Analysis, Design and Implementation of Phase-Locked-Loop (PLL) for Grid-Connected Inverters

Dinesh Gopinath

Associate Professor
Department of Electrical Engineering
College of Engineering Trivandrum
Thiruvananthapuram

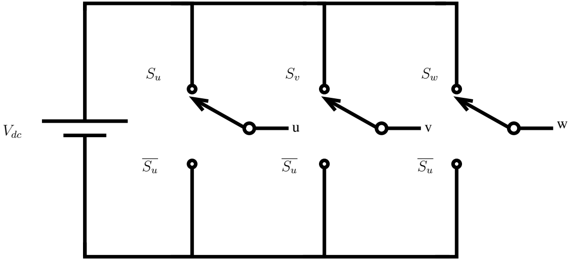
Outline

- Introduction
- Grid-Connected Inverters
- Phase-Locked-Loop for grid-connected converters
- Different PLL schemes
- Simulation Results
- Implementation in Digital Platform (FPGA)
- Experimental Results
- Conclusion

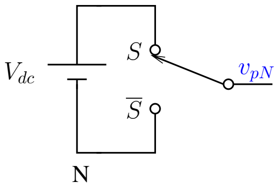
Introduction

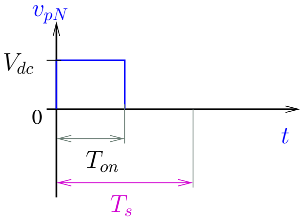
- Inverters are the interfaces for distributed energy sources with the grid
- Control of grid-connected inverters need the phase information of the source
- Phase of the source can be extracted by a Phase-Locked-Loop
- PLL can be implemented in Software
- There are different schemes of PLL implementation
- The issues are:
  - Accuracy
  - Speed
  - Disturbance rejection
  - Effect of Harmonics, Phase unbalance etc.

Three-Phase Inverters

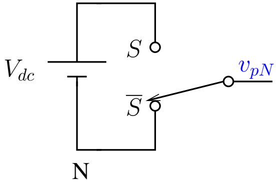


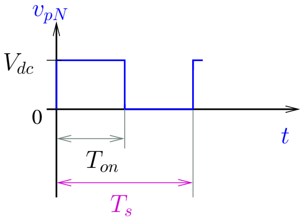
Fundamentals of inverters



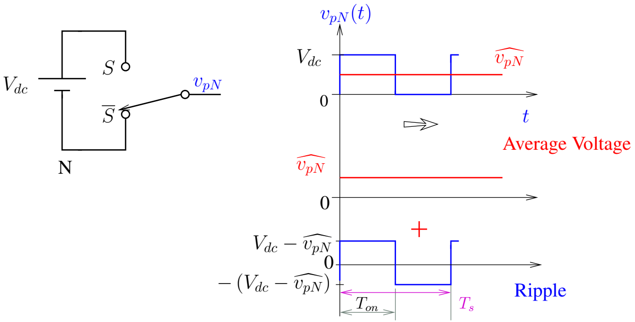


Fundamentals of inverters





Fundamentals of inverters



$$v_{pN}(t) = V_{dc} \cdot S$$

$$\widehat{v_{pN}} = V_{dc} \cdot \frac{T_{on}}{T_s} = V_{dc} \cdot d$$

S=1 or 0

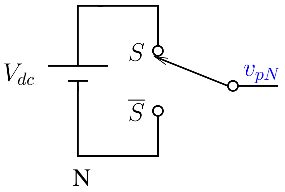
'd' is the duty ratio

$$v_{pN}(t) = d \cdot V_{dc} + v_{pN}(h)$$

Fundamentals of inverters

- If the switch ‘S’ is switched at high frequency, all the harmonics will occur at the switching frequency
- High frequency harmonics can be easily filtered by a small low-pass filter
- Or, if the waveform feeds an R-L load, the current will be mostly following the average value $\widehat{v_{pN}}$

Sine-Triangle Pulse Width Modulation



$$\widehat{v_{pN}} = d \cdot V_{dc}$$

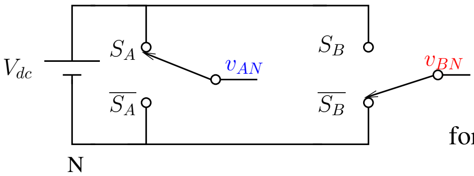
If we choose the duty ratio ‘d’ as:

$$d = 0.5 + 0.5 \cdot m \cdot \sin(\omega t)$$

Then,

$$\widehat{v_{pN}} = 0.5V_{dc} + 0.5 \cdot m \cdot V_{dc} \cdot \sin(\omega t)$$

Where, ‘m’ is modulation index, $0 \leq m \leq 1$, and ω is the required angular frequency.



