

PWM ≠ Comparator, an accurate event based digital PWM generation modeling

By Arief Noor Rahman – Power Control Design

First, let's look at the following circuit. It's a quite mundane single phase inverter which is controlled with unipolar sine PWM method.

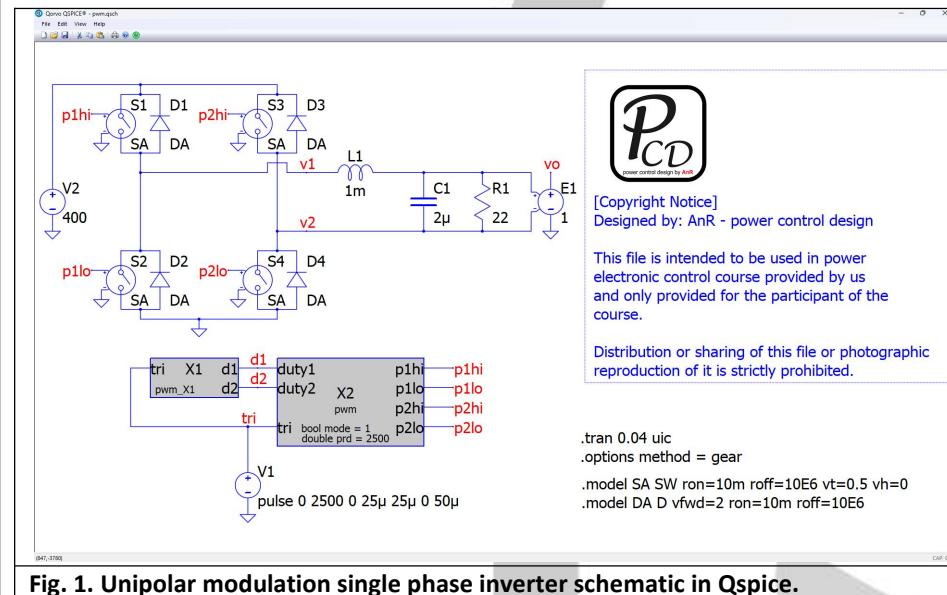


Fig. 1. Unipolar modulation single phase inverter schematic in Qspice.

In this presentation, I have prepared two methods to generate PWM which can be selected by setting the mode in the PWM block to be 0 or 1. For setting mode = 0, the PWM operates as simple comparators that compare the duty command (d1 and d2) with the triangle carrier.

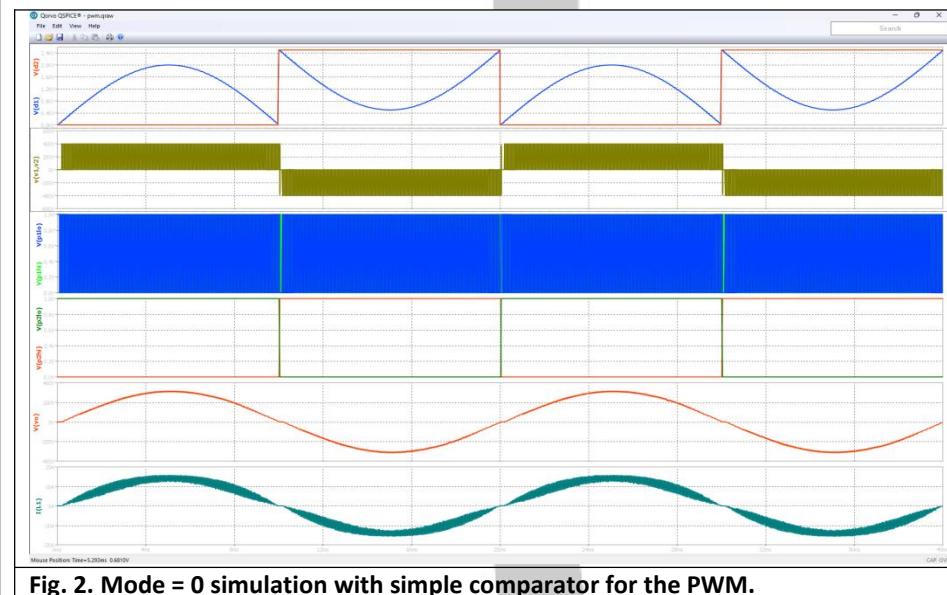


Fig. 2. Mode = 0 simulation with simple comparator for the PWM.

