Designing Stable Digital Power Supplies

Dr Ali Shirsavar
Biricha Digital Power Ltd

Agenda

- Welcome and Introduction
- Introduction to digital power
  - Benefits of digital power
  - Digital power challenges
- How can TI help with your DPS Designs
  - Digital Power solutions, development tools, boards and software from TI
  - Support from TI on your design including digital power training though Biricha Digital Power
- Step by step digital power design guidelines
  - Designing our first digital power supply
  - Converting analog controllers into digital using bilinear transforms
  - Phase margin erosion and stability in your power supplies
  - Demonstration of changing the loop's behaviour on the fly under digital control
- Questions and Answers

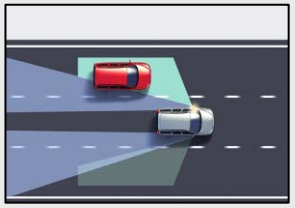
The Need for Power Management


Renewable Energy Generation


Telecom Digital Power



AC Drives, Industrial & Consumer Motor Control




Automotive Radar & Electric Power Steering

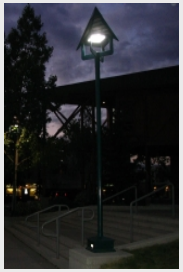
Power Management

Power Line Communications

Consumer & Automotive Digital Power

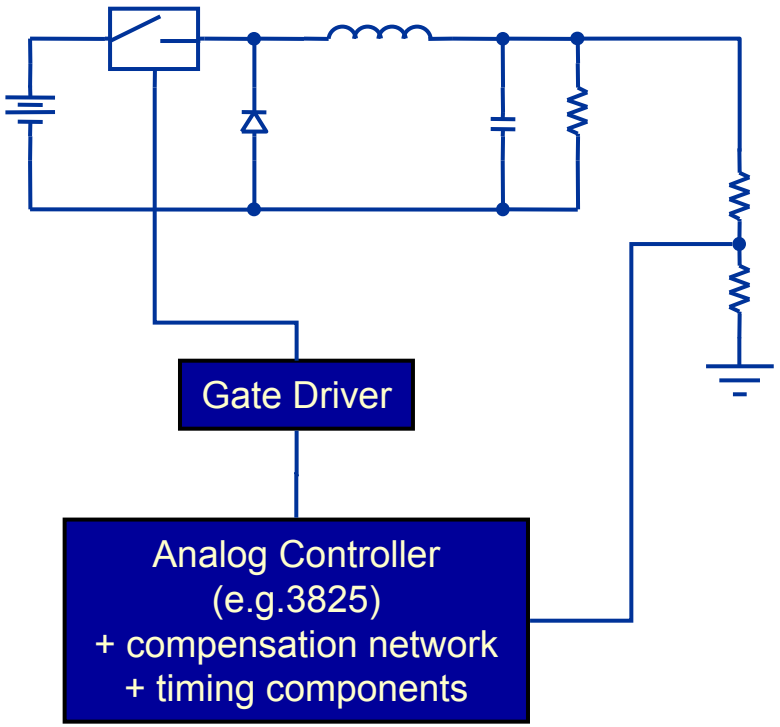


Lighting

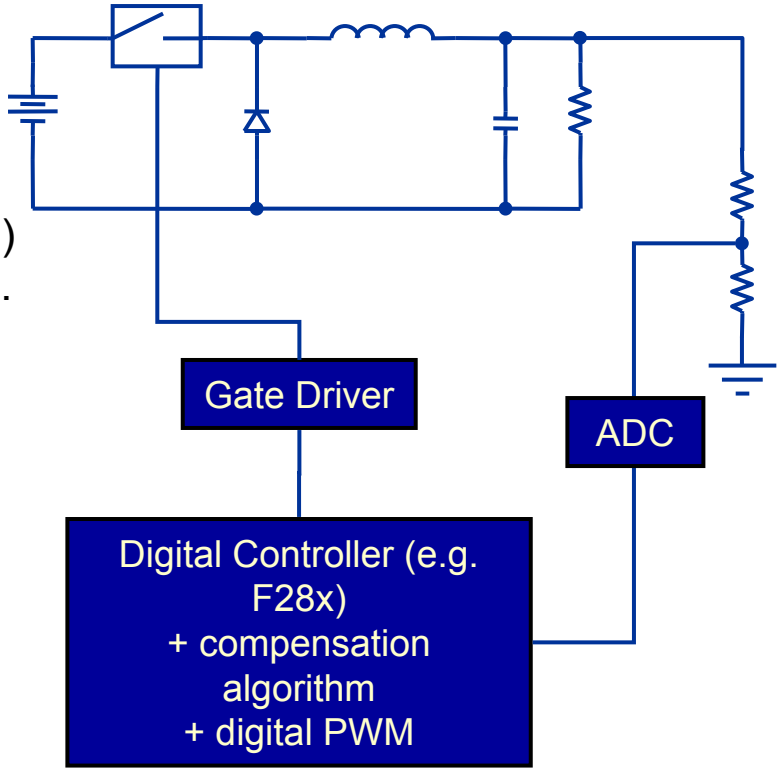
Introduction to Digital Power

- In analog world:
  - Advantages
    - High bandwidth/resolution
    - Low cost
    - Theory well defined and understood
  - Challenges
    - Component drift/aging/tolerances
    - Hardware based - not flexible
    - Limited to classical control theory only
    - No intelligent control over performance
    - Number of power stages/topologies is more limited
    - Large BoM for complex systems



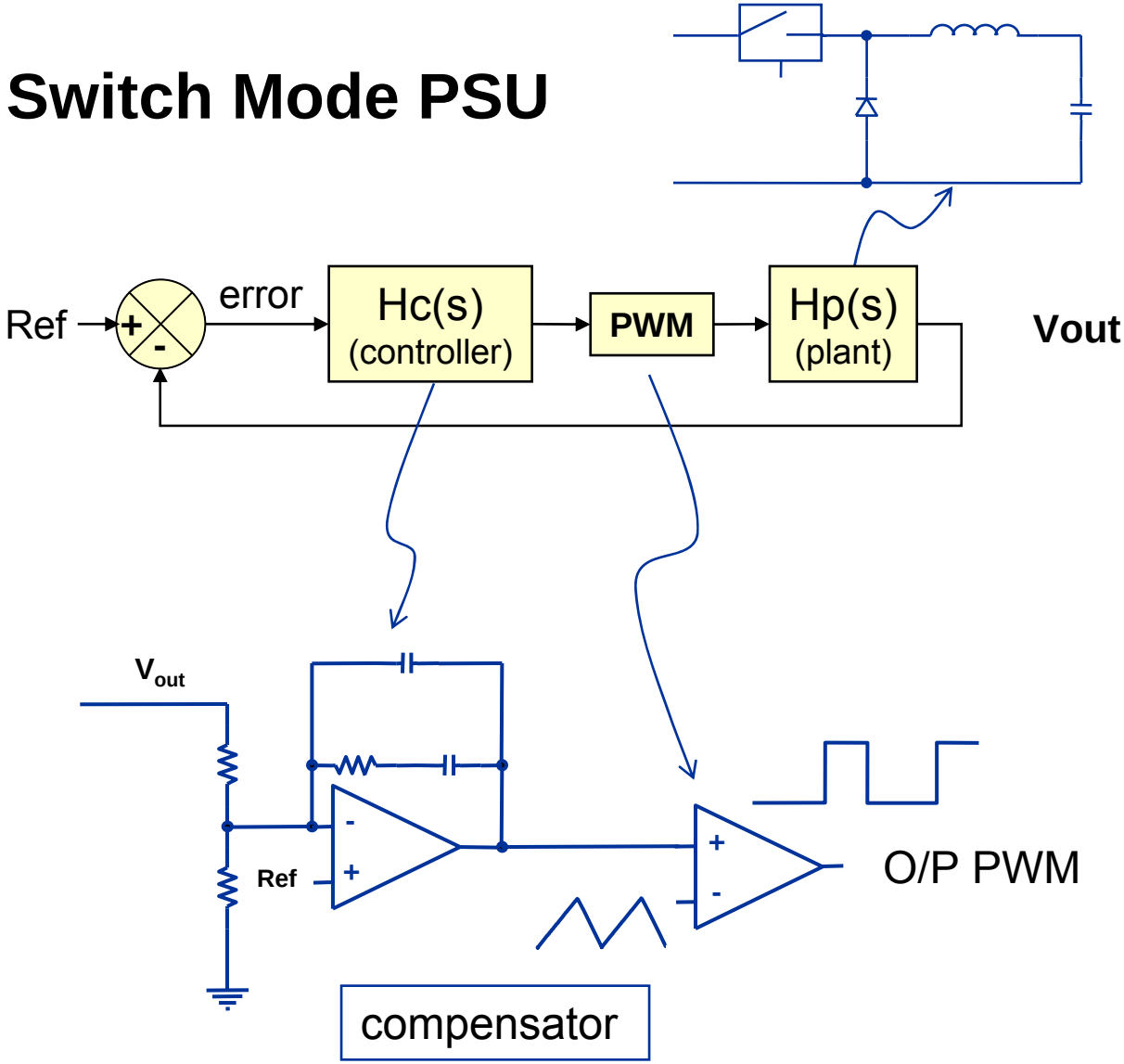
Introduction to Digital Power

- In digital world:
  - Advantages
    - Insensitive to environment (temp, drift,…)
    - S/W programmable / flexible solution (i.e. one design for multiple supplies)
    - Advanced control possible (non-linear, multi-variable)
    - Can perform multiple loops and “other” functions
    - Could reduce real estate
    - Failure prediction
  - Challenges
    - Bandwidth limitations (sampling loop)
    - PWM frequency and resolution limits
    - Numerical problems (quantization)
    - CPU performance limitations
    - Bias supplies, interface requirements
    - Theory not well understood
    - Current mode may be difficult
    - More expensive?



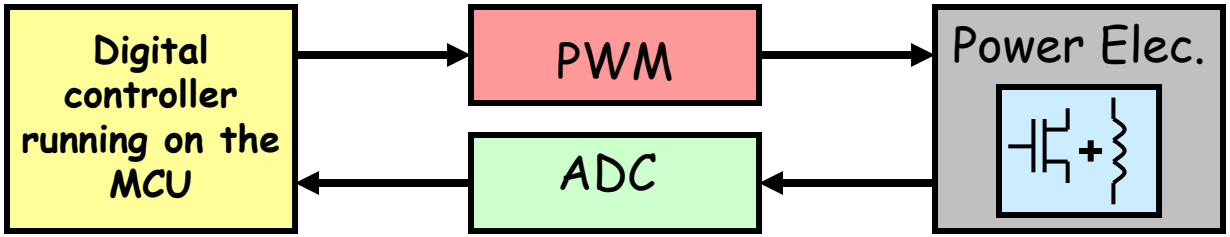
Traditional Analog Switch Mode PSU

- Typically we tune the controller (also known as the compensator) by selecting the position of poles and zeros so as to achieve the desirable gain and phase margins
- This is done through appropriate selection of the component values

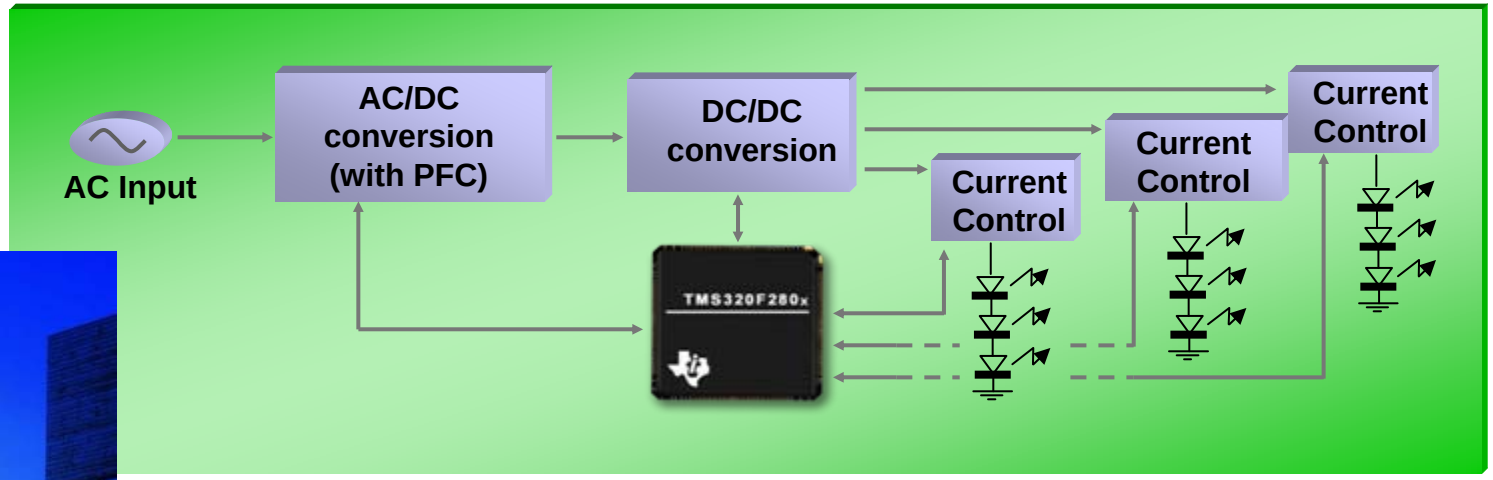


Typical Digital Digital PSU

- ADC and PWM are implemented in the MCU's H/W
- All we need to do is set up the PWM/ADC & program the z-domain control algorithm into the chip
- There is a great deal of help from TI to allow you to do this
  - Software libraries
  - Development boards
  - In-depth workshop
  - App Notes/Design guides