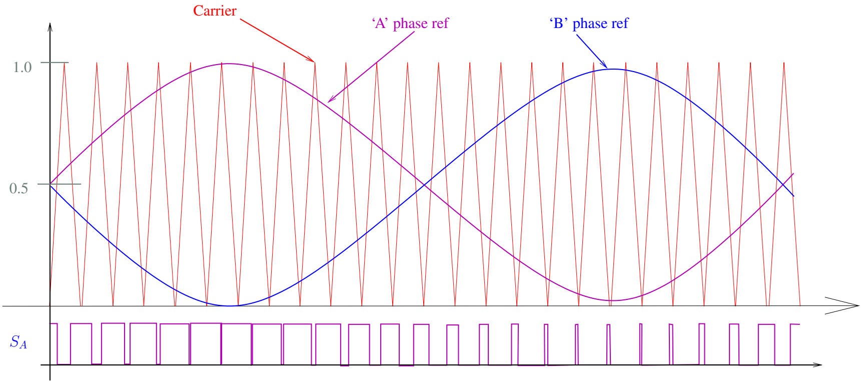
$$v_{AB} = v_A - v_B$$

$$\widehat{v_{AB}} = m \cdot V_{dc} \cdot \sin(\omega t)$$

for:

$$\begin{aligned} d_A &= 0.5 + 0.5 \cdot m \cdot \sin(\omega t) \\ d_B &= 0.5 - 0.5 \cdot m \cdot \sin(\omega t) \end{aligned}$$

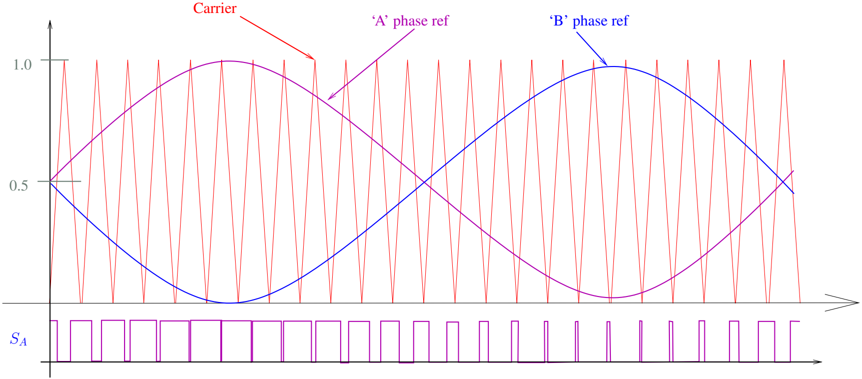
Sine-Triangle Pulse Width Modulation



The control law:
S_A ON: $v_{ref} > v_{tri}$
S_B ON : $-v_{ref} > v_{tri}$

For three-phase inverter, three sine wave references which are 120^o phase separated are used.

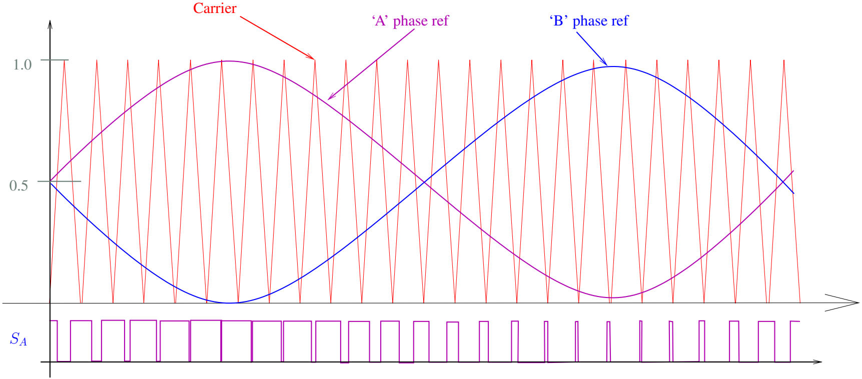
Sine-Triangle Pulse Width Modulation



The control law:
S_A ON: $v_{ref} > v_{tri}$
S_B ON : $-v_{ref} > v_{tri}$

ω is the angular frequency of the required output voltage.
$\omega t = \theta$ is the phase of the output at any instant.

