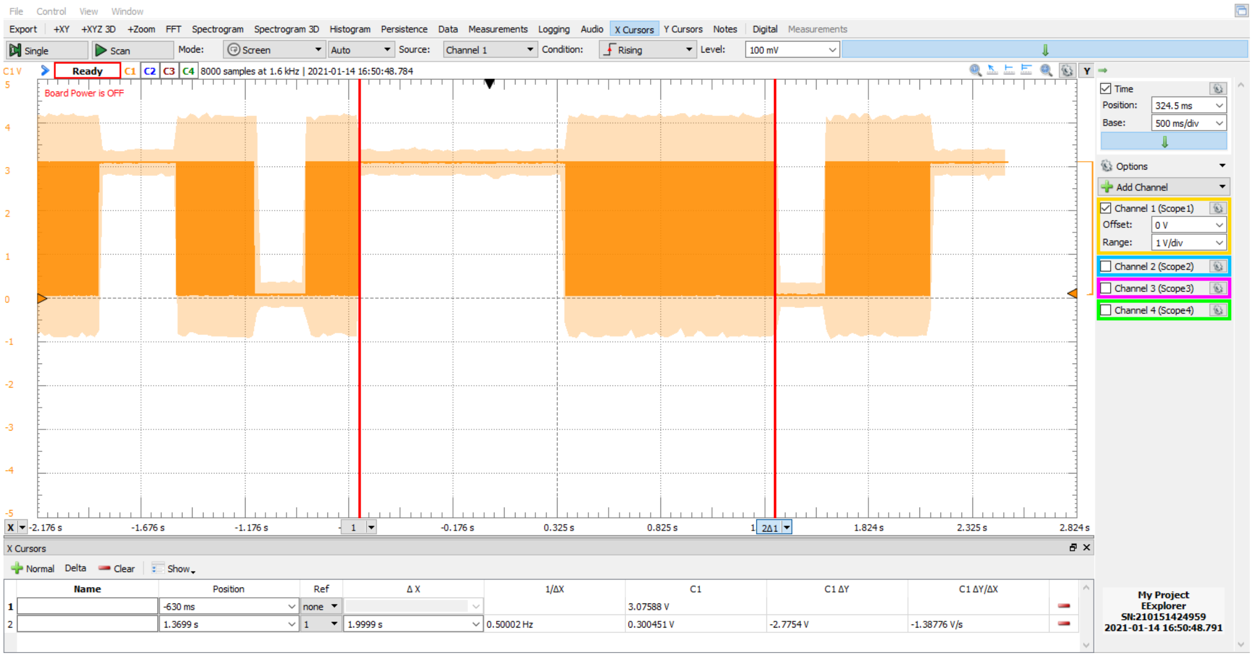
☑ Channel 1 (Scope1)
Offset: 0 V
Range: 1 V/div

X Cursors

|  | Name | Position | Ref | Δ X | 1/ΔX | C1 | C1 ΔY | C1 ΔY/ΔX |
|---|---|---|---|---|---|---|---|---|
| 1 |  | -980.02 ms | none |  |  | 3.07012 V |  |  |
| 2 |  | -230.07 ms | 1 | 749.95 ms | 1.3334 Hz | 0.44735 V | -2.6228 V | -3.49727 V/s |

My Project
EExplorer
SN:210151424959
2021-01-14 16:48:26.114

Quarter Dot Note



Mode: Screen Auto Source: Channel 1 Condition: Rising Level: 100 mV

8000 samples at 1.6 kHz | 2021-01-14 16:50:48.784

☑ Time
Position: 324.5 ms
Base: 500 ms/div

☑ Channel 1 (Scope1)
Offset: 0 V
Range: 1 V/div

X Cursors

|  | Name | Position | Ref | Δ X | 1/ΔX | C1 | C1 ΔY | C1 ΔY/ΔX |
|---|---|---|---|---|---|---|---|---|
| 1 |  | -630 ms | none |  |  | 3.07588 V |  |  |
| 2 |  | 1.3699 s | 1 | 1.9999 s | 0.50002 Hz | 0.300451 V | -2.7754 V | -1.38776 V/s |

My Project
EExplorer
SN:210151424959
2021-01-14 16:50:48.791

Whole Note

**Meets criteria.**

## TEST 13

The circuit is built like the given figure. The speaker’s output is audible and not distorted.

**Meets criteria.**

# CONCLUSION

All tests met the requirements of both the basic and advanced requirements.