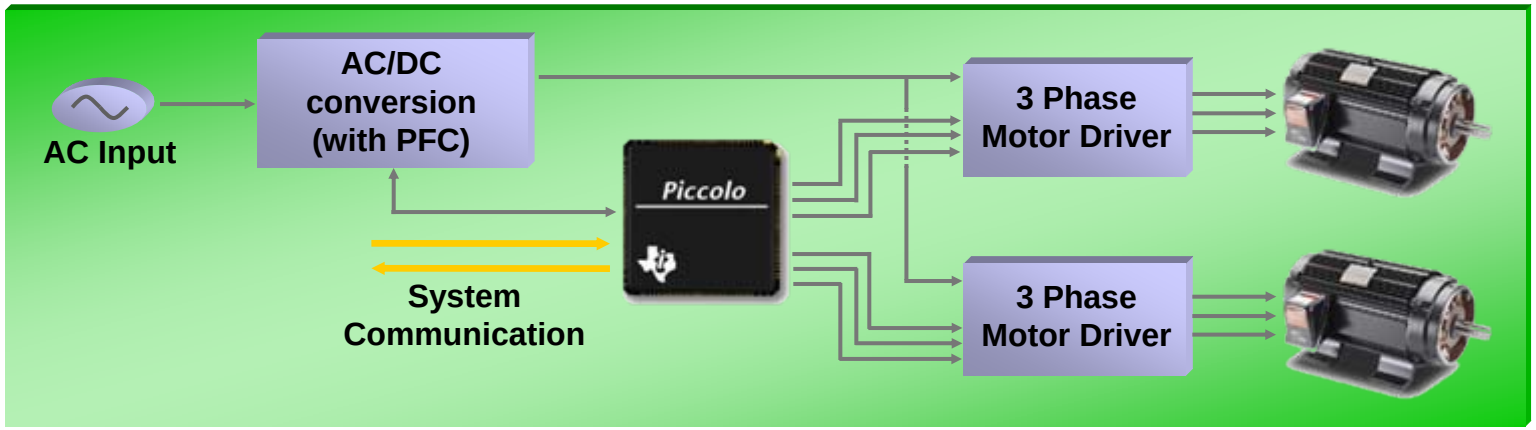


Sample Application: Commercial LED Lighting



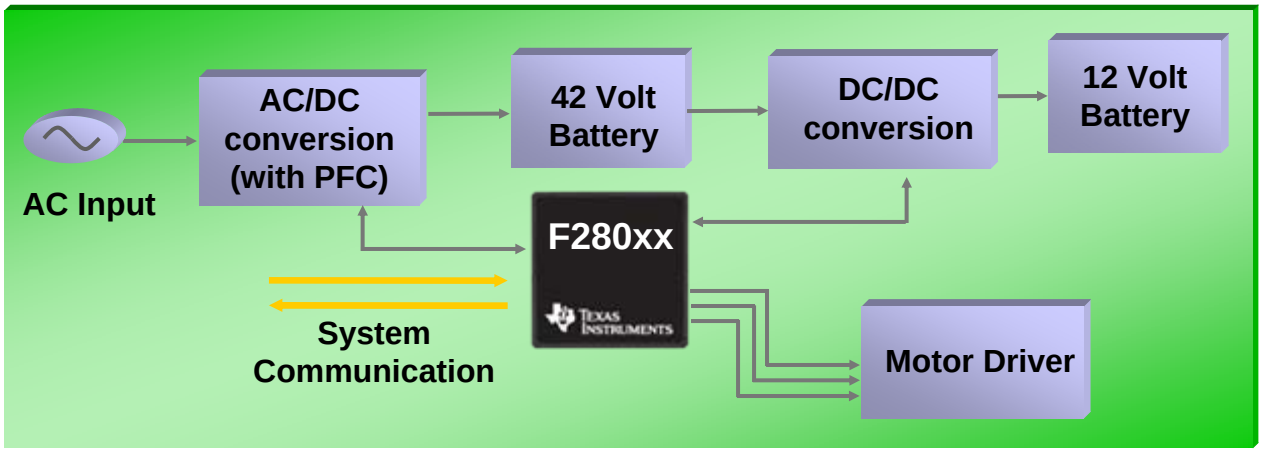
- One MCU controls both power stage and LED lighting
- Intelligent current control
  - Automatic operating state (such as blown string) detection and protection
  - Temperature monitoring for thermal runaway prevention
- Precise operating voltage and current control for precise light intensity, color mix/temp control, and increased efficiency
- Uniform platform for different LED types or configurations
- Easy system networking
  - One C2000 MCU can control LED lighting and power line communication system
  - DALI and other lighting control standards easily added
  - On chip peripherals (SPI, UART, etc) simplify interfacing with other systems

Sample Application: Appliance



- Single MCU solution, reduced system cost
  - Piccolo controls power stage and motor drivers
  - On chip oscillator and VREG reduce external components
- Advanced motor control
  - Voltage-Frequency or Field Oriented Control techniques for three phase motors
  - Piccolo has the performance to implement sensorless control
  - Advanced 3-phase motor control software libraries and training material from TI
- Digital Power
  - Digitally controlled AC/DC conversion stage with integrated PFC
- Easy system networking
  - On chip peripherals (SPI, UART, etc) simplify interfacing with other components in the system

Sample Application: Architectures for hybrid vehicles



- **Single MCU solution, reduced system cost**
  - "Piccolo" controls DC/DC conversion, AC/DC conversion, and communication
- **Digital Power**
  - Fast, accurate and sophisticated control of current and voltage for fast battery protection, long battery life, and fast charging
  - Replacement of multiple PWM and other ICs by implementing multiple control loops with one controller, to reduce system complexity and cost
  - Implementation of proper PFC and DC/DC conversion control to achieve high efficiency
- **Easy system networking**
  - Implementation of additional system control functions, such as system monitoring and supervision, with the same controller, to reduce system complexity and cost

Digital Power Challenges

- Designing digital power supplies requires an understanding of many engineering skills
  - Analog power supply design
  - Power supply stability criteria
  - Continuous time control
  - Embedded systems programming
  - Discrete time control theory

Most engineers will have studied these subjects at university years ago, but will have used only one or two of them in their jobs

Frequently Asked Questions by Engineers

- *I am an analog engineer and have not programmed since university*
  - Programming digital power supplies has a very specific template/code structure and these are available through TI. Furthermore there are numerous examples, application notes, development tools and in-depth training to start you off on your first DPS design
- *I am not confident with discrete time control theory and digital control loop design*
  - Although discrete time control theory is a vast topic, only a very small fraction of this is needed for digital power. Through our in-depth training and free on-line tools we will teach you all you need to design stable digital control loops
- *I am an embedded systems programmer and know little about switch mode power supplies*
  - Through our training programs and free software tools, we will help learn the algorithms that are needed for stable digital power supply programs

How Does TI Help Engineers in Their DPS Designs ?

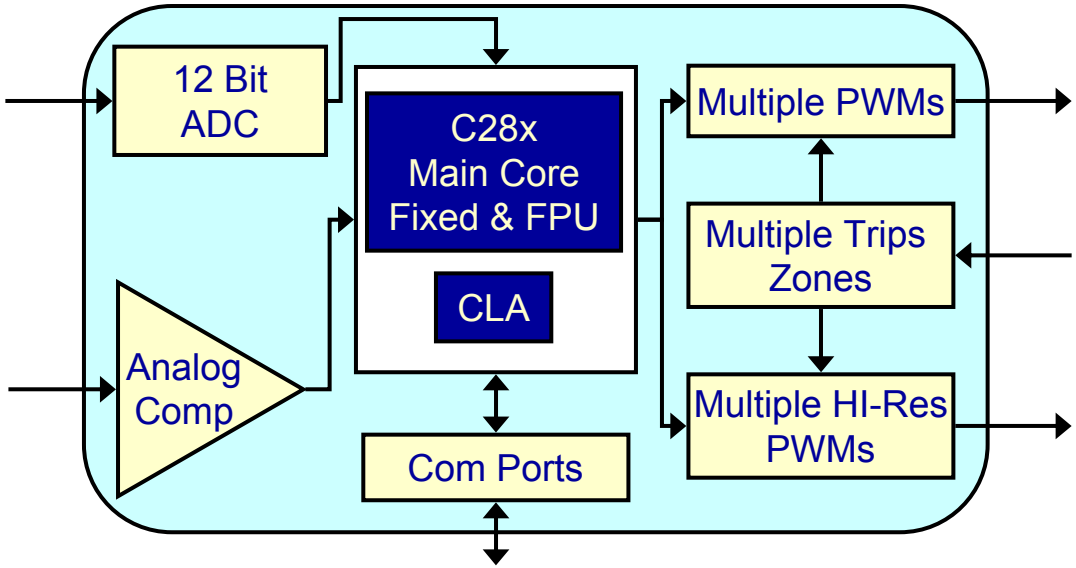
- Extensive portfolio of MCUs dedicated to digital power
  - Fixed point/floating point/dual core
  - Dedicated peripherals
  - Hi-resolution PWMs and Fast ADCs
- Extensive Hardware Development Tools
  - From simple Buck to PFC to Phase Shifted Full Bridge
- Extensive software libraries dedicated to digital power
  - TI digital power libraries
  - Biricha Digital's Chip Support Library (CSL)
- In-depth Training
  - Biricha Digital's 3 day hands-on digital power design workshops

Texas Instruments' TMS320C2000™ Family

- Digital power, Industrial, and motor control applications
- Dedicated peripherals & functionality
  - ADC
  - PWM/Hi-Res PWM
  - Analog Comparators
  - Fast Trip Zones
  - Fast interrupts
- Fixed and floating point
- Excellent dev tools and libraries

TI's Piccolo Parts Dedicated to Digital Power

- Significantly Lower System Cost
  - Single Supply – 3.3V only
  - On-Chip Reset
  - Lower Power Consumption
  - Minimal Support Pins
  - Small Packages
  - 2 x On-Chip Oscillators
- Increased CPU Performance
  - Control Law Accelerator
- Enhanced PWM Capabilities
  - High resolution PWM period → resonant converters
  - Enhanced PWM triggering → peak current mode
- Improved Analog
  - Ratio-metric ADC
  - Improved triggering
  - Analog comparators



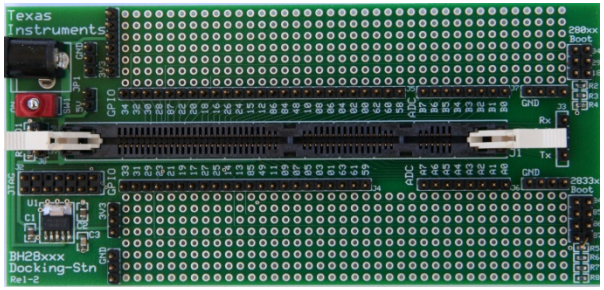
Industry's Most Extensive set of Digital Power Hardware Development Tools

- A multitude of DPS boards are now available based on a range of versatile control cards

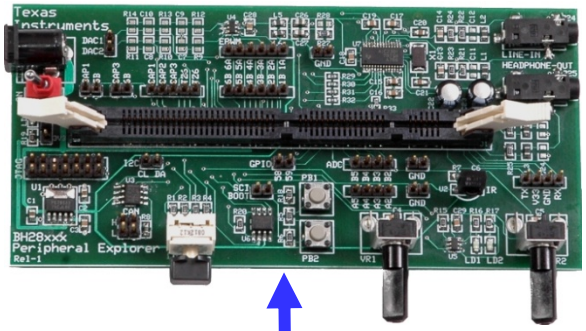
DIM100 controlCARDs - pin compatible regardless of C2000 MCU

	ControlCARD	Part Number	Descri
	"Piccolo" F28027	TMDSCNCD28027	F2802
	"Piccolo" F28035	TMDSCNCD28035	F2803
	"Piccolo" F28069	TMDXCNCD28069	F2806
	F28044	TMDSCNCD28044	F2804
	F2808	TMDSCNCD2808	F2808
	"Delfino" F28335	TMDSCNCD28335	F2833
	"Delfino" C28343	TMDSCNCD28343	C28343

Docking Station / prototyper



Peripheral Explorer

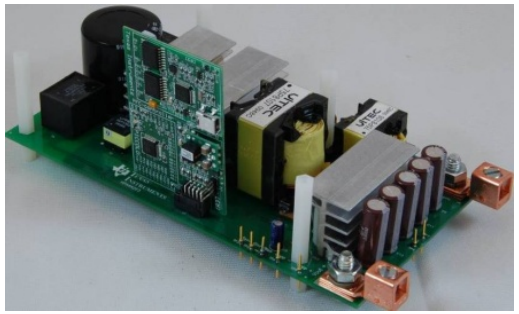


PFC – 2phsIL

controlCARD

Multi-Rail/Phase DCDC

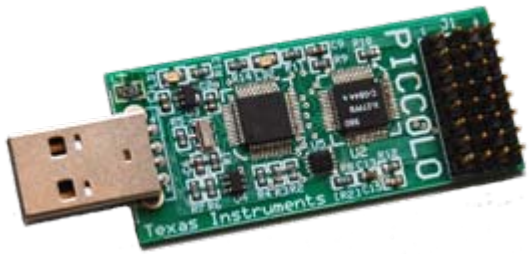
Resonant LLC+ SR

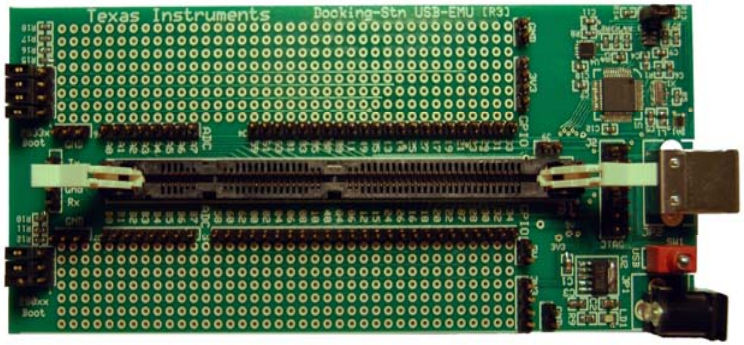


PSFB+SR + Peak Current Mode Control

PFC – BridgeLess

Generic Low Cost Development Hardware





Piccolo controlSTICK - $39

- On board USB JTAG emulation
- Small, USB stick form factor
- Access to all control peripherals through header pins
- Example projects show how to use Piccolo's features
- Ultra low-cost
- Designed to be an evaluation tool to teach new users about Piccolo and how to use the peripherals such as PWM, ADC, and setup a Piccolo project

Piccolo USB Experimenter's Kit - $79

- On board USB JTAG emulation
  - No external emulator or power supply needed
  - Has connection for external emulator and power supply
- Comes with Piccolo controlCARD
- Access to all Piccolo pins
- Prototyping Area
- Low cost
- Designed to be a development tool for users who want to start developing with Piccolo
- *controlCARD only - $49*

And many many more…. Please visit www.ti.com

Software Dev Tools and Libraries

- Code Composer Studio (CCS)

C/C++ - main.c - Code Composer Studio (Licensed)

File Edit View Navigate Project Target Tools Scripts Window Help

C/C++ Projects | Target Configu...