While using simple comparators is acceptable for most converter simulations, nevertheless it will actually fail to accurately simulate the case where the duty command abruptly changes from near 0 to near PRD on the subsequent switching period which occurred at the zero crossing for a unipolar sine PWM modulation and totem pole PFC. The resulting simulation with simple comparators will show a clean transition at both zero crossings which is not actually true when compared to hardware measurement.

To improve the digital PWM generation in simulation, we need to first understand the method used inside an MCU as depicted in Fig. 3. The main difference is PWM generation is actually event based instead of comparator based. That evaluates the output at discrete events when (1) counter == ZERO, (2) counter == PRD, (3) carrier == CMPA/B instead of continuously as used with simple comparators. In my guess, there could be two reasons for this method: (1) to follow the standard practice as used in analog PWM IC which uses comparator + SR latch to avoid spurious switching due to noise or (2) smaller silicon area compared to comparators.

S/W force	TB Counter equals:					Actions
	Zero	Comp A	Comp B	Period		
SW X	Z X	CA X	CB X	P X		Do Nothing
SW ↓	Z ↓	CA ↓	CB ↓	P ↓		Clear Low
SW ↑	Z ↑	CA ↑	CB ↑	P ↑		Set High
SW T	Z T	CA T	CB T	P T		Toggle

Fig. 3. The rule definition for the event based PWM output generation used in C2000 MCU family.

Ref: Texas Instruments, "TMS320F2803x Microcontrollers - Technical Reference Manual", 2022.

```
(a) inst->p1 = inst->d1_sample > tri;
if(inst->d1_sample <= 0.5)
{
    if((tri >= inst->tri_prev) && (inst->tri_prev2 >= inst->tri_prev))
    {
        inst->p1 = 0;
    }
}
else if(inst->d1_sample >= (prd - 0.5))
{
    if((tri <= inst->tri_prev) && (inst->tri_prev2 <= inst->tri_prev))
    {
        inst->p1 = 1;
    }
}
else
{
    if((tri >= inst->d1_sample) && (inst->tri_prev <= inst->d1_sample))
    {
        inst->p1 = 0;
    }
    if((tri <= inst->d1_sample) && (inst->tri_prev >= inst->d1_sample))
    {
        inst->p1 = 1;
    }
}
```

Code snippet 1: (a) simple comparator based PWM (mode = 0), (b) event based PWM generation (mode = 1). Both implemented in Qspice C-code.

From code snippet 1, we can observe the difference implementation between simple comparator (mode = 0) and proper event based PWM (mode = 1). For mode = 1, the algorithm consisted of checking if the duty command is 0 or PRD then force the output to switch right at the proper carrier discontinuity even, and if duty is in between 0 and PRD, output switching will only occur right at the carrier == CMPA/B.

Additional algorithms for the PWM C-code are, duty command update synchronization and complementary output generation with deadtime with the detail can be evaluated directly on the source code available in Github.

