Artificial Sine Wave Generation Using SX Communications Controller

Application Note11
Chris Fogelklou
November 2000

1.0 Introduciton

Sine waves are used extensively in the telecommunications industry, and are traditionally difficult to implement in software without using code-consuming table lookups or complex math routines. One easy solution is to create an artificial sine wave, which utilizes the properties of gravity and creates a near-perfect sine wave. This signal is sufficient for applications such as DTMF (Dual-Tone Multi-Frequency) generation, FSK generation,PSK generation and many other applications that require frequency generation.

In the past, telephony functions such as FSK (frequency shift keying) generation and detection, DTMF (dual tone multi frequency) dialing generation and detection, and Caller ID could not be implemented with an 8-bit embedded MCU because performance levels were not high enough to support them. As a result, either a custom MCU had to be designed or a 16 or 32 bit device is used. Now the Ubicom SX communications controller,with performance reaching 100 MIPS (million instructions per second) and deterministic interrupt architecture overcome this roadblock by providing the ability to perform these functions in software.

Unlike other MCUs that add functions in the form of additional silicon, the SX Series uses its industry-leading performance to execute functions as software modules, or Virtual Peripheral. These are loaded into a high-speed on-chip flash/EEPROM program memory and executed as required. In addition, a set of on-chip hardware Peripherals is available to perform operations that cannot readily be done in software, such as comparators, timers and oscillators.

2.0 Description of Sine Wave Virtual Peripheral

2.1 Principle Used

When a ball is thrown into the air, it has a constant downward acceleration until it has a velocity of zero. At this point it obtains a positive velocity towards the ground until it hits the ground. What were to happen if the ball were to continue through the ground, once again accelerating towards the ground? It would decelerate until its velocity reached zero and once again would gain velocity towards the ground. Passing the ground, it would begin decelerating and the cycle would continue…

This type of algorithm can be implemented in an interrupt service routine. The first block of the Interrupt service routine services the PWM, which serves as a D/A converter, outputting the current value of the sin wave to the external circuitry.

2.2 Program Description

This program demonstrates the generation of an artificial SINE wave using the properties of gravity. The sine wave starts at a defined point in time with a defined velocity. The main loops indefinitely, after initializing some registers. A PWM output outputs the current value of the sine wave. Because of properties inherent in the design of the Ubicom PWM, the resolution of the output SINE wave varies inversely with the frequency.

2.3 Interrupt Service Routine

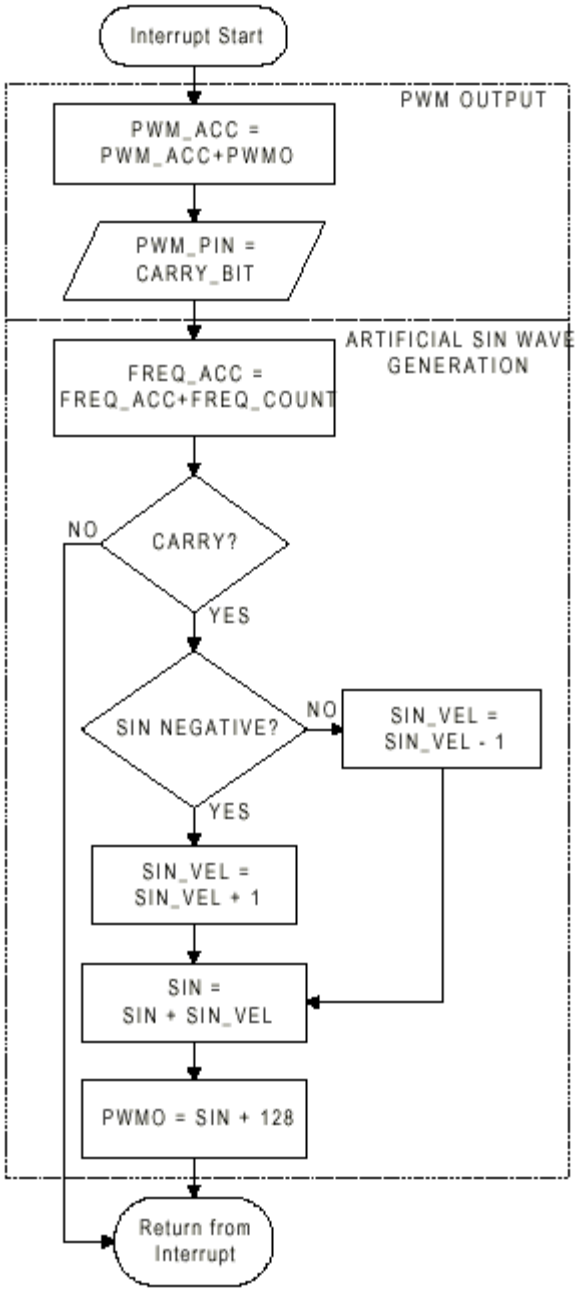

Figure 2-1. Interrupt Service Routine Flowchart

3.0 Different Sections of Sine Wave Virtual Peripheral

This documentation provides a brief overview of different sections involved in "Artificial Sine Wave Generation Using SX Communications Controller". It also makes use of PWM Virtual Peripheral module.