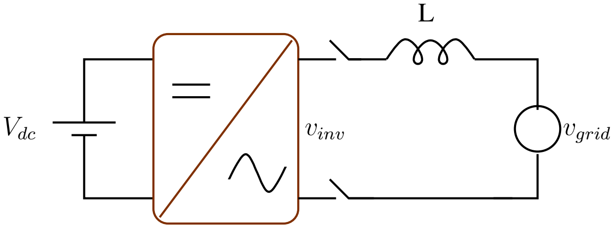
Sine-Triangle Pulse Width Modulation



The control law:
S_A ON: $v_{ref} > v_{tri}$
S_B ON : $-v_{ref} > v_{tri}$

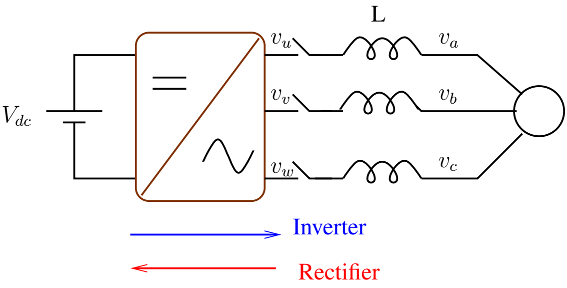
For other modulation schemes such as Space Vector PWM also, the phase of the output voltage is required to derive the duty ratios.

Grid-connected inverters



- Voltage, frequency should be the same
- Phase can be used to control the power-flow

Grid-connected inverters

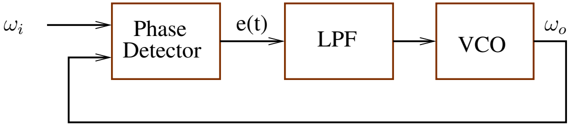


- Voltages, frequency should be the same
- Phase sequence must be the same
- Phase can be used to control the power-flow

Grid-connected inverters

- The phase of the grid voltages is a must for active/reactive power control
- The phase can be obtained by Phase-Locked-Loop

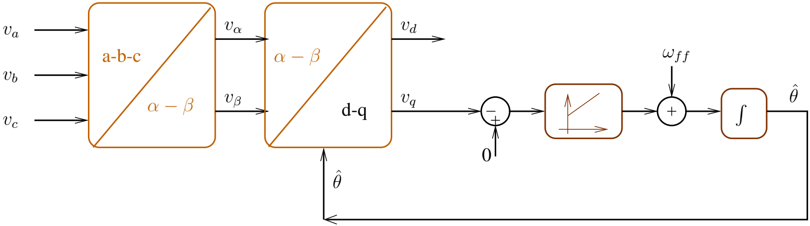
Phase-Locked-Loop



Objective: To synthesize the phase/frequency information of the system accurately

- Will a Zero-Crossing-Detector (ZCD) do the job?
- ZCD based tracking is slow
- Quadrature waveform technique is another method
- Not the best method when the frequency is varying/ has harmonics

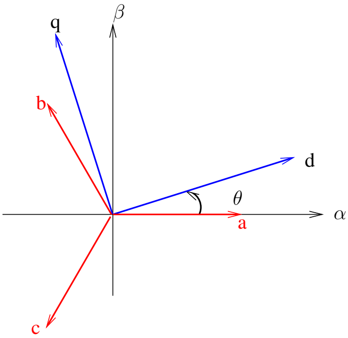
3-phase Phase-Locked-Loop: d-q frame based PLL



$$v_{as} = V_m \cos\theta \quad (1)$$

$$v_{bs} = V_m \cos\left(\theta - \frac{2\pi}{3}\right) \quad (2)$$

$$v_{cs} = V_m \cos\left(\theta + \frac{2\pi}{3}\right) \quad (3)$$



$$\begin{aligned} v_\alpha &= v_{as} - \frac{1}{2}v_{bs} - \frac{1}{2}v_{cs} \\ &= \frac{3}{2}v_{as} \quad (4) \\ v_\beta &= \frac{\sqrt{3}}{2}\left(v_{bs} - v_{cs}\right) \quad (5) \end{aligned}$$