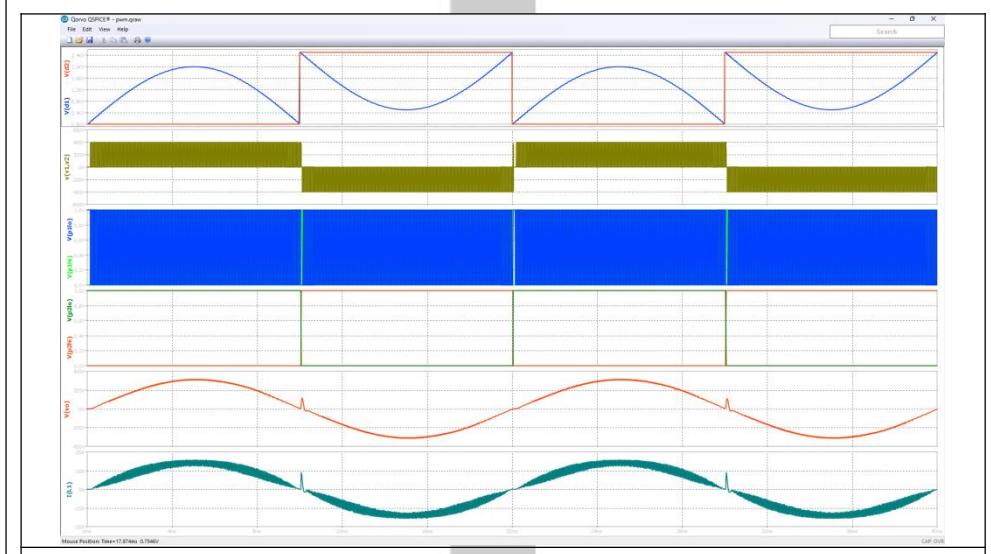


Fig. 4. Mode = 1 simulation with proper event based PWM generation

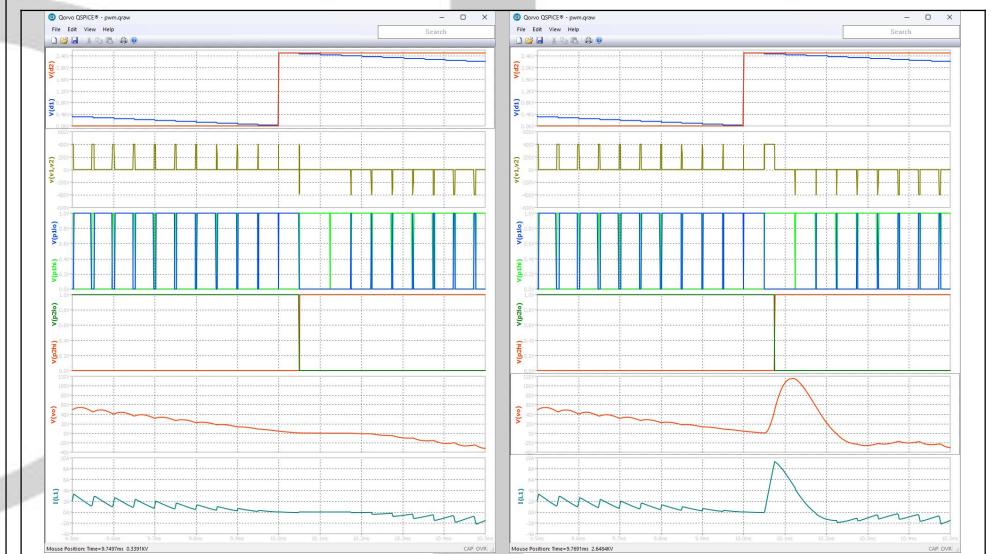


Fig. 5. Waveform comparison at positive to negative zero crossing between mode = 0 and mode = 1.

The simulation result with mode = 1 is shown in Fig. 4. Compared to mode = 0 result, the difference is clearly visible at positive to negative output zero crossing where a significant spike/disturbance occurred. From Fig. 5, we can see how the current spike occurred due to, despite the d1 and d2 are synchronized but the

p1 and p2 outputs switching are occurred slightly differently with p2 slightly delayed. Which then causes sudden large duration of positive pulse between leg 1 and leg 2 switching node v(v1,v2), and thus sudden increase in output voltage and current accordingly.

Conclusion:

1. The simple PWM comparator while suitable for most converter topology but has been proven to be inaccurate for application with large change in duty command where it will not show the realistic distortion that occurred on the real MCU.
2. The effect of PWM output inaccuracy causes significant problem when simulating inverter with unipolar modulation or totem pole PFC with example and reason provided.
3. This algorithm for event based PWM generation that closely resemble actual MCU PWM generation has been introduced with some explanation according for the short snippet. And the whole code to be accessible from my Github page.
4. The PWM algorithm provided still uses external carrier for clarity for general reader as it is hopefully somewhat easier to understand.

Note: This version is actually a much simplified version from another version my internally developed for self use of whole MCU timing control (PWM and sampling) where all the timing are handled with accurate future discontinuity event prediction.

Such approach is possible for digital control modeling due to the inherent sampling delay behavior and predetermined timing event control for period K according to computation at period K-1.

However, the code is going to be too complex to explain, as some friend whom I tried to teach about the code are all given up understanding the code.

power control design by **AnR**