- F2808_Template [Active - Debug]
  - Binaries
  - Includes
  - Debug
  - main.c
  - DSP280x_Headers_nonBIOS.cmd
  - F2808_Template.pjt
  - csl280x_ml.lib
  - csl_2808_RAM_lnk.cmd
  - macros.ini

main.c

```
12      SYS_init();
13      ADC_init();
14      GPIO_config( GPIO_34, GPIO_DIR_OUT, false );
15
16
17
18  while (1)
19  {
20      for ( i = 0 ; i < 360; i++ )
21      {
22          a = sin(i * (pi/180.0) );
23          SYS_msDelay(1000);
24          GPIO_tog( GPIO_34 );
25
26      }
27  }
28  }
```

Console

C-Build [F2808_Template]

```
"C:/tidcs/c28/DSP280x/v170/DSP280x_headers/
md/DSP280x_Headers_nonBIOS.cmd"
"C:/tidcs/c28/CSL_C280x/v100/common/lib/csl
80x_ml.lib"
<Linking>
'Finished building target:
F2808_Template.out'
' '
Build complete for project F2808_Template
```

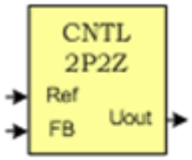
Problems

0 errors, 0 warnings, 0 infos

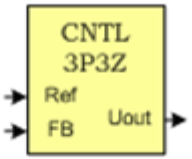
Description	Resource	Pa

Writable | Smart Insert | 28 : 2

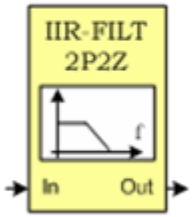
Free Open Source Libraries Provided by TI



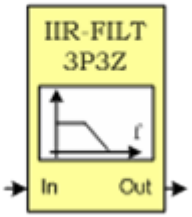
Control 2-pole / 2-zero



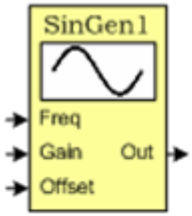
Control 3-pole / 3-zero



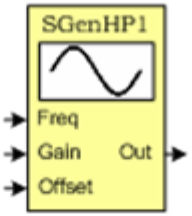
2nd order IIR filter



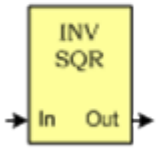
3rd order IIR filter



Sine Wave generator

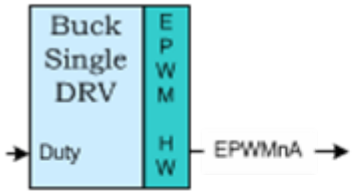


High precision Sine Gen

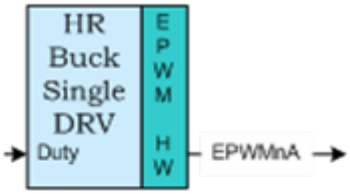


Inverse Square function

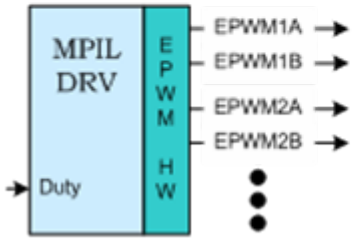
Soft Start and Sequencing



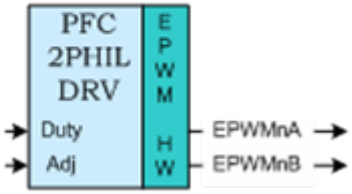
Buck Single Output



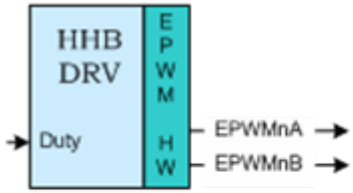
High Resolution Buck



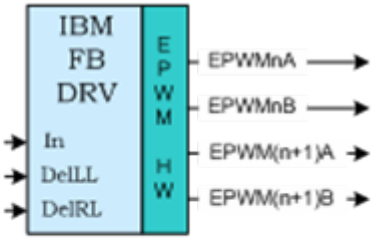
Multi-Phase Interleaved



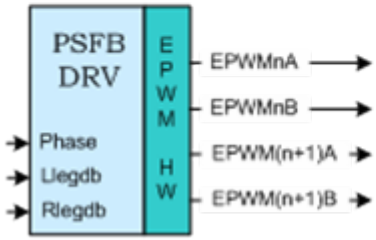
Power Factor 2-phase Interleaved



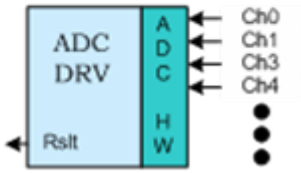
Half H-Bridge



IBM method Full Bridge



Phase Shifted Full Bridge



Analog-Digital Converter driver

Propriety APIs Dedicated to Digital Power

- Biricha Digital's Chip Support Library (CSL)
  - Highly optimised code
    - Takes the pain out of programming
    - Hardware Abstraction Layer stops you from having to deal with registers directly
  - Aimed at digital power supply designers
    - Analog engineers – Programming made easy!
    - Embedded systems engineers
  - Numerous functions:
    - Dedicated to digital power
    - Quick chip setup
    - Quick ADC/PWM
    - Digital power control algorithms

Example 1: Dedicated Digital Voltage Mode Controller Functions with the CSL

```
#define K   (.78)
#define A1  (+1.46818)
#define A2  (-0.314933)
#define A3  (-0.153248)
#define B0  (1.784224053)
#define B1  (-1.629063952)
#define B2  (-1.780916725)
#define B3  (1.632371281)

CNTRL_3p3zInit(&my_piccolo_3p3z
       ,_IQ15(REF)
       ,_IQ26(A1),_IQ26(A2),_IQ26(A3)
       ,_IQ26(B0),_IQ26(B1),_IQ26(B2),_IQ26(B3)
       ,_IQ23(K),MIN_DUTY,MAX_DUTY );
```

Example 2: CSL Simplifies Creating a 100kHz PWM

```
EPwm1Regs.TBCTL.bit.SYNCOSEL = TB_SYNC_IN;
EPwm1Regs.TBCTL.bit.PHSEN = TB_ENABLE;
EPwm1Regs.TBPHS.half.TBPHS = 100;
EPwm1Regs.TBPRD = PWM1_TIMER_TBPRD;
EPwm1Regs.TBCTL.bit.CTRMODE = TB_COUNT_UP;
EPwm1Regs.ETSEL.bit.INTSEL = ET_CTR_ZERO;
EPwm1Regs.ETSEL.bit.INTEN = PWM1_INT_ENABLE;
EPwm1Regs.ETPS.bit.INTPRD = ET_1ST;
```

```
PWM_config( PWM_MOD_1, PWM_freqToTicks(100000), PWM_COUNT_UP );
```

This software bundle is free to all attendees of TI's 3 Day Digital Power Training Workshop
We will talk about training next

Digital Power Design Workshop

- Designing digital power supplies requires an understanding of many engineering skills
  - Analog power supply design
  - Power supply stability criteria
  - Embedded systems programming
  - Continuous and discrete time control theory
- Grasping all of the above by a design engineer is time consuming and expensive
- Therefore TI is providing Engineers with in-depth training

In order reduce your development time and learning curve TI has developed am in-depth hands-on engineering workshop

Abridged Workshop Syllabus

- **Day 1: Introduction to Digital Power & Programming**
  - The C2000 family's development tools & features
  - Use Biricha Digital's libraries to run MCU code with minimal programming
  - Programming the C2000 for digital power (Interrupts, Templates, PWMs etc)

- **Day 2: Voltage Mode PSU Design**
  - Step by step design of digital power supplies
  - Stable Analogue and digital power supply Design
  - Discrete time control theory, Z transforms & digital convolution

- **Day 3: Digital Peak Current Mode**
  - Piccolo B and the CLA
  - Analogue and Digital peak current mode control
  - Running multiple power supplies

Designing Our First Digital Power Supply

- There is plenty of information on:
  - TI's portfolio of dedicated digital power MCUs
  - Hardware development tools available
  - Software libraries provided to help

- All we need is to learn how to design digital power supply
  - During the 3 day workshop we assume no previous knowledge and teach analogue and digital power supply control loop design in detail
  - For this 1 hour session there is little time so we start with a pre designed analogue power supply and concentrate on digital design only

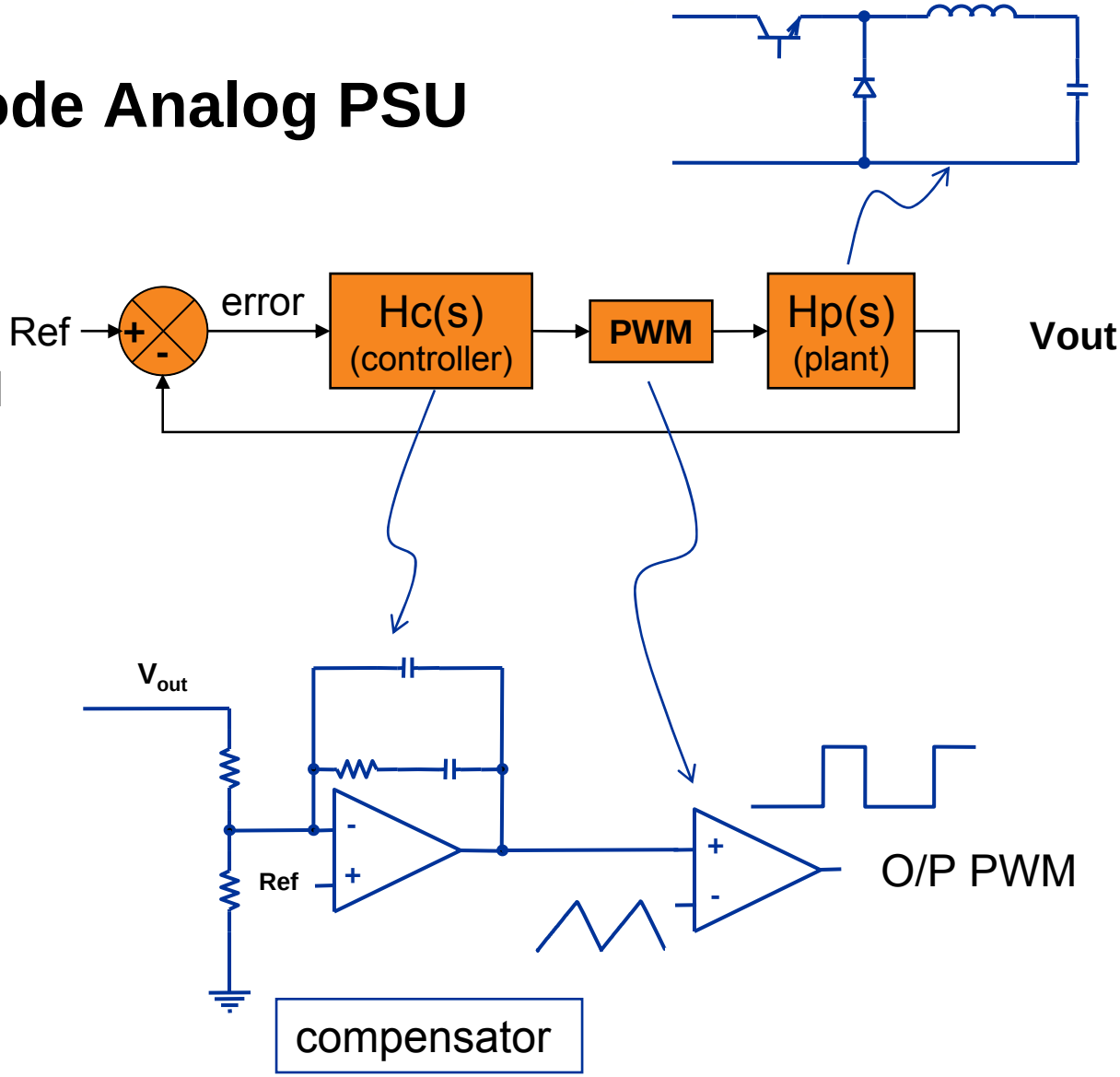
Analog Compensator Design – A Short Review

- Analog PSUs are (almost) always designed in the frequency domain
  - We superimpose a sinusoid of a certain frequency (say 10 Hz) on our PWM and we measure how the gain and phase of this sinusoid is modified by the time it goes through our plant (i.e. PSU)
  - We increase the frequency of our injected sinusoid and measure again, we repeat this for all frequencies of interest (say 10Hz to ½ Fs) and plot the Bode plot

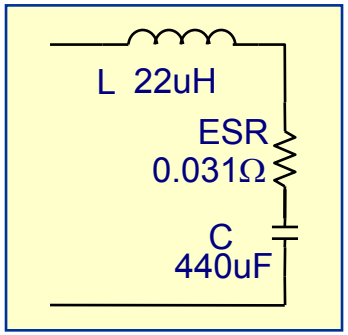
In short we plot the "open loop" gain and phase of the PSU (i.e. its Bode plot) and design the compensator such that we get appropriate gain and phase margins

Typical Voltage Mode Analog PSU

- Typically we tune the compensator by selecting the position of poles and zeros so as to achieve the desirable gain and phase margins
- To do this we need the Transfer Functions Hc(s) & Hp(s) and we can then plot the Bode plot and calculate the component values



Bode Plot of the Buck Converter Used in the Lab



$$f_r = \frac{1}{2\pi\sqrt{LC}} = 1617\ Hz$$

Double pole

PWM Gain = 21.58dB

$$f_{esr} = \frac{1}{2\pi\ ESR\ C} = 11668.25\ Hz$$

ESR Zero

High frequency phase: 2 poles and 1 zero (-180° + 90°) = -90°

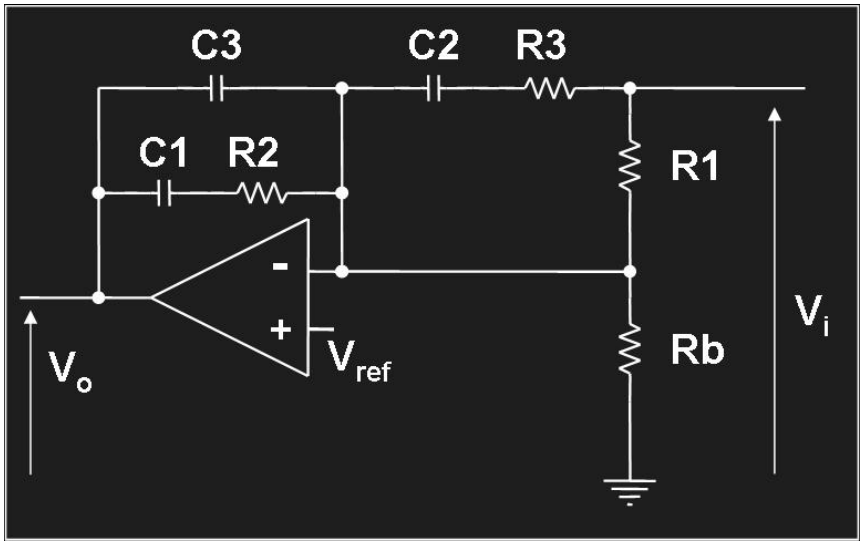
Gain (dB) & Phase (degrees)

100
0
-100
-200

10Hz
100Hz
1.0KHz
10KHz
100KHz
1.0MHz

Frequency (Hz)

Type III Compensator used to Stabilize our Buck Converter



- Type III:
  - 3 poles & 2 zeros
  - Used for voltage mode buck
  - Transfer function:

$$H_c(s) = \frac{\omega_{p0}}{s} \frac{\left(\frac{s}{\omega_{z1}}+1\right)\left(\frac{s}{\omega_{z2}}+1\right)}{\left(\frac{s}{\omega_{p2}}+1\right)\left(\frac{s}{\omega_{p3}}+1\right)}$$