Synchronously rotating reference frame, assuming no zero-sequence components,

$$v_d = v_\alpha \cos\theta + v_\beta \sin\theta \quad (6)$$

$$v_q = -v_\alpha \sin\theta + v_\beta \cos\theta \quad (7)$$

Let $\hat{\theta}$ be the PLL's output, which is an estimated value. Then,

$$\begin{bmatrix} v_d \\ v_q \end{bmatrix} = \begin{bmatrix} \cos\hat{\theta} & \sin\hat{\theta} \\ -\sin\hat{\theta} & \cos\hat{\theta} \end{bmatrix} \cdot \begin{bmatrix} v_\alpha \\ v_\beta \end{bmatrix} \quad (8)$$

Substituting the expressions for v_α and v_β from Eqns. (4) and (5), we get:

$$v_d = \frac{3}{2} V_m \cos(\theta - \hat{\theta}) \quad (9)$$

Let $(\theta - \hat{\theta}) = \delta$. Then when the PLL is estimating θ closely,

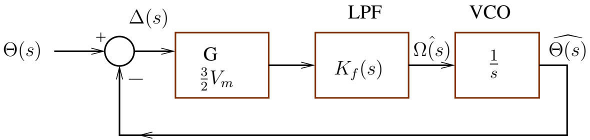
$$v_d \approx \frac{3}{2} V_m \quad (10)$$

Similarly we get,

$$v_q = \frac{3}{2} V_m \sin(\theta - \hat{\theta}) \quad (11)$$

$$= \frac{3}{2} V_m \sin \delta \quad (12)$$

Now, for very small δ, the loop can be linearised.



Now, $\hat{\omega} = \frac{d\hat{\theta}}{dt} = K_f \cdot e$.

Where, $e = v_q = \dfrac{3}{2} V_m \sin \delta$, and K_f is the gain of the loop-filter.

For $\hat{\theta} \approx \theta$, $\sin \delta \approx \delta$

$\therefore\ e = \frac{3}{2} V_m \cdot \delta$

3-Phase PLL : Contd...

Closed-loop transfer function is,

$$H_c(s) = \frac{\hat{\Theta}(s)}{\Theta(s)} = \frac{E_m K_f(s)}{s + E_m K_f(s)}$$

Where, $E_m = \frac{3}{2} V_m$.
We can use a P-I controller for $K_f(s)$, given as:

$$K_f(s) = K_p + \frac{K_I}{s} = K_p \frac{1 + s\tau}{s\tau}$$

Then,

$$H_c(s) = \frac{K_p E_m \cdot s + \frac{K_p E_m}{\tau}}{s^2 + K_p E_m s + \frac{K_p E_m}{\tau}}$$

3-Phase PLL : Contd...

Comparing with the standard second-order form:
The natural frequency,

$$\omega_n = \sqrt{\frac{K_p E_m}{\tau}}$$

and the damping ratio,

$$\zeta = 0.5\sqrt{K_p E_m \tau}$$

Considering the sampling delays (T_s), the plant T.F is ,

$$H_{plant}(s) = \frac{1}{1 + sT_s} \frac{1}{s} E_m$$

The open-loop T.F is then,

$$H_{OL}(s) = K_p \cdot \frac{1 + s\tau}{s\tau} \cdot E_m \frac{1}{s} \frac{1}{1 + sT_s}$$

