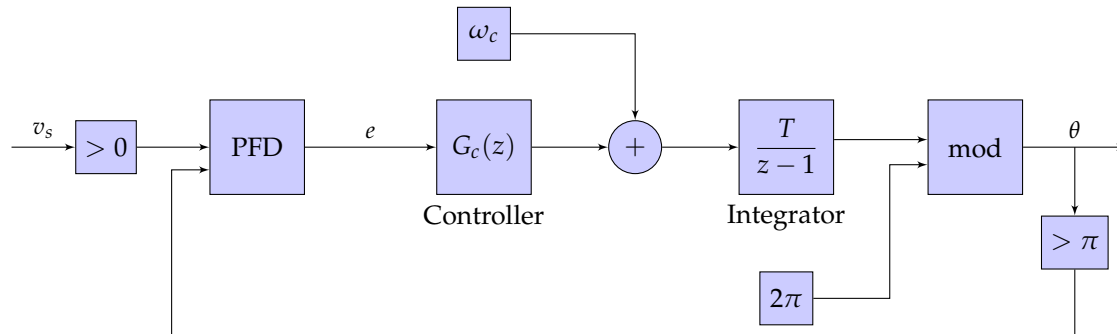


## DIGITAL CONTROL OF POWER ELECTRONICS

## Single Phase PLL using fixed point Q format



We are going to convert the low pass filter design to a digital controller. Below are the gain, pole location, and transfer function for the controller/ low pass filter/ loop filter.

$$k = 105075 \quad \omega_p = 258.6 \quad (1)$$

$$G_{LF}(s) = \frac{105075}{s + 258.6} \quad (2)$$

Below is the matlab code to develop the digital controllers for a low pass controller

```

1  % low pass single phase pll
2  clear
3  syms s wp k p
4
5  gol = 1/(2*p) * k/(s+wp) * 1/s
6  gcl = gol/(1+gol)
7
8  clear
9  s=tf('s');
10 opts = bodeoptions('cstprefs');
11 opts.FreqUnits = 'Hz';
12 opts.XLim= {[.1 10000]}
13
14 kvco = 1
15
16 fbw = 10
17 zeta = 1
18 w = 2 * pi * fbw
19 a = 1
20 b = -(2^2*zeta^2*2*pi*w^2)
21 c = -(2*pi)^2*w^4
22
23 k1 = (-b+sqrt(b^2-4*a*c))/2
24 k2 = (-b-sqrt(b^2-4*a*c))/2
25 wf = 2*zeta*sqrt(k1/(2*pi))
26
27 GLPF = k1/(s+wf)
28 %digital controller
29 Ts = 1/10e3
30 Z=tf('z',Ts)

```

```
31 sback = ((Z-1)/(Z*Ts))
32 gc_dig_backward = k1 / (sback +wf)
33 gc_dig_zoh = c2d(GLPF,1/10e3,'zoh')
34 gc_dig_tustin = c2d(GLPF,1/10e3,'tustin')
35
36
37 %% calculate coeff for fixed point Q21 backward euler
38 Q = 21;
39 Bcoeff = int32(gc_dig_backward.Numerator{1}*2^Q);
40 Acoeff = int32(gc_dig_backward.Denominator{1}*2^Q);
41
42 fprintf('int32 A1 = %i;\n',Acoeff(1))
43 fprintf('int32 A0 = %i;\n',Acoeff(2))
44 fprintf('int32 B1 = %i;\n',Bcoeff(1))
45 fprintf('int32 B0 = %i;\n',Bcoeff(2))
46
47 %% calculate coeff for fixed point Q21 tustin
48 Q = 21;
49 Bcoeff = int32(gc_dig_tustin.Numerator{1}*2^Q);
50 Acoeff = int32(gc_dig_tustin.Denominator{1}*2^Q);
51
52 fprintf('int32 A1 = %i;\n',Acoeff(1))
53 fprintf('int32 A0 = %i;\n',Acoeff(2))
54 fprintf('int32 B1 = %i;\n',Bcoeff(1))
55 fprintf('int32 B0 = %i;\n',Bcoeff(2))
```

Below are the coefficients for backwards euler.