  - Where:

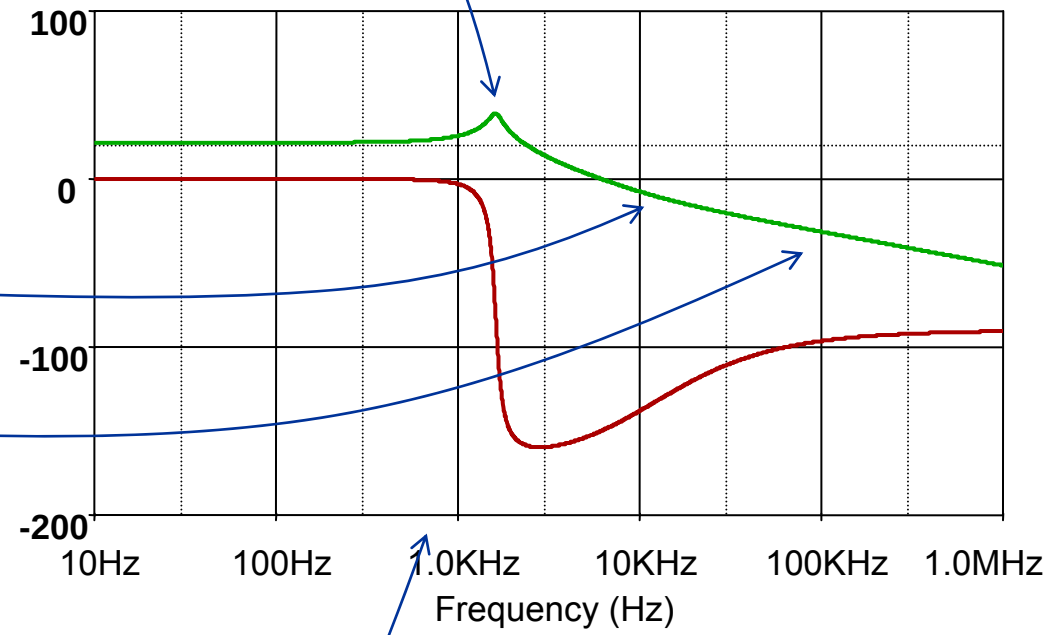
$$\omega_{z1} = \frac{1}{R_2\, C_1};\ \omega_{z2} = \frac{1}{C_2\,(R_1+R_3)};\ \omega_{p0} = \frac{1}{R_1\,(C_1+C_3)};\ \omega_{p2} = \frac{(C_1+C_3)}{R_2\, C_1\, C_3};\ \omega_{p3} = \frac{1}{R_3\, C_2}$$

Analog Compensator Design Example

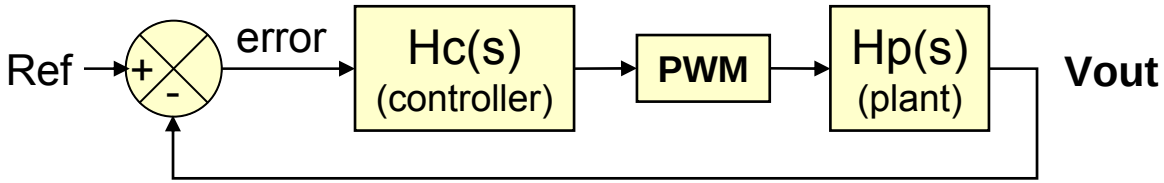
- Type III Controller (simple approximate method)
  - Place two zeros at F_r
    - *i.e. @ 1617Hz*
  - Place one pole at F_{esr}
    - *i.e. 11668.25 Hz*
  - Place second pole at $F_s/2$
    - *i.e. 100 kHz*
  - Place Pole at Origin at:

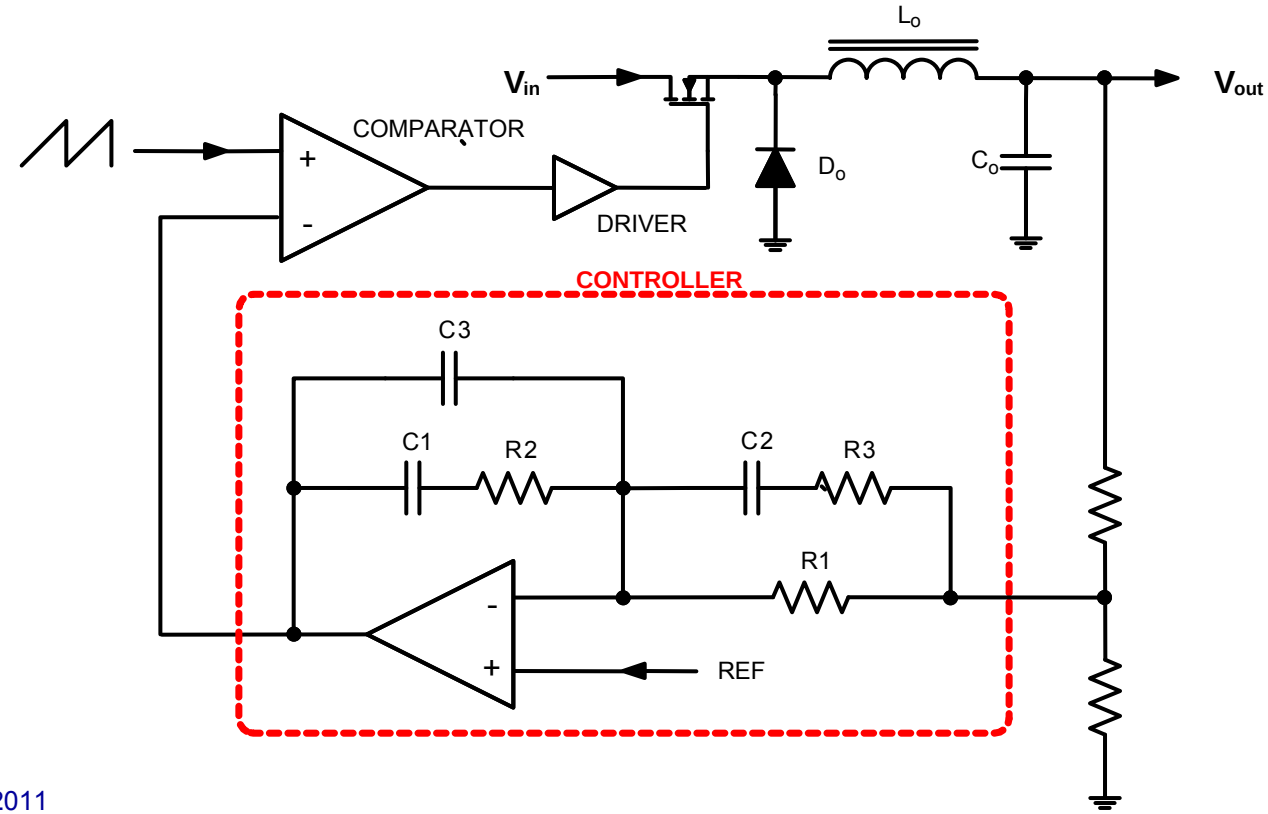
$$f_{p0} = \frac{V_{ramp} \times F_x}{V_{in}}$$

  - i.e. for a V_{ramp} of 1V, V_{in} = 12 and F_x of 10kHz → $f_{p0} = 833Hz$



Example – Our BDP-106 Buck Converter in Analog World

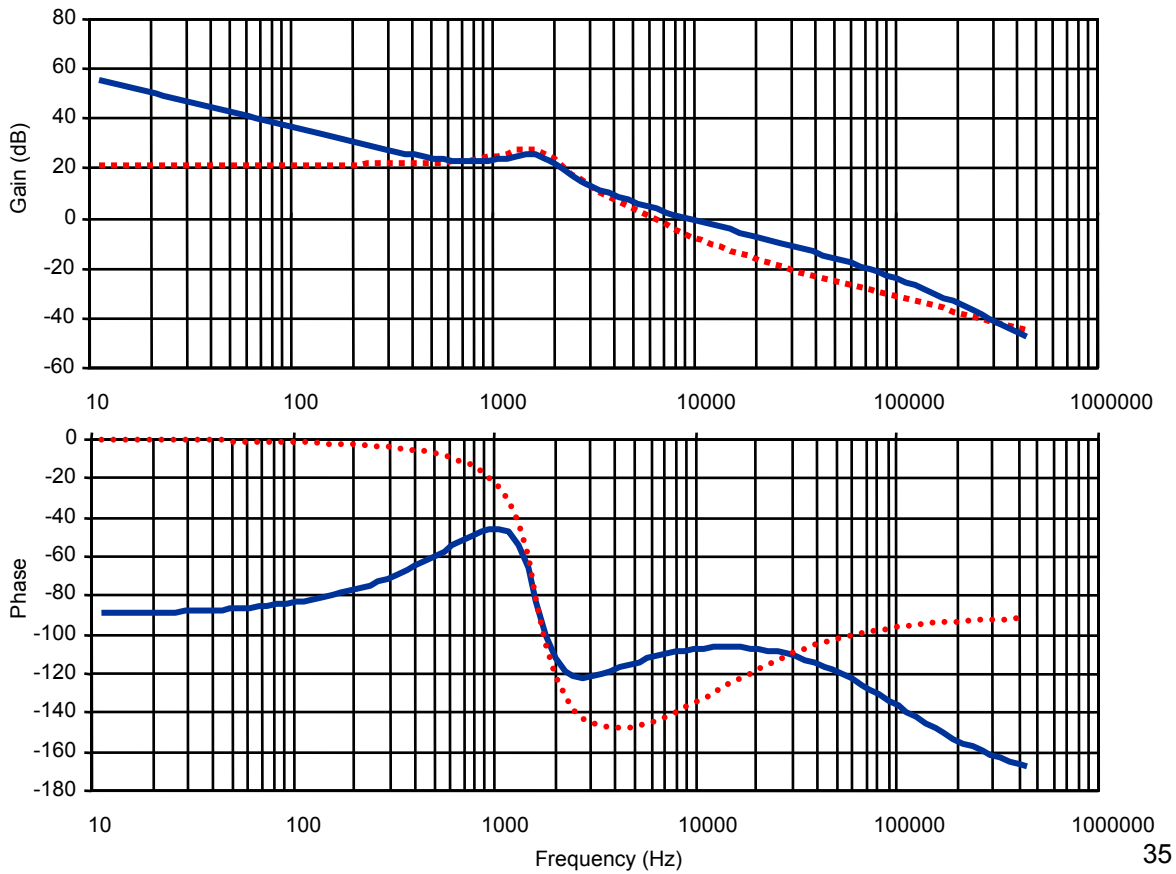




Example – Our BDP-106 Buck Converter in Analog World

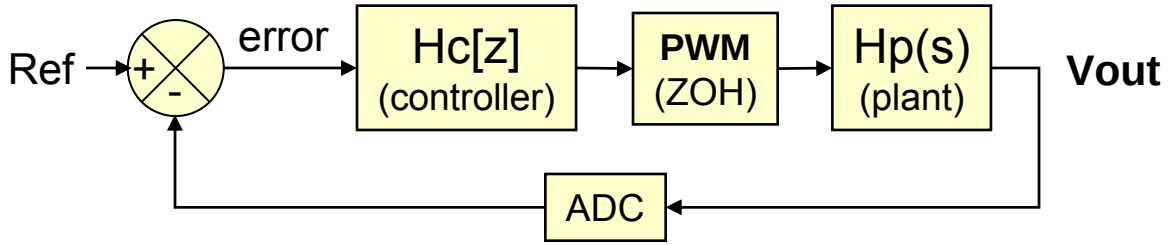
- Red (dotted) Trace → Original power stage without compensation
- Blue (solid) Trace → Open loop gain after compensation
- Using the transfer function and component values our analog type III compensator poles and zeros are selected such that:
  - Very high gain at DC, i.e. pole at origin
  - Fx = 10kHz as desired
  - $\phi_M \cong 75°$
  - Slope of gain plot at Fx = 20dB/decade
  - G_M better than 40dB

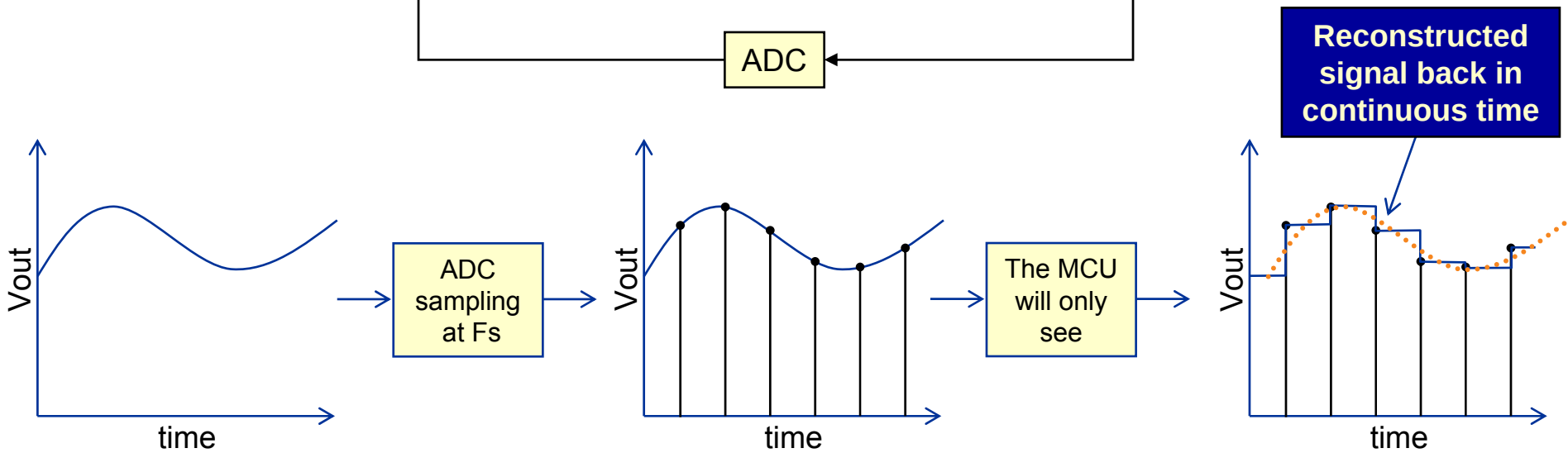
∴This power supply is stable → All we need to do is to convert our Hc(s) from Analog (continuous time) to digital (discrete time)



Converting our Analog Design in to Digital Domain

- The design of an analog PSU is carried out in “continuous time”
- A digital power supply is a “discrete time” system