Design of the PI controller

Using the symmetric optimum method, PI controller's cross-over frequency, $\omega_c = \frac{1}{\alpha T_s}$.
and,

$$\tau = \alpha^2 T_s$$

$$K_p = \frac{1}{\alpha} \cdot \frac{1}{E_m T_s}$$

Where, α is a scalling factor.
For these values,

$$\zeta = \frac{\alpha - 1}{2}$$

α	2
T_s	$102.4\mu s$
τ	$409.6\mu s$
V_m	339 V
K_p	9.6
ω_c	777.12 Hz

For critical damping, $\zeta = 0.707$,

ζ	0.707
α	2.414
T_s	$102.4\mu s$
τ	$596.73\mu s$
K_p	7.96
ω_c	643.85 Hz

An alternate method

Settling time (3 ac cycles),

$$t_s = \frac{4}{\zeta\omega_n}$$

$$K_p = \omega_n \zeta \frac{2}{E_m}$$

$$T_p = \frac{4\zeta^2}{K_p E_m}$$

$$K_I = \frac{K_p}{T_p}$$

$$K_{P(pu)} = K_p \cdot \frac{V_{base}}{\omega_{base}}$$

An alternate method

Settling time (3 ac cycles),

$$t_s = \frac{4}{\zeta \omega_n}$$

$$K_p = \omega_n \zeta \frac{2}{E_m}$$

$$T_p = \frac{4\zeta^2}{K_p E_m}$$

$$K_I = \frac{K_p}{T_p}$$

$$K_{P(pu)} = K_p \cdot \frac{V_{base}}{\omega_{base}}$$

An alternate method

Settling time (3 ac cycles),

$$t_s = \frac{4}{\zeta \omega_n}$$

$$K_p = \omega_n \zeta \frac{2}{E_m}$$

$$T_p = \frac{4\zeta^2}{K_p E_m}$$

$$K_I = \frac{K_p}{T_p}$$

$$K_{P(pu)} = K_p \cdot \frac{V_{base}}{\omega_{base}}$$

An alternate method

Settling time (3 ac cycles),

$$t_s = \frac{4}{\zeta \omega_n}$$

$$K_p = \omega_n \zeta \frac{2}{E_m}$$

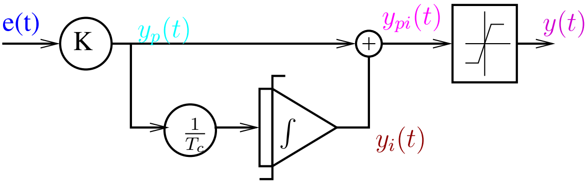
$$T_p = \frac{4\zeta^2}{K_p E_m}$$

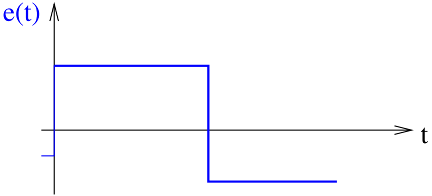
$$K_I = \frac{K_p}{T_p}$$

$$K_{P(pu)} = K_p \cdot \frac{V_{base}}{\omega_{base}}$$

PI Controller

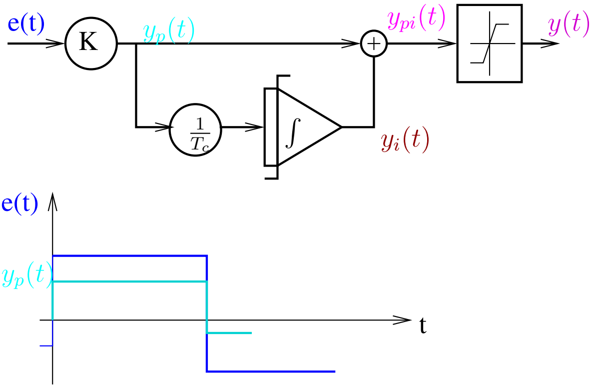
- Zero steady state error
- Better stability prospects compared to I - controller





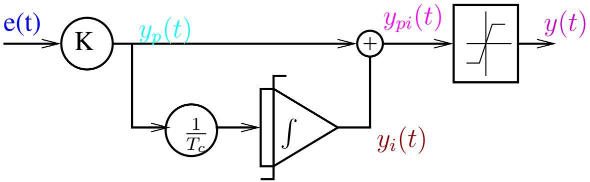
PI Controller

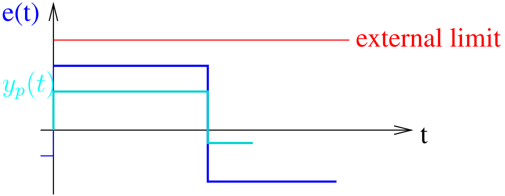
- Zero steady state error
- Better stability prospects compared to I - controller