The below five sections of the sine wave Virtual Peripheral module can be inserted in a main source code at appropriate locations to meet the requirement of the sine wave generation.

It consists of five sections:

(a) Equates Section

(b) Bank Section

(c) Initialization Section

(d) Interrupt Section

(e) Watch Section

3.1 Equates Section

This section gives the equates of the sine wave Virtual Peripheral module and it also defines the output pin for a sine wave Virtual Peripheral.

This section also gives different types of frequency generated by the sine wave Virtual Peripheral by calling the value defined in the initialization section.

```
; _begin SINEWAVE
f697_h       equ          $012         ; 697Hz specified for DTMF Frequency
f697_l       equ          $09d

f770_h       equ          $014         ; 770Hz specified for DTMF Frequency
f770_l       equ          $090

f852_h       equ          $016         ; 852Hz specified for DTMF Frequency
f852_l       equ          $0c0

f941_h       equ          $019         ; 941Hz specified for DTMF Frequency
f941_l       equ          $021

f1209_h      equ          $020         ; 1209Hz specified for DTMF Frequency
f1209_l      equ          $049

f1336_h      equ          $023         ; 1336Hz specified for DTMF Frequency
f1336_l      equ          $0ad
f1447_h      equ          $027         ; 1447Hz specified for DTMF Frequency
f1447_l      equ          $071

f1633_h      equ          $02b         ; 1633Hz specified for DTMF Frequency
f1633_l      equ          $09c
f1300_h      equ          $022         ; 1300Hz Signifies HIGH data in Bell202 Spec
f1300_l       equ         $0b7

f2100_h      equ          $038         ; 2100Hz Signifies LOW data in Bell201 Spec
f2100_l      equ          $015

;*********************************************************************************************
; Pin Definition
;*********************************************************************************************
swPwmPin     equ          ra.0         ; sine wave PWM output pin of port RA
; _end
```

3.2 Bank Section

This section describes the use of the banks in the sine wave Virtual Peripheral. The bank used in the sine wave Virtual Peripheral module (BANK 1) should be same in the main source template, if used with other Virtual Peripheral modules.

```
; _begin SINEWAVE
 swSinGenBank       =      $      ; specified in the BANK 1.
 swFreqAccLow       ds     1      ; 16-bit accumulator which decides to increment the sine  wave
 swFreqAccHigh      ds     1      ;
 swFreqCountLow     ds     1      ; 16-bit counter which decides which frequency for the sine wave
 swFreqCountHigh    ds     1      ; Freq_count = Frequency * 6.83671552
 swSin              ds     1      ; The current value of the imitation sin wave
 swSinvel           ds     1      ; The velocity of the sin wave
 swPwm0Acc          ds     1      ; PWM accumulator
 swPwm0             ds     1      ; current PWM output
; _end
```

3.3 Initialisation Section

It provides the initialization part of the sine wave Virtual Peripheral module with the different constants and the values. In this section a cosine wave or a sine wave with a 90 degrees out of phase can also be generated.

```
; _begin SINEWAVE
             _bank  swSinGenBank
             mov    swSin,#32                ; init variables. A sine wave starts at 1,
                                             ; A cos wave starts at 0.
             mov    swSinvel,#0

;*************************************************************************************************
; Comment the above two instructions and uncomment the below two instructions
; to have a  sine wave
;*************************************************************************************************
;            mov    swSin,#-4                ; use these values for a wave which is
                                             ; 90 degrees out of phase
;            mov    swSinvel,#-8

             mov    swFreqCountHigh,#f1300_h
             mov    swFreqCountLow,#f1300_l  ; set up variables for 1300 Hz.
             mov    !option,#%00011111       ; The specified value will enable the
                                             ; wreg register and rtcc interrupt

 main_loop
             jmp       main_loop             ; do nothing (Interrupts will handle -- the rest).
; _end
```

3.4 Interrupt Section

It provides with the interrupt service routine of the artificial sine wave generation using SX communications controller that is to be handled when a interrupt comes. The flow of the interrupt service routine can be known by the flowchart given above in Figure 2-1.