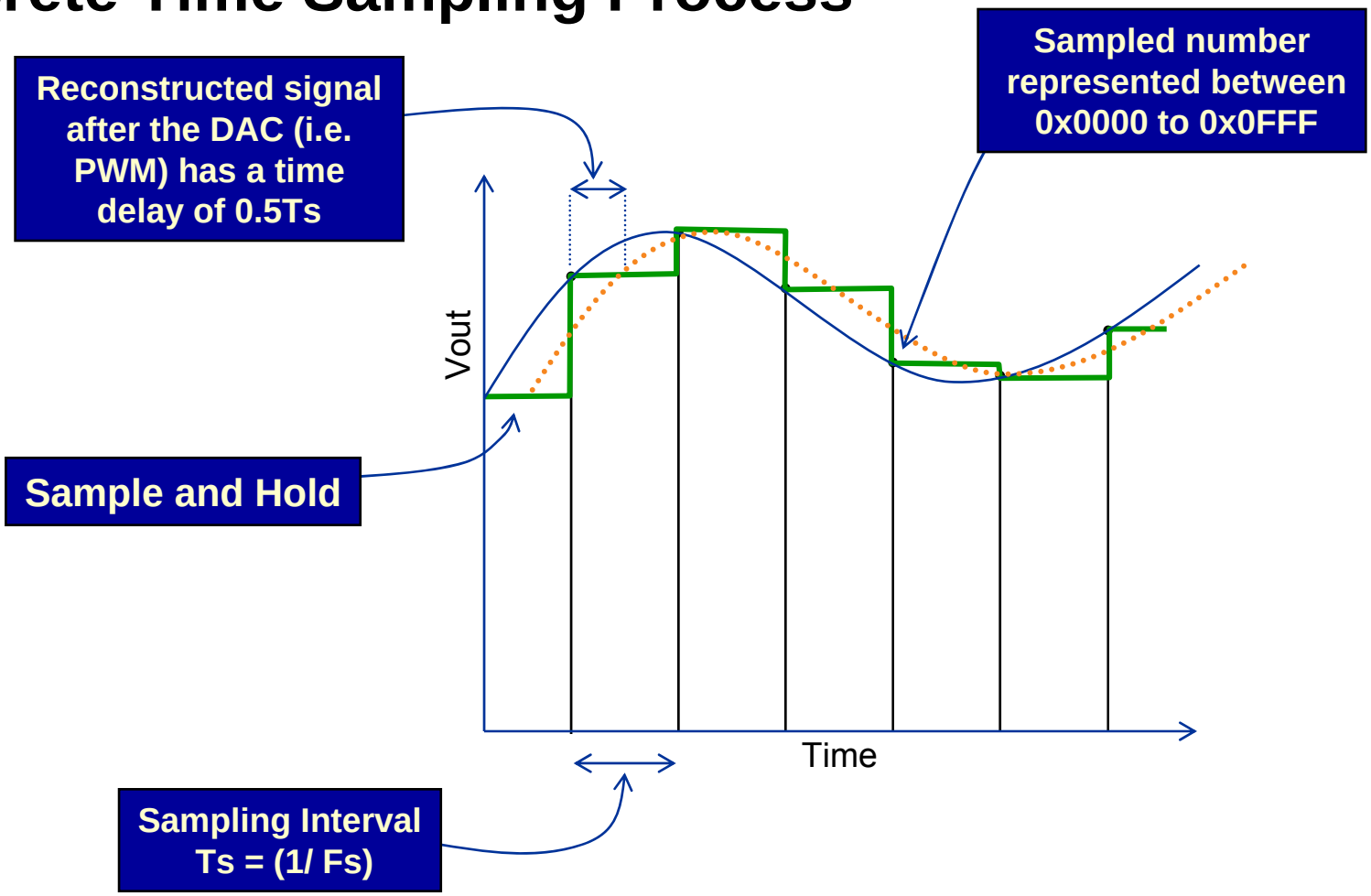


Actual Vout in continuous time

Sampling at Fs we end up with a “discrete” number of samples i.e. discrete time

The MCU will only have the value of Vout when we sampled it and has to wait until the next sample for an update

Discrete Time Sampling Process

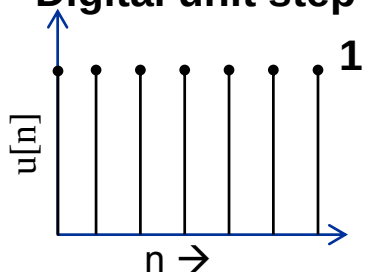


Now that we have converted our analog signal into discrete domain we need to be able to manipulate it

Representing Discrete Time Signals

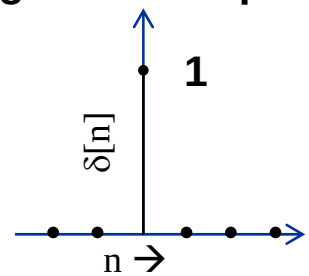
- Unit Step:
  - $u[n] = 0, n < 0$
  - $u[n] = 1, n \geq 0$

Digital unit step



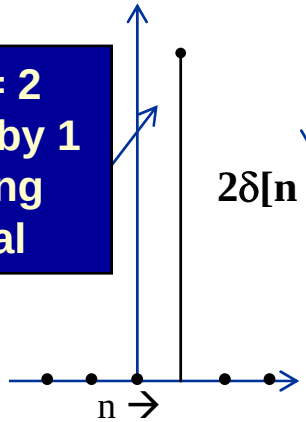
- Unit Impulse
  - $\delta[n] = 0, n \neq 0$
  - $\delta[n] = 1, n = 0$

Digital unit impulse



Important: This is how we represent "delay" in the digital world

Size = 2
Delayed by 1 sampling interval



$2\delta[n - 1]$

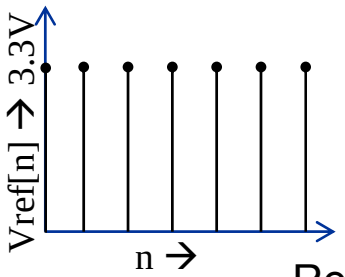
- Scaling and delaying
  - The digital signal can be scaled
  - And it can be delayed in time

Digital PSU:

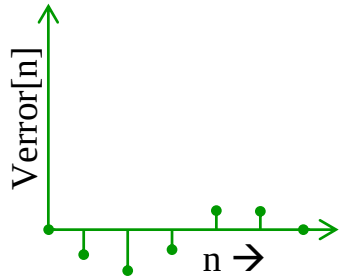
Magic box with a digital transfer function that will take the error, manipulate it (e.g. using a digital type III controller) and output a discrete number between 0 and 100 representing 0% to 100% duty

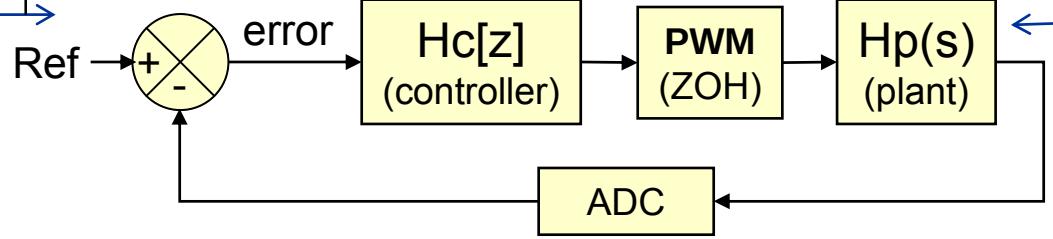
Takes 0 – 100 from previous block and creates a PWM between 0% and 100% duty

Our Input

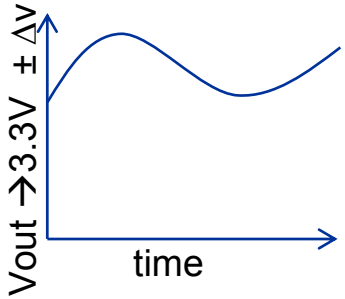


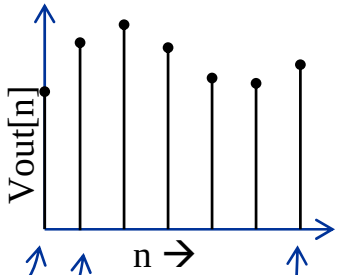
Vref[n] – Vout[n]





Analog Plant; i.e. the power stage





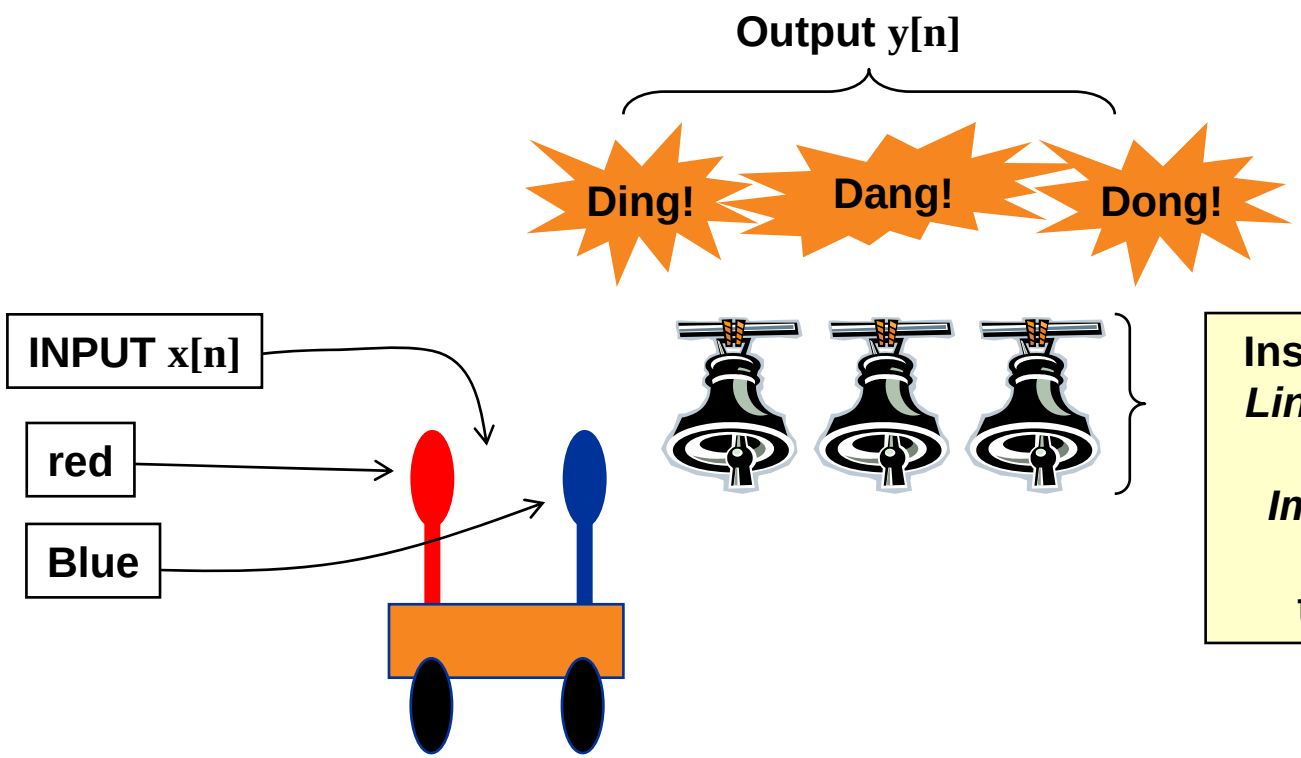
n = 0 → Vout[0]
n = 1 → Vout[1]
…
n = 6 → Vout[6]

Important: x axis now represents "sample number" [n] meaning one sampling interval

Inside the Magic Box

- The digital controller/compensator takes the sampled error signal and manipulates it
- Just like an analog circuit, it can be described with a digital equivalent of a "differential equation"
  - In digital world this is called the "linear difference equation"
  - We need the linear difference equation because it is the algorithm that the MCU needs to run in order to manipulate/process the error signal
- Just like an analog circuit it has a transfer function
  - This is called the "z" transfer function – we will talk about this later
  - We need this transfer function so that we can convert our tried and tested analog transfer function in the "*s*" domain (e.g. Type III's H(s)) into the digital world

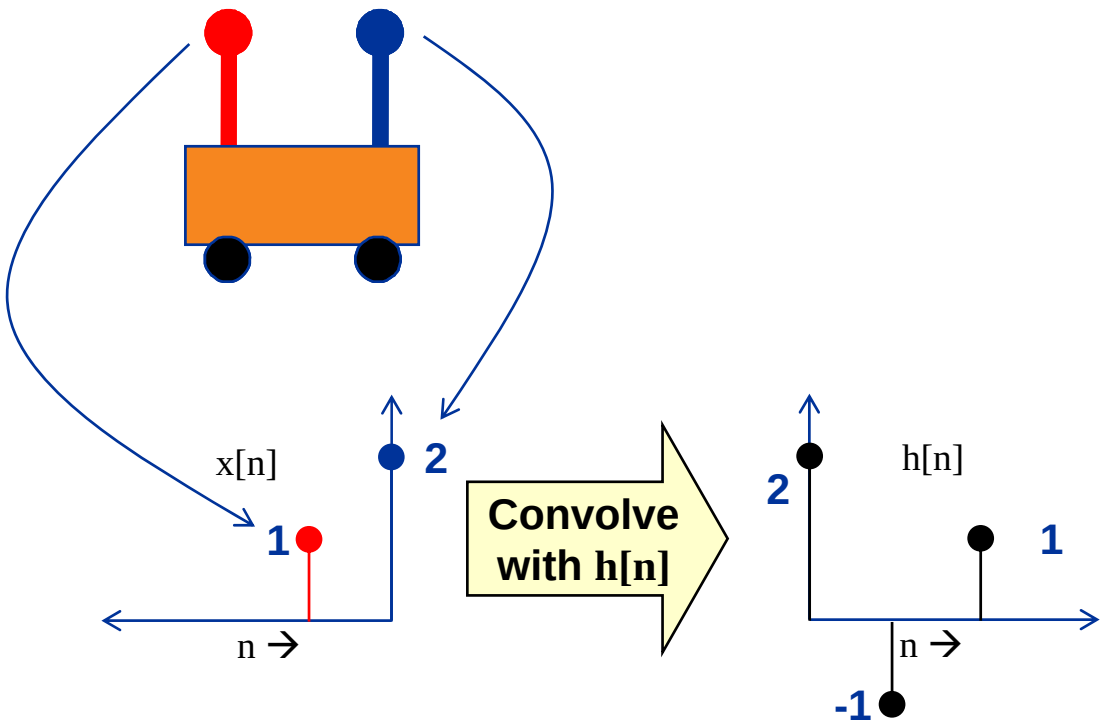
Digital Convolution: What happens inside the Magic Box



Inside the magic box is the *Linear Difference Equation* or the *Impulse Response* Hc[n]
Later we will convert to "z" domain→ Hc[z]