PI Controller

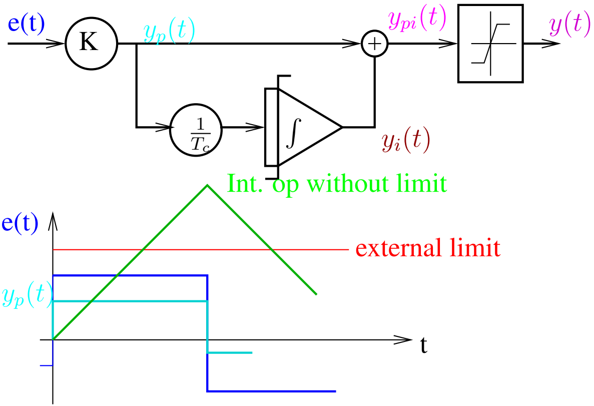
- Zero steady state error
- Better stability prospects compared to I - controller



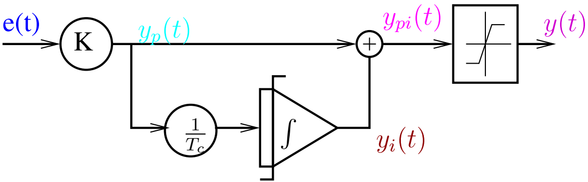


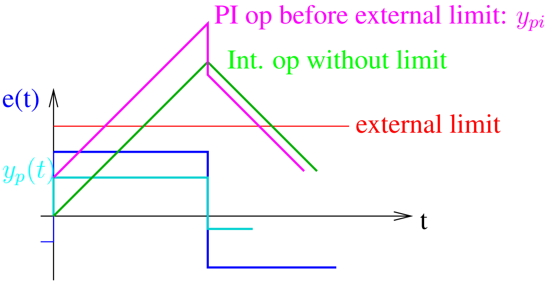
PI Controller

- Zero steady state error
- Better stability prospects compared to I - controller



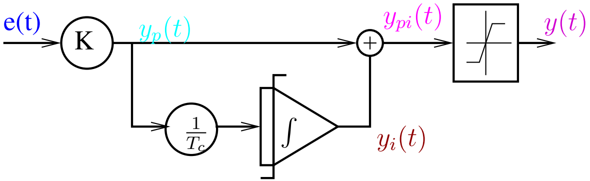
- Zero steady state error
- Better stability prospects compared to I - controller

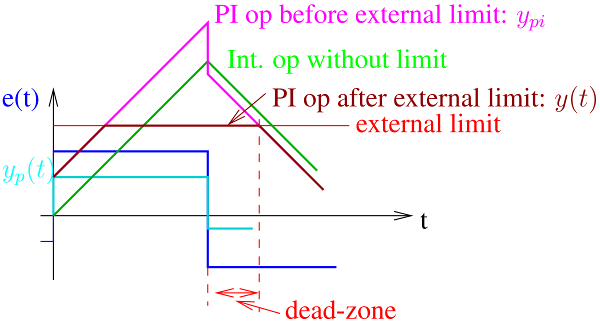




PI Controller

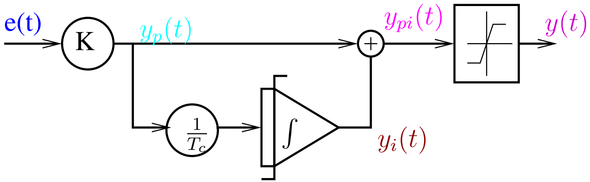
- Zero steady state error
- Better stability prospects compared to I - controller

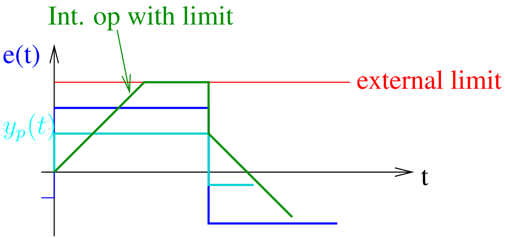




PI Controller

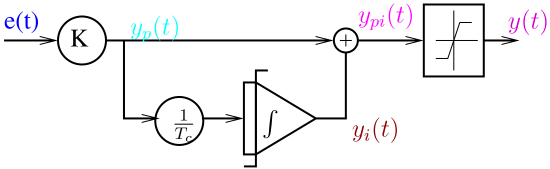
- Zero steady state error
- Better stability prospects compared to I - controller