The interrupt service routine of the sine wave Virtual Peripheral module with a "retiw" value of -163 at an oscillator frequency of 50MHz, this code runs every 3.26us.

```
; _begin SINEWAVE
;*****************************************************************************************
PWM_OUTPUT
; This outputs the current value of pwm0 to the PWM_pin. This generates an analog voltage at
; PWM _Pin after filtering
;*****************************************************************************************
             add    swPwm0Acc,swPwm0    ; add the PWM output to the accumulator
             snc
             jmp    :carry              ; if there was no carry, then clear the
; PWM-pin
             clrb   swPwmPin
             jmp    swPwmOut
:carry
             setb   swPwmPin            ; Otherwise set the swPWMPin
swPWMOut


;*****************************************************************************************
sine_generator
; This routine generates a synthetic sine wave with values ranging from -32 to 32. Frequency is
; specified by the counter.
;*****************************************************************************************
             _bank  swSinGenBank
             add    swFreqAccLow,swFreqCountLow      ; advance sine at frequency
             jnc    : no_carry                       ; if lower byte rolls over
             inc    swFreqAccHigh                    ; carry over to upper byte
             jnz    :no_carry                        ; if carry causes rollover
             mov    swFreqAccHigh,swFreqCountHigh    ; then add freq counter to accumulator
             jmp    :change_sin
:no_carry
             add    swFreqAccHigh,swFreqCountHigh    ; add the upper bytes of the accumulators
             jnc    : no_change
:change_sin
             mov    w,++swSinvel                     ; if the velocity of sine
             sb     swSin.7                          ; is positive, accelerate
             mov    w,- -swSinvel                    ; it. Otherwise decelerate it.
             mov    swSinvel,w
             add    swSin,w                          ; add the velocity to sin
             mov    swPwm0,swSin                     ; mov the value of SIN into the
                                                     ; PWM output
             add    swPwm0,#128                      ; add 128 to put it in the center of
                                                     ; the PWM output
: no_change
sin_generator_out
;*****************************************************************************************
:ISR_DONE
; This is the end of the interrupt service routine. Now load 163 into w and perform a retiw to
; interrupt 163 cycles from the start of this one. (3.26us@50MHz)
;*****************************************************************************************
             mov    w,#-163                          ; interrupt 163 cycles after this interrupt
             retiw                                   ; return from the interrupt
; _end
```

3.5 Watch Section

It consists of the watch variables to be observed when the sine wave Virtual Peripheral module is made to run and status of the variables can be known. This feature can be used only with "SX_Key".

```
; _begin SINEWAVE
      watch   swFreqAccLow, 16,uhex          ; 16-bit accumulator to the sine wave
      watch   swFreqCountLow,16,uhex         ; 16-bit count which decides frequency of sine wave
      watch   swSin,8,sdec                   ; current value of imitation sin wave
      watch   swSinvel,8,sdec                ; velocity of sine wave
      watch   swPwm0,8,udec                  ; current PWM output
      watch   swPwm0Acc,8,udec               ; PWM accumulator
; _end
```

4.0 Features

4.1 Creating The Wave

During the positive half of the wave cycle, the program just increments the velocity (accelerates) until the threshold point is reached and then the velocity (decelerates) is decrement until the negative threshold point is reached. The velocity again accelerates until the positive threshold point is reached and the process continues. This new velocity is added to the current value of the sine wave. The final task is to load the new value of the sine wave into the PWM register, and to add #128 to the PWM output to center the wave at 2.5V DC.

4.2 Timing

The initial step of the artificial sine wave generator is to determine if it is time to update the value of the sine wave. The 16-bit FREQ_COUNT register determines the rate at which the wave is updated. Each cycle of the wave is made up of 32 separate points, meaning that the 16-bit FREQ_ACC register must roll over 32 times to cycle through an entire period of the sine wave. If we combine these factors with the interrupt rate of 3.26us, we can calculate the value to load to the FREQ_COUNT register for any given frequency.

With a FREQ_COUNT value of 1, it will take 65536 interrupts for the 16-bit FREQ_ACC register to roll over.

One Period = 32 separate points.

Therefore, there will be 32 rollovers x 65536 interrupts for one period.

One Period = 2 097 152 interrupts

Since the ISR rate = 3.26us.

One period (s) is 2 097 152 x 3.26us = 6.836715520 sec

Frequency = 0.14627Hz.

Resolution = 0.14627 Hz

Maximum output frequency = 9.6kHz.

Output frequency = FREQ_COUNT x 0.14627Hz

FREQ_COUNT = (desired frequency) x 6.83671552

The 16-bit value of FREQ_COUNT must be loaded into two separate 8-bit registers,

FREQ _LOW and FREQ_COUNT_HIGH.

4.3 Circuit Design Procedure

The simplest version of the circuit requires only two components for the PWM output, a resistor and a capacitor.