@ 1st sampling interval: output → (Blue x Ding) + (Red x 0)
@ 2nd sampling interval: output → (Blue x Dang) + (Red x Ding)
@ 3rd sampling interval: output → (Blue x Dong) + (Red x Dang)
@ 4th sampling interval: output → (Blue x 0) + (Red x Dong)

Digital Convolution: What happens inside the Magic Box



@ 1st sampling interval: output y[0] → (2 x 2) + (1 x 0) = 4

@ 2nd sampling interval: output y[1] → (2 x -1) + (1 x 2) = 0

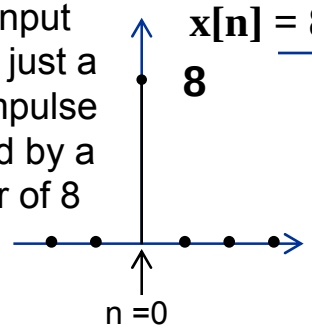
@ 3rd sampling interval: output y[2] → (2 x 1) + (1 x -1) = 1

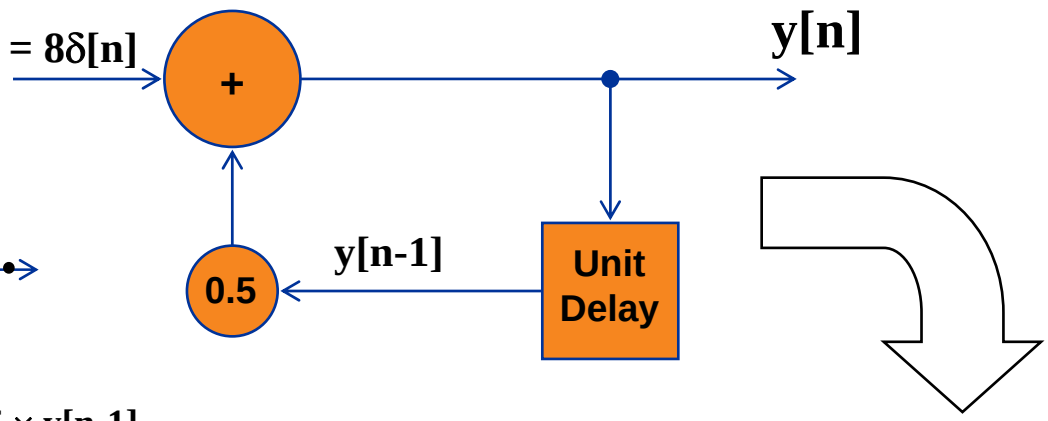
@ 4th sampling interval: output y[3] → (2 x 0) + (1 x 1) = 1

Inside the magic box Hc[n]; later we will convert to "z" domain→ Hc[z]

Inside the Magic Box

Lets assume that our input x[n] is just a unit impulse Scaled by a factor of 8





$$y[n] = x[n] + 0.5\, y[n-1]$$

Lets evaluate: y[n] = x[n] + 0.5 × y[n-1]

@ n=0:

[0 – 1]

y[0] = x[0] + (0.5 × y[-1])

y[0] = 8 + (0.5 × 0) → y[0] = 8

@ n = 1:

y[1] = x[1] + (0.5 × y[0])

y[1] = 0 + (0.5 × 8) → y[1] = 4

@ n = 2:

y[2] = x[2] + (0.5 × y[1])

y[2] = 0 + (0.5 × 4) → y[2] = 2

@ n = 3:

y[3] = x[3] + (0.5 × y[2])

y[3] = 0 + (0.5 × 2) → y[3] = 1

- **Linear difference equation:**
  - y[n-1] means the previous value of output "y[n]" or in other words it means "y[n] delayed by one sampling interval"
  - Because the output y[n] depends on the previous value of itself, this is a "recursive" linear difference equation
  - The output in this case in an exponential decay and therefore the system is stable

Converting Linear Difference Equations into z Transfer Functions

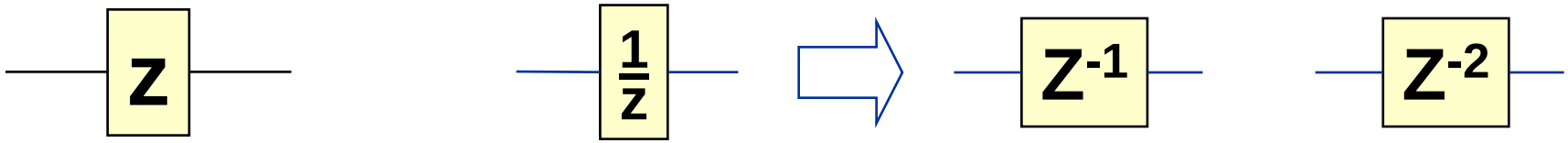
- As you can imagine a computer excels at calculating the recursive linear difference equation on the previous slide
- We can very easily implement any equation of type: $y[n] = x[n] + y[n-1]$ …. on our MCU using C or assembly
- The problem is that a linear difference equation does not readily translate into an "analog" transfer function
- We need a way of converting our analog controller's H(s) into a linear difference equation in order to calculate it using our MCU
  - e.g. We need to "convert" Hc(s) for our type III controller into a linear difference equation

$$H_c(s) = \frac{R_1 + R_3}{R_1 R_3 C_3} \frac{\left(s + \frac{1}{R_2 C_1}\right)\left(s + \frac{1}{(R_1 + R_3)C_2}\right)}{s\left(s + \frac{C_3 + C_1}{R_2 C_1 C_3}\right)\left(s + \frac{1}{R_3 C_2}\right)}$$

y[n] = …..?

z Transforms

- z Transforms allow analog transfer functions to be converted into the digital domain
- Similar to Laplace in the analog world, they make the job of manipulating transfer functions very easy
  - Very convenient and compact
  - Like the s domain, we deal with poles and zeros
- We define the "z" operator as a unit advance in time



Advance in time by 1 sampling interval

Unit delay (i.e. previous value) i.e. delay by one sampling interval → e.g. y[n -1]

Unit delay (previous value) e.g. y[n -1]

2 unit delays e.g. y[n -2]

Linear Difference Equation to H[z] Conversion

- Consider the following linear difference equation:

$$y[n] = 4\,x[n] + 3\,x[n-1] + 2\,y[n-1]$$

Substituting for the coefficients of y[n] and x[n]:

$B_0 = 4$, $B_1 = 3$ and $A_1 = 2$

$$y[n] = B_0\,x[n] + B_1\,x[n-1] + A_1\,y[n-1]$$

Unit delay (previous value)

- Transforming onto z domain:

$$y[z] = B_0\,x[z] + B_1\,x[z]\,z^{-1} + A_1\,y[z]\,z^{-1}$$

- Finally:

$$H[z] = \frac{y[z]}{x[z]} = \frac{B_1\,z^{-1} + B_0}{-A_1\,z^{-1} + 1}$$

Where:
$x[z]$ is the input; in our case the error signal
And $y[z]$ is the output of the controller; in our case new value of duty

z Transfer Function H[z]

- z transfer functions allow us to analyze the behavior of our digital system (just like Laplace)
- There is a direct link between H(s) and H[z]
  - So we can easily convert our analog controller in the s domain to an equivalent digital controller in the z domain
- Starting from a z transfer function H[z] we can very easily get back to the linear difference equation

$$H[z] = \frac{y[z]}{x[z]} = \frac{B_1 z^{-1} + B_0}{-A_1 z^{-1} + 1} \rightarrow \mathbf{y[n] = B_0\, x[n] + B_1\, x[n-1] + A_1\, y[n-1]}$$

- We can then get our MCU to calculate the linear difference equation during every cycle

H[z] to Linear Difference Equation Conversion

- Convert the following z transfer function to a linear difference equation:

$$H[z] = \frac{y[z]}{x[z]} = \frac{B_1\, z^{-1} + B_0}{-A_1\, z^{-1} + 1}$$

$$y[z]\left(-A_1\, z^{-1} + 1\right) = x[z]\left(B_1\, z^{-1} + B_0\right)$$

$$y[z] = B_0\, x[z] + B_1\, x[z]\, z^{-1} + A_1\, y[z]\, z^{-1}$$

$$\mathbf{y[n] = B_0\, x[n] + B_1\, x[n\text{-}1] + A_1\, y[n-1]}$$

Steps for Designing a Digital PSU controller

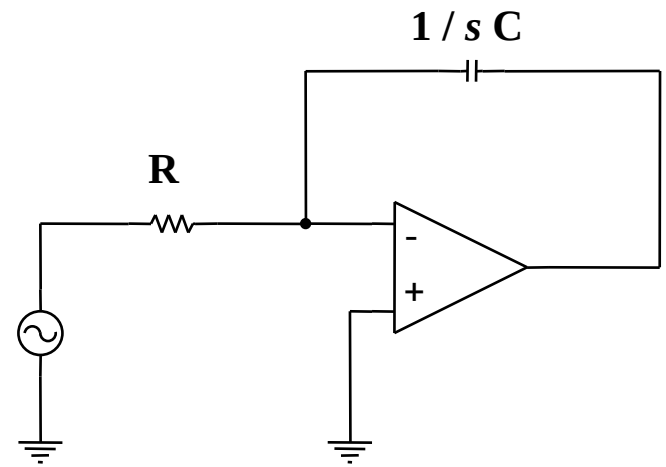
- Step 1: Design stable analog controller → H(s)
- Step 2: Convert analog design into digital → H[z]
  - Bilinear Transform (we will talk about this shortly)
- Step 3: Convert H[z] to the Linear Difference Equation
- Step 4: Get the MCU to calculate the Linear Difference Equation every cycle

Converting an Analog Transfer function H(s) into its Digital Counterpart H[z]

- Bilinear Transform
  - Also know as Tustin and Trapezoidal
  - It converts an analog transfer function in s domain into an equivalent digital transfer function in z domain
  - It is not an "exact" conversion:
    - It is an approximation
    - The lower the cross over frequency with respect to your sampling frequency the better the approximation:
    - For conservative design Fx <= Fs / 20
    - If starting with a stable analog design, it will always have poles and zeros in the stable region; but phase margin could be a problem due to digitization delays – we will talk about this later
  - All you need to do is to replace s operators in H(s) with:
    - Where Ts = sampling interval = 1/Fs

$$s \Rightarrow \frac{2}{T_s}\frac{(z-1)}{(z+1)}$$

Example:



1 - Derive H(s)

$$H(s) = -\frac{1}{s\,R\,C}$$

2 – Convert to H[z]

$$H[z] = -\frac{1}{\frac{2}{T_s}\frac{(z-1)}{(z+1)}R\,C} \rightarrow \frac{y[z]}{x[z]} = \frac{-T_s\,(z+1)}{2\,(z-1)R\,C}$$

3 – Convert to Linear Difference Equation

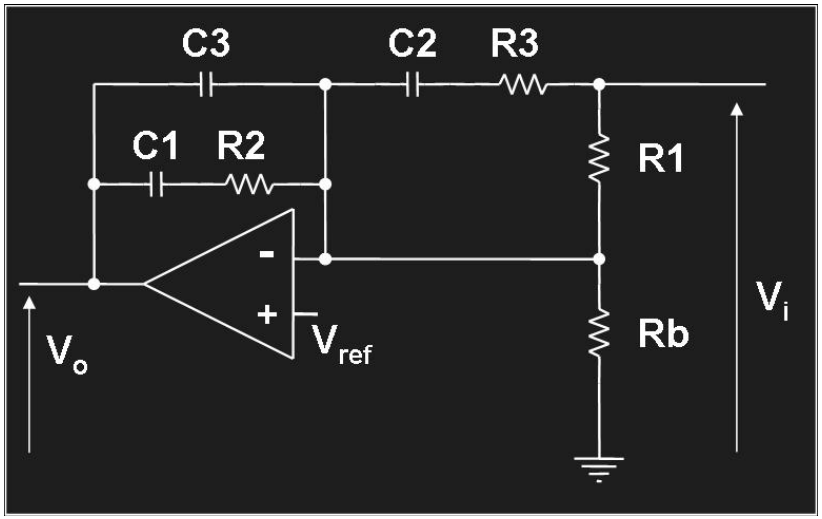
$$2\,RC\;y[z]\,z - 2\,RC\;y[z] = -T_s\;x[z]\,z - T_s\;x[z] \rightarrow 2\,RC\;y[z] - 2\,RC\;y[z]\,z^{-1} = -T_s\;x[z] - T_s\;x[z]\,z^{-1}$$

$$y[z] = \left(\frac{-T_s}{2\,RC}\right)x[z] + \left(\frac{-T_s}{2\,RC}\right)x[z]\,z^{-1} + y[z]\,z^{-1}$$

X and y delayed by 1 sampling interval

$$y[n] = \left(\frac{-T_s}{2\,RC}\right)x[n] + \left(\frac{-T_s}{2\,RC}\right)x[n-1] + y[n-1]$$

Digital Type III Controller (3P3Z)



- From previous slides

$$H_c(s) = \frac{\omega_{p0}}{s} \frac{\left(\frac{s}{\omega_{z1}}+1\right)\left(\frac{s}{\omega_{z2}}+1\right)}{\left(\frac{s}{\omega_{p2}}+1\right)\left(\frac{s}{\omega_{p3}}+1\right)}$$

* Where we select the position of the poles and zeros as described in the previous slides

- Applying bilinear transform just like our simple integrator example we have a "3 pole 3 zero" (3P3Z) digital controller:

$$H[z] = \frac{y[z]}{x[z]} = \frac{B_3\, z^{-3} + B_2\, z^{-2} + B_1\, z^{-1} + B_0}{-A_3\, z^{-3} - A_2\, z^{-2} - A_1\, z^{-1} + 1}$$

Convert to linear difference equation

$$\mathbf{y[n] = A_1\, y[n-1] + A_2\, y[n-2] + A_3\, y[n-3] + B_0\, x[n] + B_1\, x[n-1] + B_2\, x[n-2] + B_3\, x[n-3]}$$

- And now we know **ALL** the coefficients (please see next slide)

Digital Type III Controller (3P3Z)

- We have derived the following LDE and we now know ALL the coefficients

$$\mathbf{y[n] = A_1\, y[n-1] + A_2\, y[n-2] + A_3\, y[n-3] + B_0\, x[n] + B_1\, x[n-1] + B_2\, x[n-2] + B_3\, x[n-3]}$$

- Output coefficients are:

$$A_1 = -\frac{\left(-12 + T_s^2\, \omega_{p2}\, \omega_{p3} - 2\, T_s\left(\omega_{p2} + \omega_{p3}\right)\right)}{\left(2 + T_s\, \omega_{p2}\right)\left(2 + T_s\, \omega_{p3}\right)}$$

$$A_2 = -\frac{\left(-12 - T_s^2\, \omega_{p2}\, \omega_{p3} - 2\, T_s\left(\omega_{p2} + \omega_{p3}\right)\right)}{\left(2 + T_s\, \omega_{p2}\right)\left(2 + T_s\, \omega_{p3}\right)}$$

$$A_3 = \frac{\left(-2 + T_s\, \omega_{p2}\right)\left(-2 + T_s\, \omega_{p3}\right)}{\left(2 + T_s\, \omega_{p2}\right)\left(2 + T_s\, \omega_{p3}\right)}$$