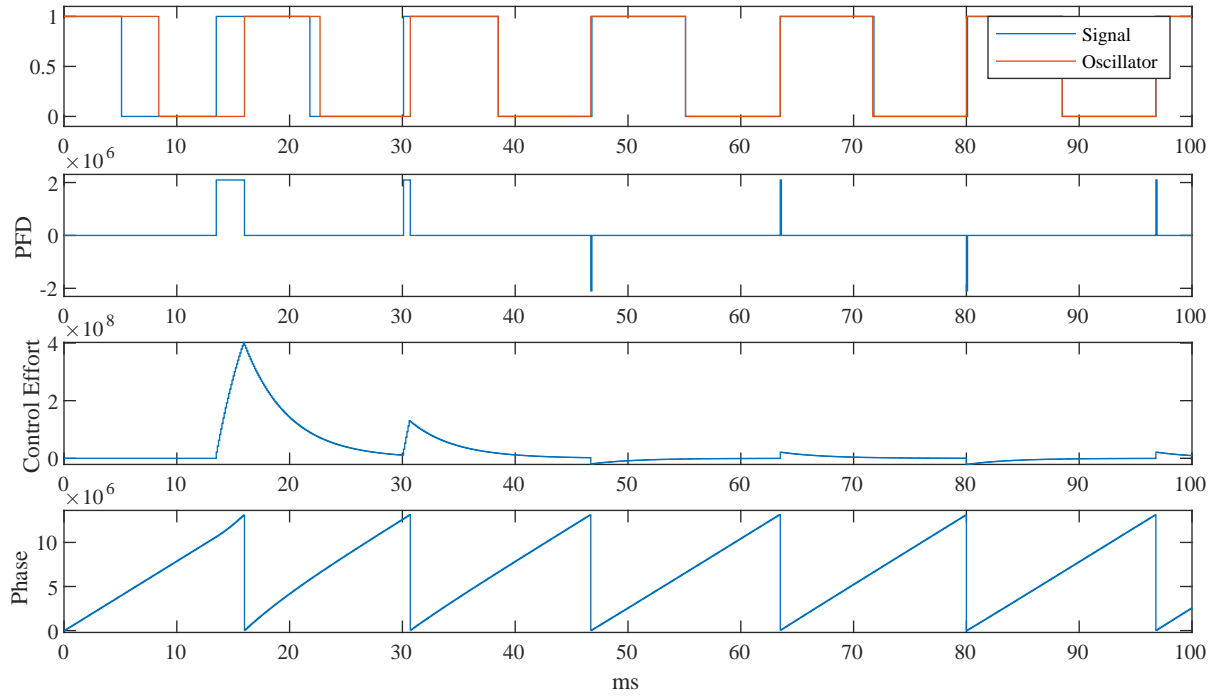
```
1 int32 A1 = 2151392;
2 int32 A0 = -2097152;
3 int32 B1 = 22035983;
4 int32 B0 = 0;
```

Below are the coefficients for tustin.

```
1 int32 A1 = 2097152;
2 int32 A0 = -2043604;
3 int32 B1 = 10877327;
4 int32 B0 = 10877327;
```

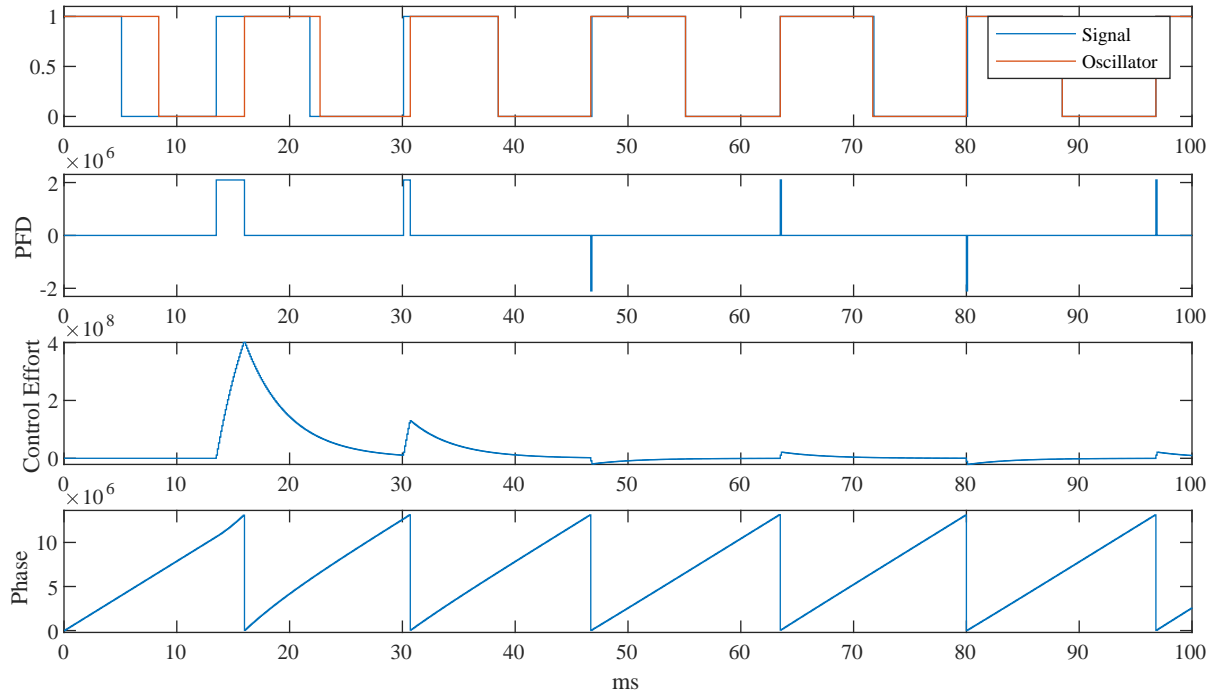
Use backward euler we arrive at a controller with.

