Artificial Sin Wave Generation Using the SX Microcontroller

Chris Fogelklou, Scenix Semiconductor, Inc.

Introduction

Sin waves are used extensively in the telecommunications industry, but they are traditionally difficult to implement in software without using code-eating 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 sin wave. This signal is close enough to the real thing to perform applications such as DTMF (Dual-Tone Multi-Frequency) generation, FSK generation, PSK generation, and many other applications that require frequency generation.

In the past, such telephony functions 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 used. Now, the 8-bit Scenix Semiconductor SX Series MCUs, which have performance reaching 100 MIPS (million instructions per second) and a deterministic interrupt architecture, overcome this roadblock by providing the ability to perform these functions in software.

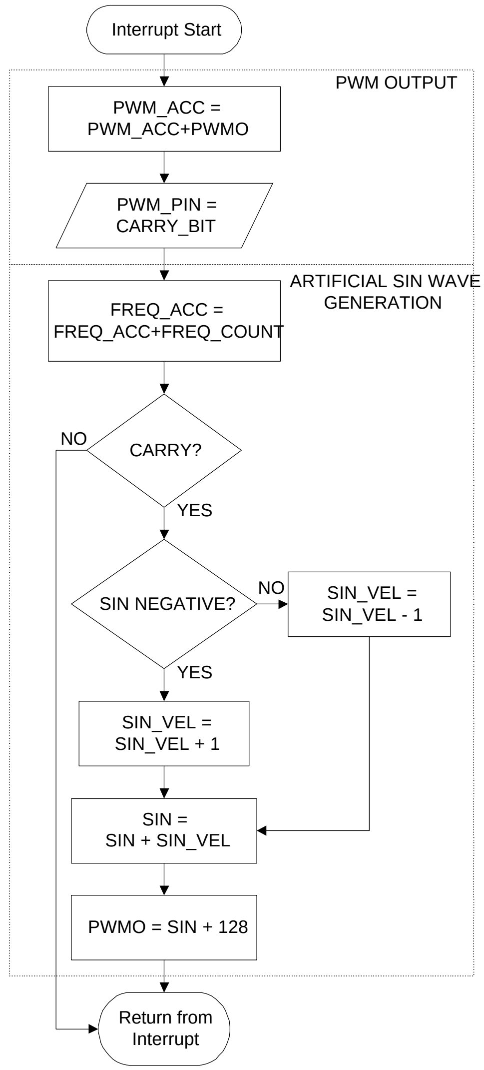
Unlike other MCUs that add functions in the form of additional silicon, the SX Series uses its industry-leading performance to execute functions as software modules, or Virtual Peripherals. These are loaded into a high-speed (10 ns access time) on-chip flash/EEPROM program memory and executed as required. 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.

How It Works

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 if there was no ground, just a center of gravity? The ball would continue past the center of gravity, once again decelerating to a velocity of zero and re-accelerating towards the center of gravity, and the cycle would continue…

Chris Fogelklou, Scenix Semiconductor, Inc.

This type of algorithm can be implemented in an interrupt service routine using this flowchart…



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.

<u>Timing</u>

The first process of the artificial sin wave generator is to determine if it is time to update the value of the sin wave. The 16-bit FREQ_COUNT variable determines the rate at which the wave is updated. The FREQ_COUNT variable is added into the FREQ_ACC variable on each pass of the ISR, and the sine wave is updated with a new value every time the FREQ_ACC variable rolls over. An entire cycle of the sine wave is made up of 32 separate values. If we combine these factors with the interrupt rate of 3.26us, we can calculate the value to load into the FREQ_COUNT variable 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 once.

1 Period = 32 separate values.

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

Therefore 1 period = 2 097 152 interrupts

Since the ISR rate = 3.26us.

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

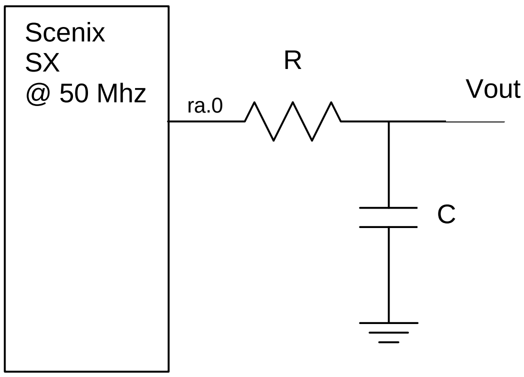
Therefore frequency = 0.14627Hz.

Therefore, resolution = 0.14627 Hz
Maximum output frequency = 9.6kHz.
Output frequency = FREQ_COUNT * 0.14627Hz
FREQ_COUNT = (desired frequency) * 6.8367

<u>Creating The Wave</u>

This is the easy part. The program just increments the velocity (accelerates) if the wave is negative, or decrements the velocity (decelerates) if the wave is positive. This new velocity is added to the current value of the sin wave. The final task is to load the new value of the sin wave into the PWM register, and to add #128 to the PWM output to center the wave at 2.5VDC.

Chris Fogelklou, Scenix Semiconductor, Inc.



Circuit Design

The simplest version of the circuit requires only two components for the PWM output, a resistor and a capacitor. ← Here is a block diagram of the circuit.

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