Digital Type III Controller (3P3Z)

- Input coefficients are:

$$B_0 = \frac{\left(T_s\ \omega_{p0}\ \omega_{p2}\ \omega_{p3}\left(2+T_s\ \omega_{z1}\right)\left(2+T_s\ \omega_{z2}\right)\right)}{\left(2\left(2+T_s\ \omega_{p2}\right)\left(2+T_s\ \omega_{p3}\right)\omega_{z1}\ \omega_{z2}\right)}$$

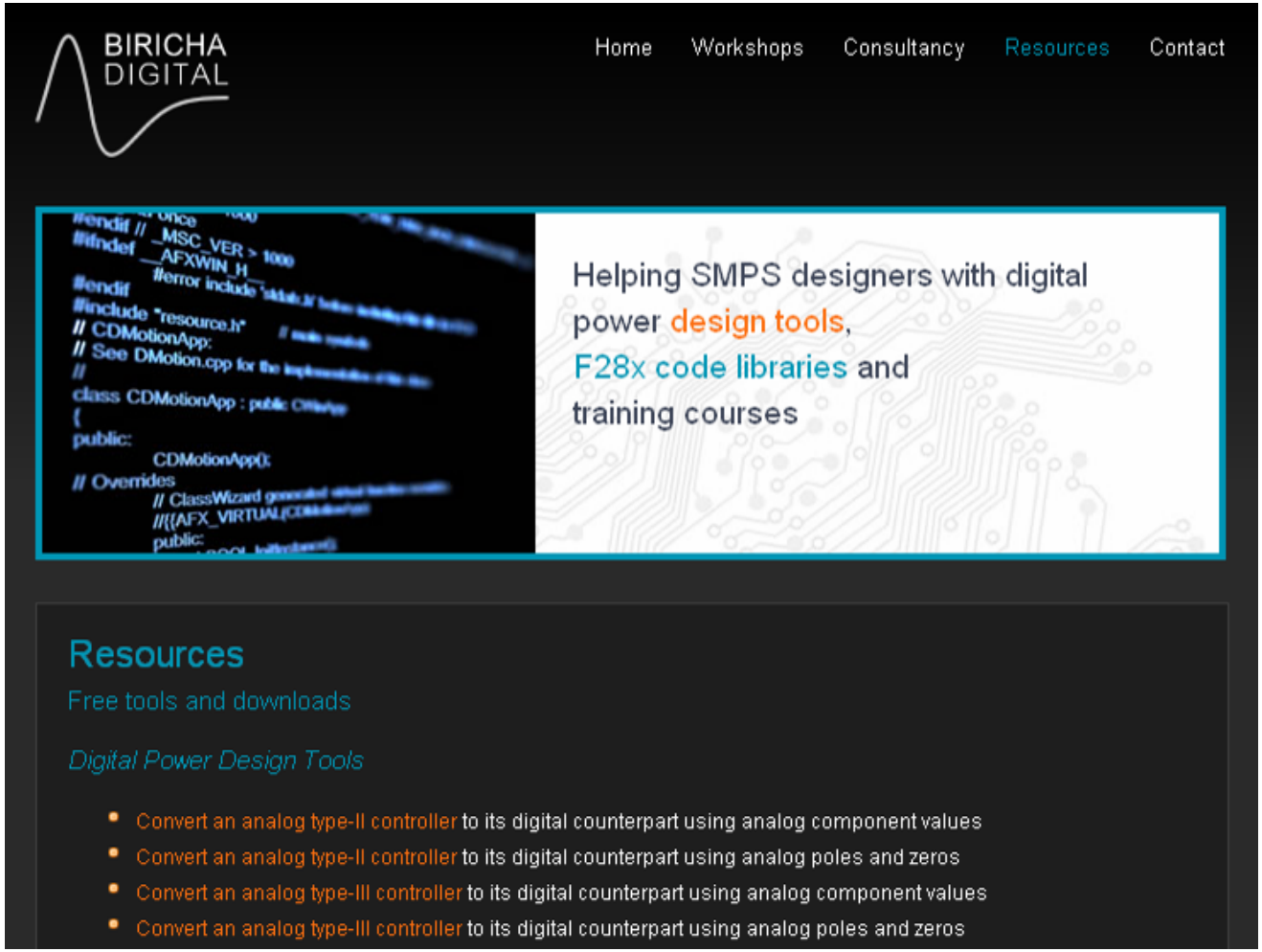
$$B_1 = \frac{\left(T_s\ \omega_{p0}\ \omega_{p2}\ \omega_{p3}\left(-4+3\,T_s^2\omega_{z1}\ \omega_{z2}+2T_s\left(\omega_{z1}+\omega_{z2}\right)\right)\right)}{\left(2\left(2+T_s\ \omega_{p2}\right)\left(2+T_s\ \omega_{p3}\right)\omega_{z1}\ \omega_{z2}\right)}$$

$$B_2 = \frac{\left(T_s\ \omega_{p0}\ \omega_{p2}\ \omega_{p3}\left(-4+3\,T_s^2\omega_{z1}\ \omega_{z2}-2T_s\left(\omega_{z1}+\omega_{z2}\right)\right)\right)}{\left(2\left(2+T_s\ \omega_{p2}\right)\left(2+T_s\ \omega_{p3}\right)\omega_{z1}\ \omega_{z2}\right)}$$

$$B_3 = \frac{\left(T_s\ \omega_{p0}\ \omega_{p2}\ \omega_{p3}\left(-2+T_s\ \omega_{z1}\right)\left(-2+T_s\ \omega_{z2}\right)\right)}{\left(2\left(2+T_s\ \omega_{p2}\right)\left(2+T_s\ \omega_{p3}\right)\omega_{z1}\ \omega_{z2}\right)}$$

We now know everything to calculate our digital coefficients but these procedures are cumbersome, so we have automated everything for you and placed it free on our website

Automated Digital Control Loop Design Tools



www.biricha.com/resources/

Automated Digital Control Loop Design Tools

Resources

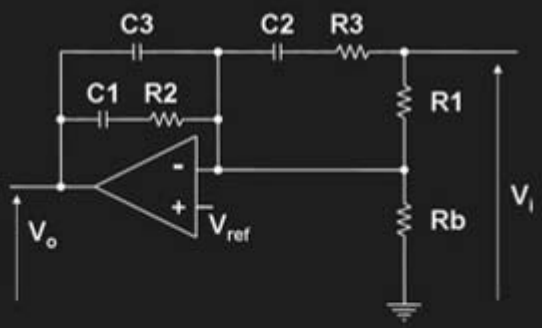
Analog to Digital Controller Converter

Convert an analog type-III controller to its digital counterpart using analog poles and zeros

This tool converts an analog type III controller to its digital counterpart (3 pole - 3 zero) using bilinear transform and analog poles and zeros.

Cross over frequency of analog pole at zero	833	Hz
Frequency of second pole	11668	Hz
Frequency of third pole	100000	Hz
Frequency of first zero	1617	Hz
Frequency of second zero	1617	Hz
Switching frequency	200000	Hz

Calculate



Automated Digital Control Loop Design Tools

Solution:

$$H(z) = \frac{U(n)}{E(n)} = \frac{B_3 Z^{-3} + B_2 Z^{-2} + B_1 Z^{-1} + B_0}{-A_3 Z^{-3} - A_2 Z^{-2} - A_1 Z^{-1} + 1}$$

$$U(n) = A_1 * U(n-1) + A_2 * U(n-2) + A_3 * U(n-3) + B_0 * E(n) + B_1 * E(n-1) + B_2 * E(n-2) + B_3 * E(n-3)$$

Where:

U(n) = current output
U(n-1) = output at the last sampling interval
U(n-2) = output two sampling intervals ago
U(n-3) = output three sampling intervals ago

E(n) = Error at the current sampling interval (i.e. reference - actual)
E(n-1) = Error at the last sampling interval
E(n-2) = Error two sampling intervals ago
E(n-3) = Error three sampling intervals ago

A_1 = 1.46818535
A_2 = -0.31493598
A_3 = -0.15324937

B_0 = 2.01823769
B_1 = -1.81826579
B_2 = -2.01328426
B_3 = 1.82321921

Correct coefficients for an equivalent digital power supply are automatically calculated

2p2z & 3p3z Library Functions

- We now have designed a digital controller and calculated the coefficients, but we still need to “program” the MCU:
  - We have written a comprehensive set of library function in order to make the programming easy and speed up your design
  - We call this library the “Chip Support Library” or the CSL
  - The CSL supports almost ALL of the C2000 series including F28069 and the Delfino
  - It has over 500 functions to easily set up peripherals such as PWMs Interrupts ADC etc
  - Most importantly it has dedicated 2p2z and 3p3z functions which only need the coefficients from the previous slide
  - The CSL is not free but all the attendees of the 3 Day workshop will receive a full commercial license as part of their course fee

Using the CSL's 3p3z Library functions

- Initialize the coefficients:
  - Ax and Bx Coefficients → from the website
  - **"REF"** is our input to the controller as per previous slide
    - Dependent on our potential divider and the ADC range (3.3V for Piccolo, 3.0V for 2808)
  - **"K"** is our scaling factor as per previous slide

```
#define A1  (+1.46818)
#define A2  (-0.314933)
#define A3  (-0.153248)
#define B0  (1.784224053)
#define B1  (-1.629063952)
#define B2  (-1.780916725)
#define B3  (1.632371281)

#define REF (_IQ15toF(2252))

#define K   (.732)
```

Using 2p2z & 3p3z Library functions

- Step 2: Declare an instance of the controller:

```
CNTRL_3p3zData Cntrl3p3z;
```

- Step 3: In main() initialize the controller:

```
CNTRL_3p3zInit(&Cntrl3p3z
,_IQ15(REF)                                  /*Ref*/
,_IQ26(A1),_IQ26(A2),_IQ26(A3)               /*a1,a2,a3*/
,_IQ26(B0),_IQ26(B1),_IQ26(B2),_IQ26(B3)     /*b0,b1,b2,b3*/
,_IQ24(0.0),_IQ24(0.9999)                    /* min, max duty */
);
```

- Step 4: In the ISR, input the ADC value into the controller and equate the output of the controller to PWM duty:

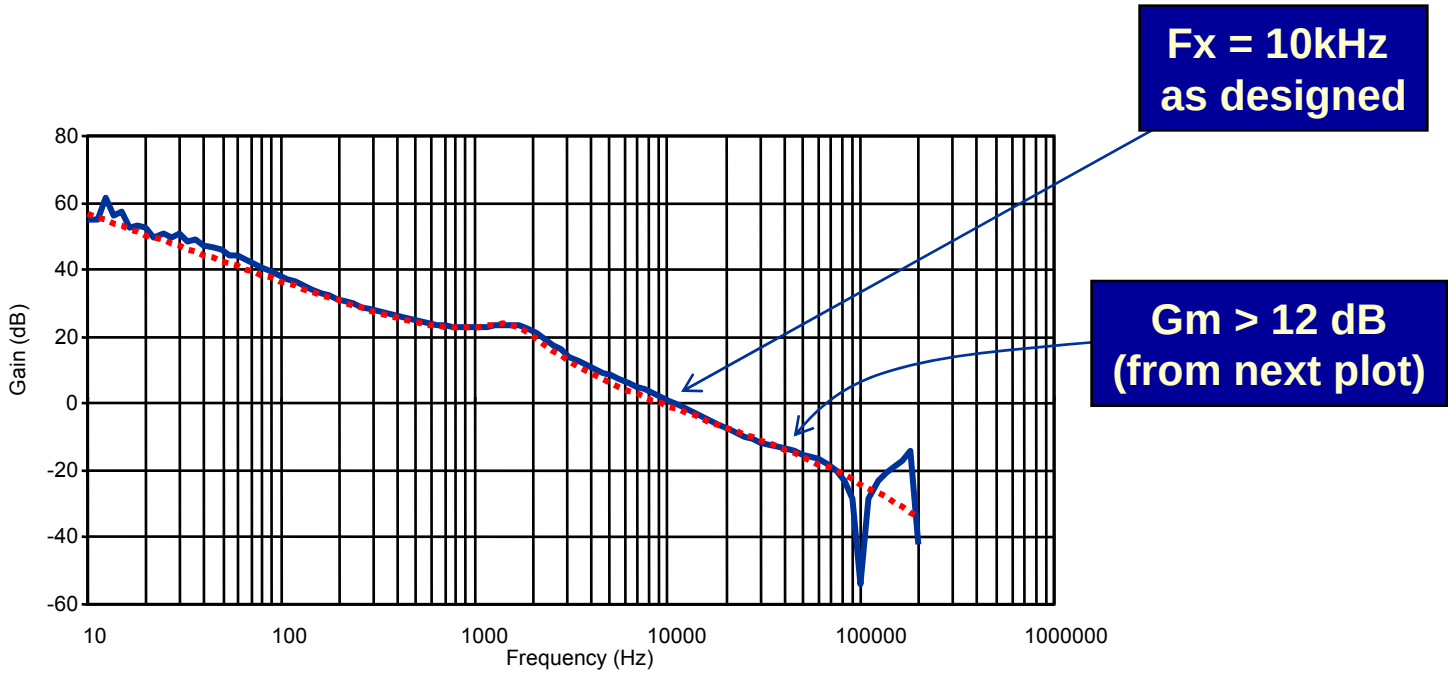
```
Cntrl3p3z.Fdbk.m_Int = ADC_getValue( ADC_MOD_1);
CNTRL_3p3z( &Cntrl3p3z );
PWM_setDutyA( PWM_MOD, Cntrl3p3z.Out.m_Int );
```

3p3z Design Example from the 3 Day Workshop

- BDP-106 Buck Converter Spec:
  - Vin = 12V; Vout = 3.3V; Iout = 2A
  - L = 22μH; DCR = 47mΩ
  - C = 440μF; ESR = 31mΩ
  - Fs = 200kHz; Fx = 10kHz
- In analog world we used a Type III controller with:
  - Fz1 = 1.61764 kHz
  - Fz2 = 1.61764 kHz
  - Fp2 = 11.6682 kHz
  - Fp3 = 100 kHz
  - Fp0 = 833 Hz