$$G_c(z) = \frac{10.51z}{1.026z - 1} \quad (3)$$



The tustin transformation will arrive a the below controller

$$G_c(z) = \frac{5.187z + 5.187}{z - 0.9745} \quad (4)$$



```
1 %% PI controller
2 damping = 1
3
4 omega_n = 2*pi*4
5
6 kp = 4*pi*damping*omega_n
7 ki = 2*pi*omega_n^2
8 GLPF = ki/s + kp
9
10 %digital controller
11 Ts = 1/10e3
12 Z=tf('z',Ts)
13 sback = ((Z-1)/(Z*Ts))
14 gc_dig_backward = ki/sback + kp
15 gc_dig_zoh = c2d(GLPF,1/10e3,'zoh')
16 gc_dig_tustin = c2d(GLPF,1/10e3,'tustin')
17
18 %% calculate coeff for fixed point Q21 backward euler
19 Q = 21;
20 Bcoeff = int32(gc_dig_backward.Numerator{1}*2^Q);
21 Acoeff = int32(gc_dig_backward.Denominator{1}*2^Q);
22
23 fprintf('int32 A1 = %i;\n',Acoeff(1))
24 fprintf('int32 A0 = %i;\n',Acoeff(2))
25 fprintf('int32 B1 = %i;\n',Bcoeff(1))
26 fprintf('int32 B0 = %i;\n',Bcoeff(2))
27
28 %% calculate coeff for fixed point Q21 tustin
29 Q = 21;
30 Bcoeff = int32(gc_dig_tustin.Numerator{1}*2^Q);
31 Acoeff = int32(gc_dig_tustin.Denominator{1}*2^Q);
32
33 fprintf('int32 A1 = %i;\n',Acoeff(1))
34 fprintf('int32 A0 = %i;\n',Acoeff(2))
35 fprintf('int32 B1 = %i;\n',Bcoeff(1))
36 fprintf('int32 B0 = %i;\n',Bcoeff(2))
```

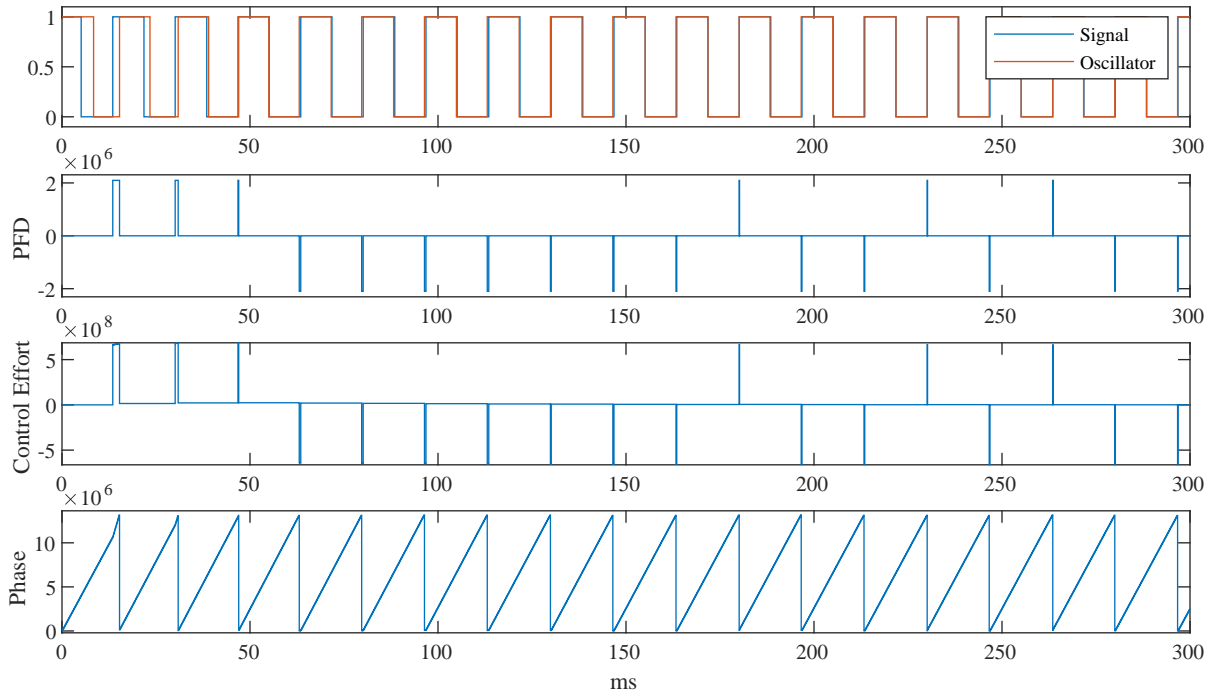
Below are the coefficients for backwards euler.

```
1 int32 A1 = 2097152;
2 int32 A0 = -2097152;
3 int32 B1 = 663170258;
4 int32 B0 = -662337939;
```

Below are the coefficients for tustin.

```
1 int32 A1 = 2097152;
2 int32 A0 = -2097152;
3 int32 B1 = 662754099;
4 int32 B0 = -661921780;
```

$$G_c(z) = \frac{316.2z - 315.8}{z - 1} \quad (5)$$



$$G_c(z) = \frac{316z - 315.6}{z - 1} \quad (6)$$