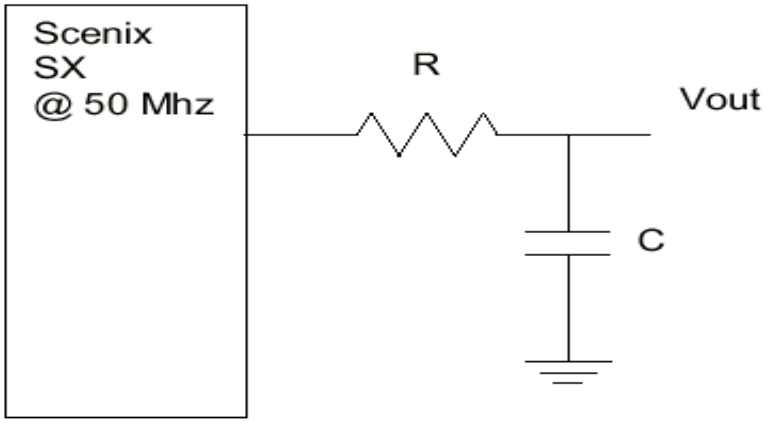

Figure 4-1. Circuit Diagram

Depending on the maximum frequency you wish to obtain, you should adjust the component values for R and C to choose the resolution of the PWM. Ideally, you should calculate the maximum sine frequency output you will use and choose the cutoff to be at this frequency.

For instance, if your maximum output frequency will be 2.1Khz,

calculate R and C:

First, choose a value for R.

R=1000 ohms

Now, calculate C:

C = 1/(2 * pi * Cutoff Frequency * R)

Therefore:

C = 1/(2 * 3.14 * 2100Hz * 1000 ohms)

And

C = 0.076uF

By having different combinations of R & C components we can control the charging and discharging time of the capacitor, which will have an effect on the sine wave generated.

5.0 Applications

The sine wave signals plays a very important role in many applications such as in DTMF (Dual-Tone Multi-Frequency) generation, FSK generation, PSK generation, and many other applications that require frequency generation.

Use the five different sections of the sine wave Virtual Peripheral module and place them in as per the template in the main source code to meet the sine wave requirements.

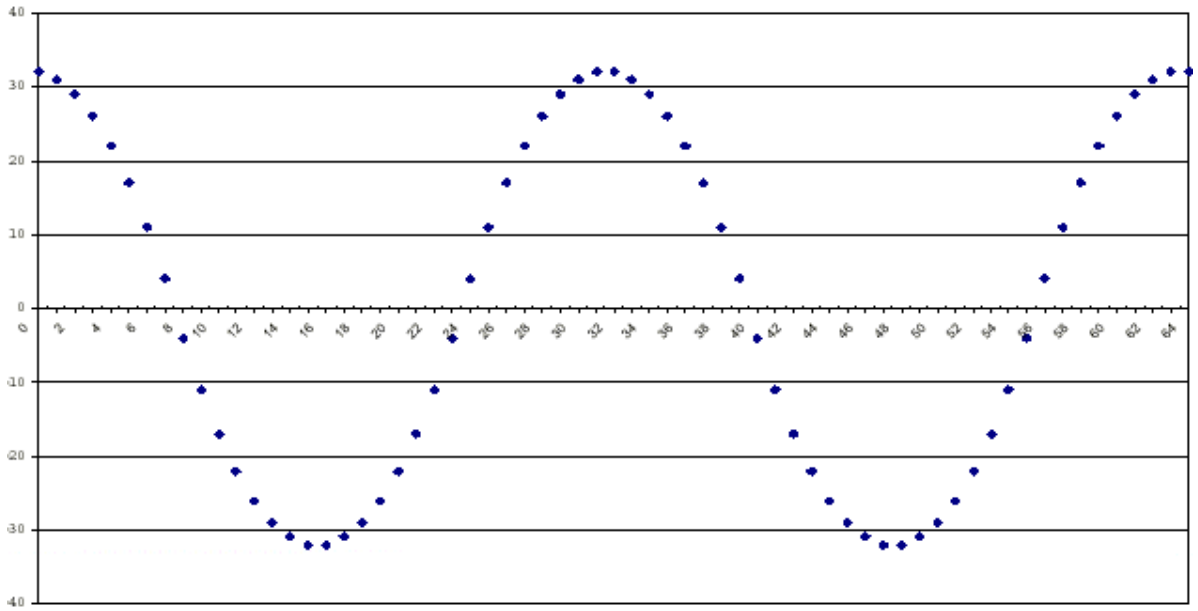

Figure 5-1. Imitation Sine Wave Output

Lit #: AN11-03

Sales and Tech Support Contact Information

For the latest contact and support information on SX devices, please visit the Ubicom website at www.ubicom.com. The site contains technical literature, local sales contacts, tech support and many other features.

1330 Charleston Road
Mountain View, CA 94043
Contact: Sales@ubicom.com
http://www.ubicom.com
Tel.: (650) 210-1500
Fax: (650) 210-8715