Analyzing Your Converter in Frequency Domain

- Gain plot of the real practical converter, running at 200kHz



Measured gain plot of the actual digital PSU

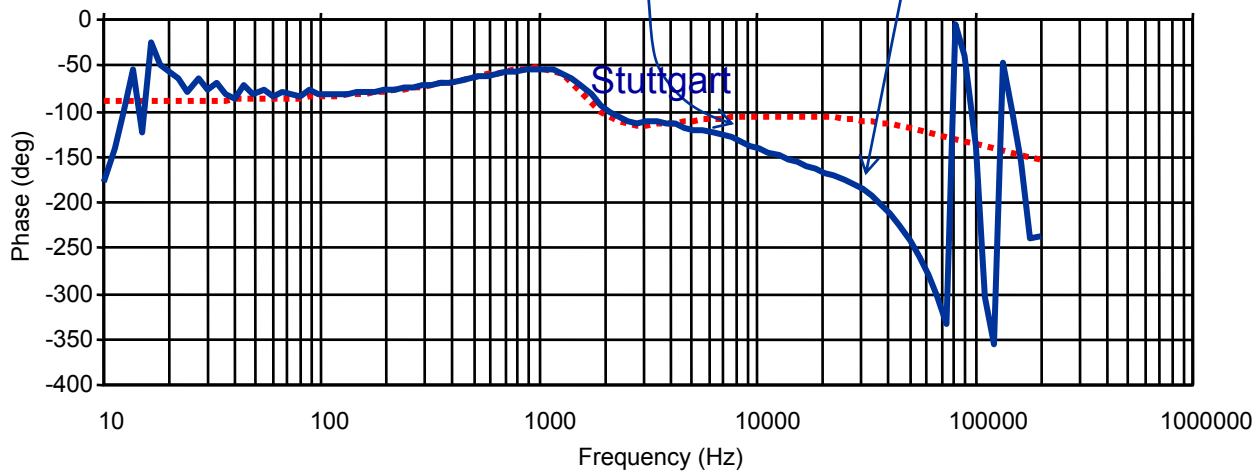
Simulated gain plot of the equivalent analog PSU

Analyzing Your Converter in Frequency Domain

- Phase plot of real practical converter

Phase erosion at Fx in the digital PSU
Analog $\phi_m \approx 75°$
Digital $\phi_m \approx 45°$

Digital Phase deteriorates rapidly as we approach switching frequency

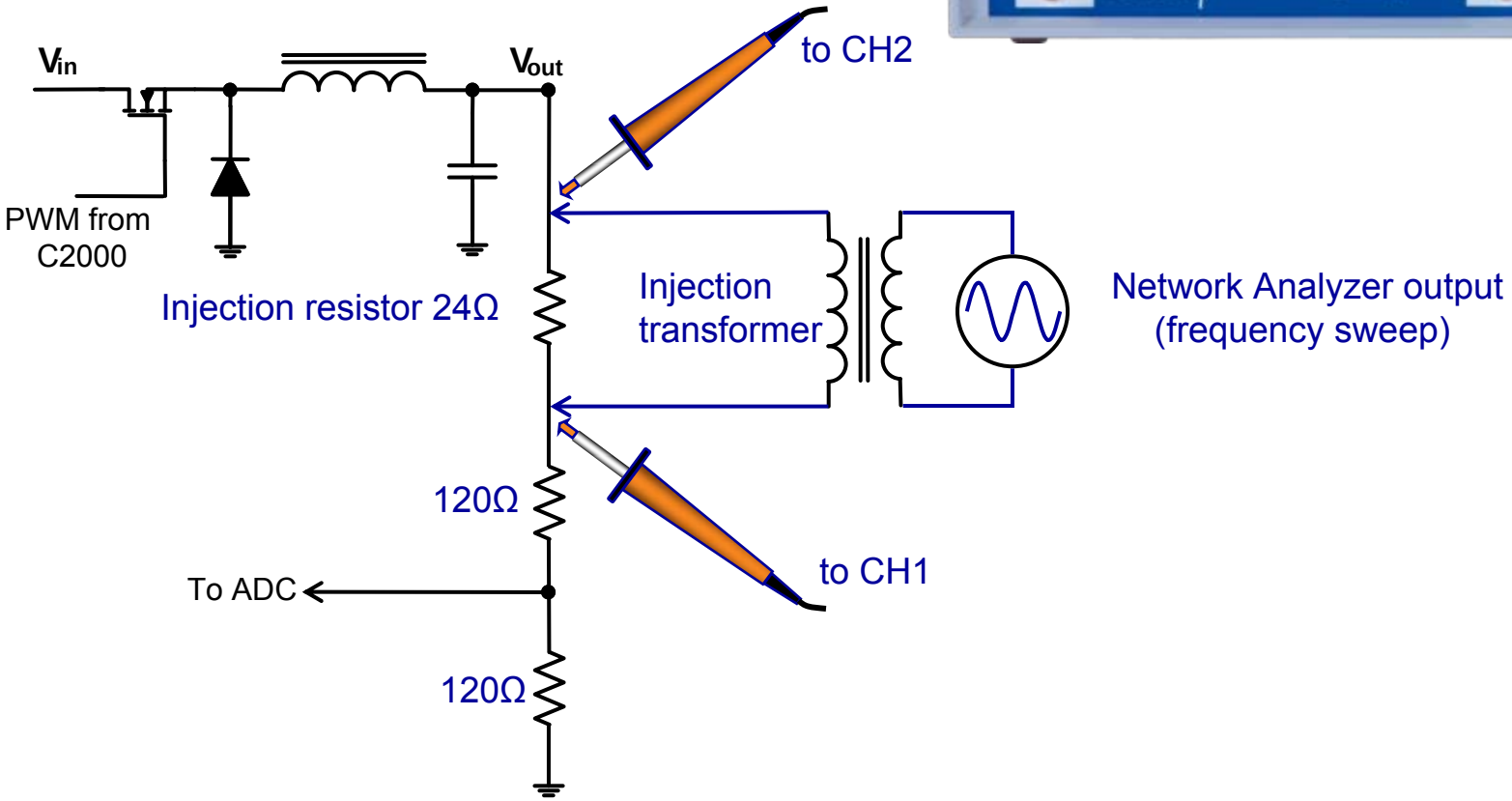


Measured phase plot of the actual digital PSU

Simulated phase plot of the equivalent analog PSU

Analyzing Your Converter in Frequency Domain

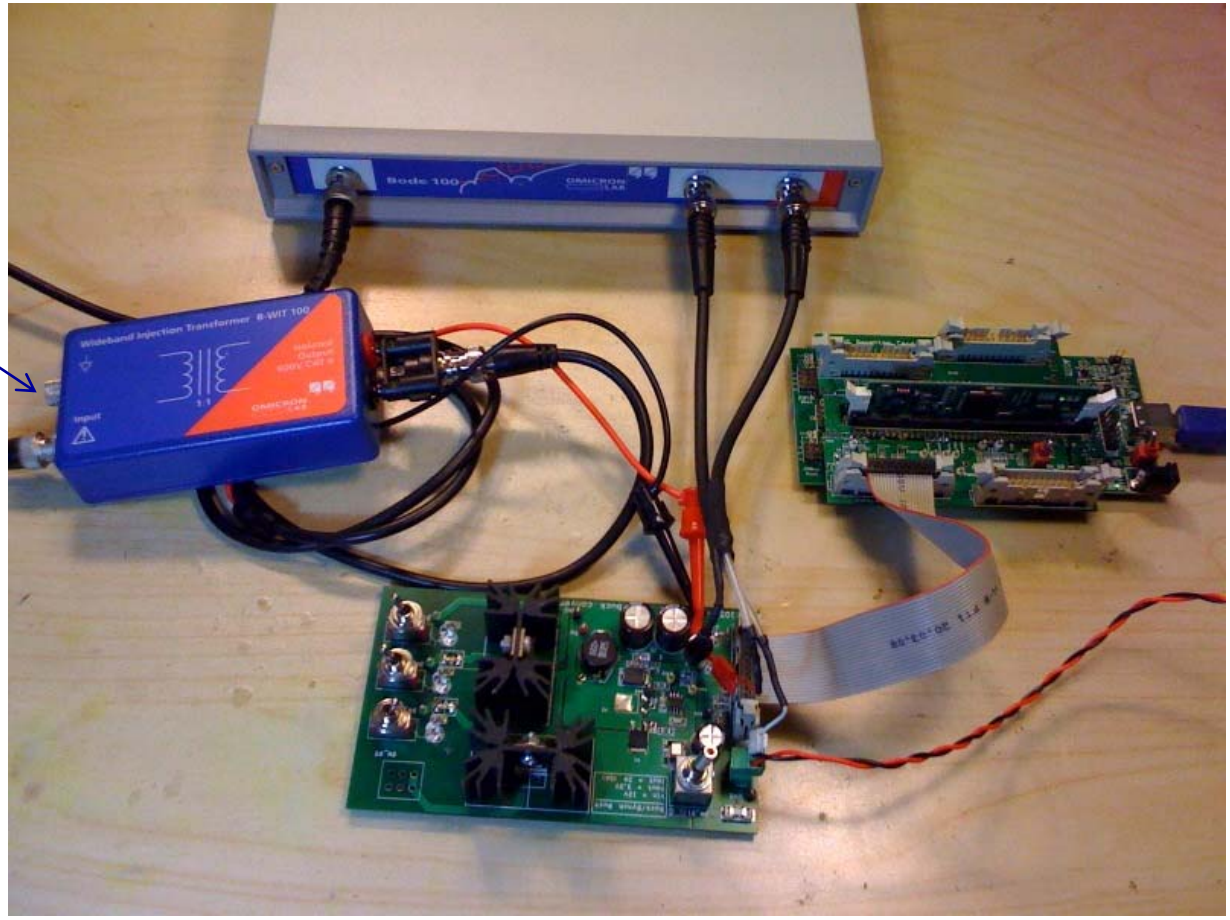
- Measurement setup with a network analyzer



Analyzing Your Converter in Frequency Domain

- Measurement setup

Injection transformer

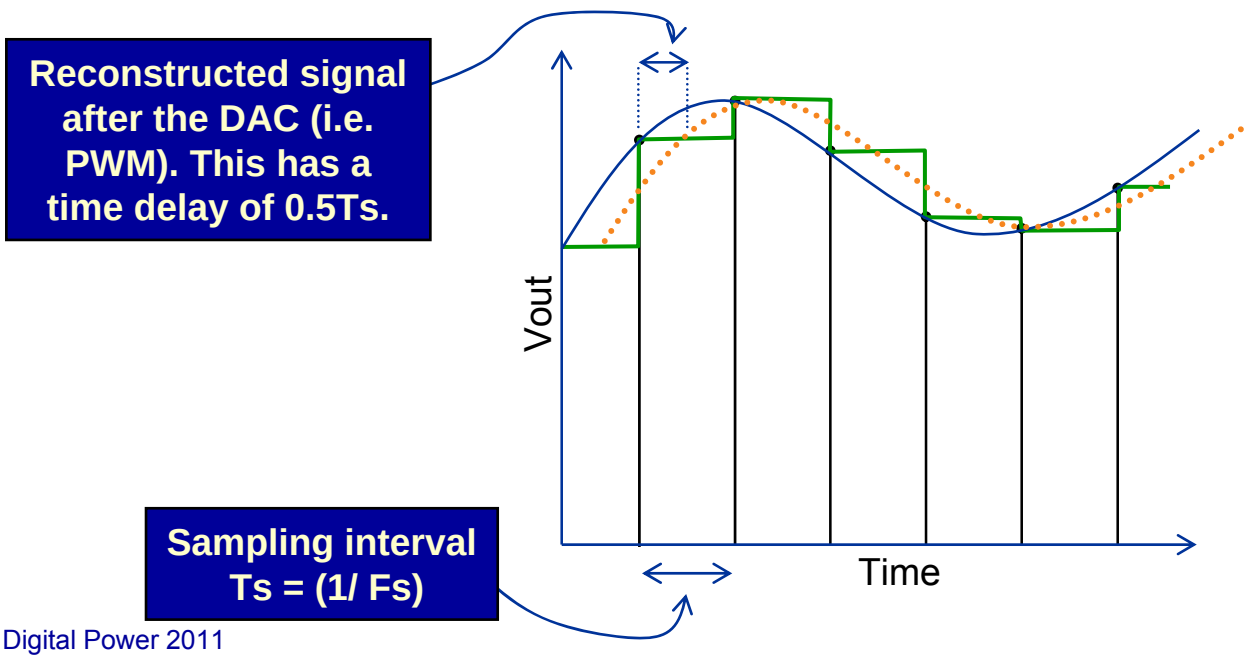


Phase Margin Erosion in Digital World (Voltage Mode Only)

- From previous plots our digital PSU has a
  - G_M of better than 12 dB
  - ϕ_M of better than 40°
- This is a "very good" power supply but we designed our analog supply to have a phase margin of 75°
  - We designed for 75 degrees on purpose!
- The digitization process has introduced an extra phase lag of about 30 degrees!
  - This is an inherent characteristic of the sampling process
  - We must calculate this phase lag and allow for it in our analog design
  - We have included extra slides about this topic at the end of this presentation

Phase Margin Erosion in Digital World

- There are two mechanisms that introduce phase lag in to our digital power supply:
  - The sampling and reconstruction process
  - The time taken from the time we sampled our voltage to the time we carried out our calculations i.e. Time delay, Td



Phase Margin Erosion in Digital World

- For a pure sine wave, phase delay ϕ at a certain frequency f is given by: $\phi = 360° \times f \times Time\ Delay$
  - The sampling process and reconstruction introduces a time delay of 0.5xTs. Therefore the phase margin erosion at Fx:

$$\phi_{sampling} = 360° \times F_x \times \frac{T_s}{2}$$

In our case:

(360° x 10kHz x 5μs /2) = 9°

  - Phase margin erosion at Fx due to calculation delay:

$$\phi_{calculation} = 360° \times F_x \times k\,T_s$$

In our case:

(360° x 10kHz x 5μs x 1 = 18°

Where:
k = in the number of sampling intervals (need not be an integer).
Worst case scenario: we do the calculation at the beginning of the first sampling interval and update in the next; therefore k = 1

Total phase delay = 27°. We designed our analog controller with a large phase margin of 75° so that our digital controller would have a perfect phase margin of 48°

Final Remarks

- Forthcoming 3 day workshops:
  - 1st – 3rd of Nov 2011 Copenhagen, Denmark
  - 22nd -24th of Nov 2011 Stuttgart, Germany
- Free of charge digital controller coefficient calculator on:
  - www.biricha.com/resources
- Chip Support Library
  - Free limited evaluation version on www.biricha.com/resources
  - Commercial license included in the price of the 3 day workshop
- Workshop price is 1795 Euros but all attendees of this seminar receive a 10% discount i.e. 1615 Euros
- More info on:

www.ti.com/biricha

Conclusion

- Digital power is gaining momentum and will be a significant player in the future of power management
- In many applications digital power has advantages over analog
- Designing digital power need not be difficult; TI provides:
  - Dedicated MCUs
  - Excellent development tools
  - Bespoke, in-depth training and support

TI would like to be partner in your digital power applications

END of Seminar
Thank you for taking part