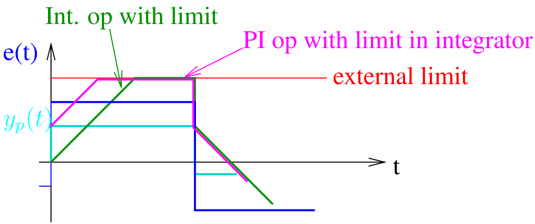




PI Controller

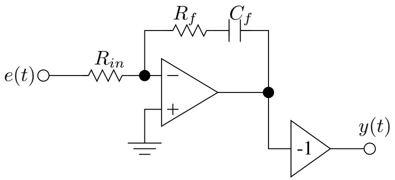
- Zero steady state error
- Better stability prospects compared to I - controller





P-I Implementation

Analog



$$y(t) = \frac{R_f}{R_{in}} e\,(t) + \frac{1}{R_{in} C_f} \int e\,(t)\,dt$$

Digital

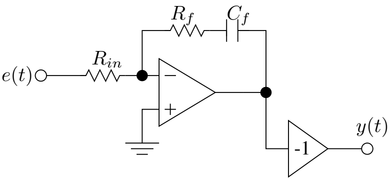
$$y(n) = Ke(n) + \frac{K}{T}\left[\frac{e(n) + e(n-1)}{2}\right] T_s + y(n-1)$$

PI Controller Transfer Function:

$$\frac{K(1+sT_c)}{sT_c}$$

P-I Implementation

Analog



$y(t) = \frac{R_f}{R_{in}} e\left(t\right) + \frac{1}{R_{in}C_f} \int e\left(t\right) dt$

Digital

$$y(n) = Ke(n) + \frac{K}{T}\left[\frac{e(n) + e(n-1)}{2}\right] T_s + y(n-1)$$

PI Controller Transfer Function:

$$\frac{K(1+sT_c)}{sT_c}$$

Digital Implementation: Fixed-point Vs Floating-Point

- Per-unit systems is adopted for fixed-point implementations
- A base value is chosen for all variables (voltage, frequency, phase, current etc)
- All variables are expressed in p.u
- p.u values are converted in to digital equivalents using fixed-point arithmetic
- Base conversion is done as per accuracy requirements

Digital Implementation: An example

- Let the input voltage be varying in the range $230V \pm 10\%$.
- The maximum voltage is then 357.742V
- If the base voltage is chosen as 360V, the maximum possible voltage is 0.9937 p.u

Digital Implementation: An example

In digital implementation, suppose we use a 12 bit ADC with an input voltage range $\pm 10V$.
The ADC will give digital equivalent values as follows:

ADC input voltage	12 bit Digital output	p.u for 5V base 5V base	p.u for 10V base 10 V base
+10 V	7FFh	2 p.u	1 p.u
+5 V	3FFh	1 p.u	0.5 p.u
+2.5 V	1FFh	0.5 p.u	0.25 p.u
+1.25 V	FFh	0.25 p.u	0.125 p.u
0 V	000h	0 p.u	0 p.u
-1.25 V	F01h	-0.25 p. u	-0.125 p.u
-2.5 V	E01h	-0.5 p.u	-0.25 p. u
-5.0 V	C01h	-1.0 p.u	-0.5 p.u
-10 V	801h	-2.0 p.u	-1.0 p.u

Digital Implementation: An Example

We may choose the input voltage-sensor in different ways.
For example:

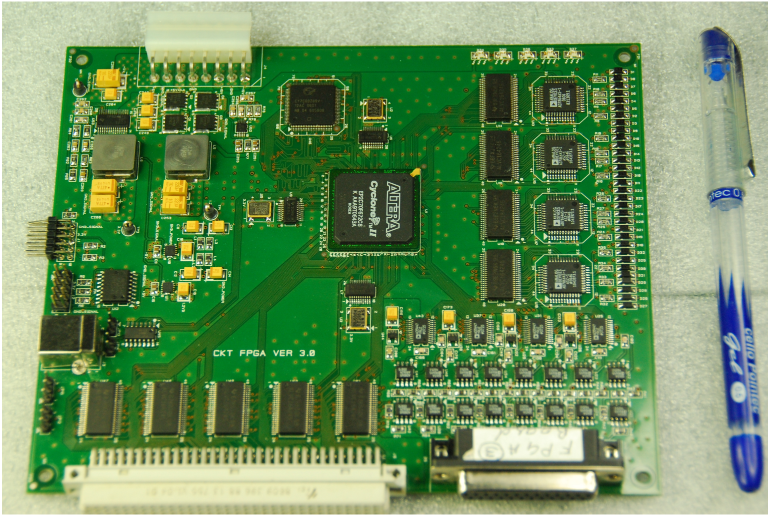
1. The nominal input voltage range of the ADC is $\pm 10V$ for the maximum expected input voltage
2. Leave enough room for the input voltage, and choose $\pm 5V$ as the nominal ADC input voltage range.

In the first case, we must choose the sensor-gain such that the ADC always get a voltage inside its range of $\pm 10V$.
In the second case, we must choose the sensor-gain in such a way that the ADC gets a nominal voltage of 5V when the input voltage is in the nominal range.
Here, however, we can have an unexpected higher voltage signal at the ADC input upto 2 p.u.
Normally, 2 p.u range is kept for signals like current, speed etc.

Some considerations

- Multiplication accuracy
  - Intermediate calculations should be done at higher base values
- Changing the base values
- Sampling frequency

FPGA based digital platform



FPGA platform: features

- 80 digital I/Os
- 16 Analog input channels (with 6.4 μs A/D conversion time per channel)
- 8 Analog output channels (with a DAC settling time of 80 ns)
- ALTERA EP2C70F672C8 FPGA with 68,416 logic elements
- USB and CAN transceiver interfaces
- On board SRAM (64K$\times$18)
- Three clocks at 20 MHz each
- JTAG interface

FPGA platform: Architecture

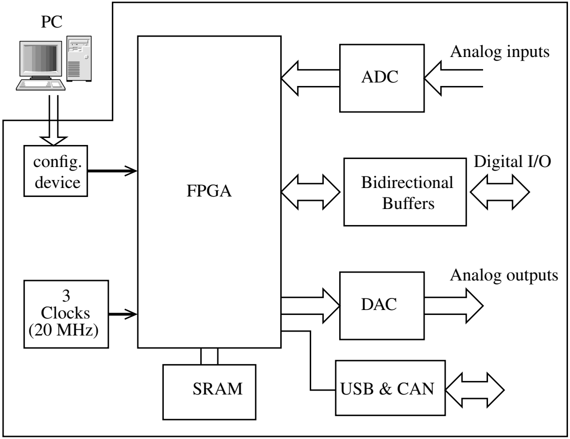
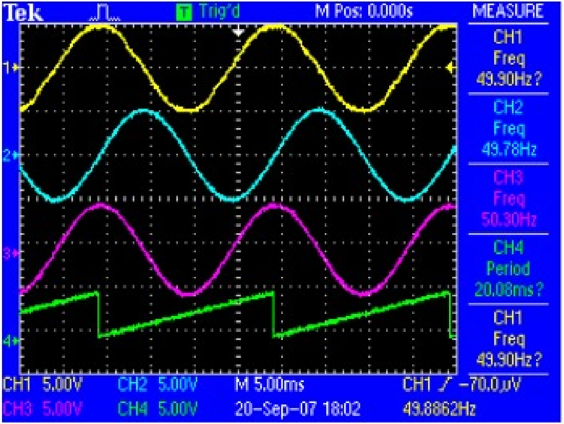


Figure: Block diagram of FPGA controller

Experimental results



CH1:Va; CH2: sine; CH3: cos; CH4:Theta

Conclusion

- Basics of Phase-Locked Loops have been explained
- PLLs can be easily implemented in software
- Digital implementation is particularly easy in FPGA platform
- There are several PLL methods which vary in complexity and accuracy

Thank You

This presentation was done in $\LaTeX$ and Beamer