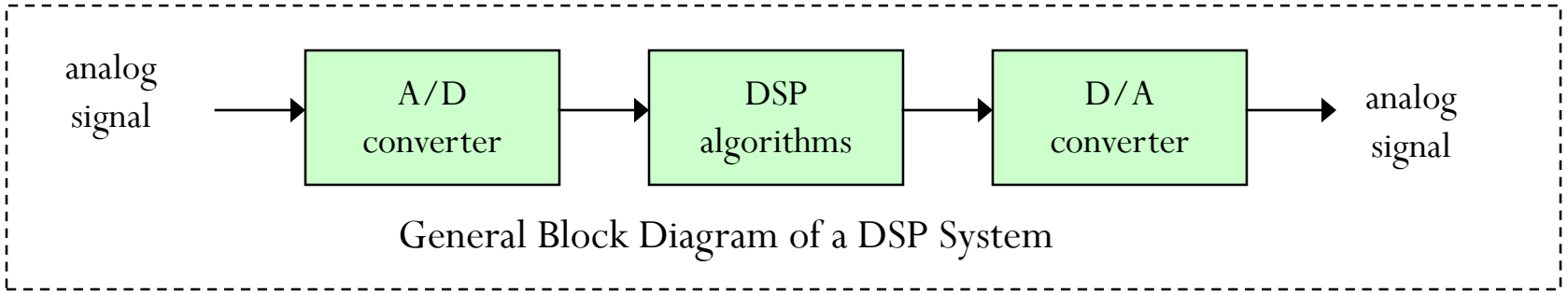
DSP

- Digital Signal Processing (DSP)
  - Is the manipulation of digital signals in order to modify their characteristic or to extract useful information. Why digital and not analog?
  - Digital signal allow programmability,
  - Digital circuit allow for stable output than analog
  - Microprocessors and computers have become so powerful
  - only digitized signal can be processed by computers.
- Digital Signal Processor (DSP)
  - DSP is a specialized microprocessor optimized for signal processing.
  - General purpose microprocessors such as Pentium series microprocessors that are used in PC are not optimized for signal processing purposes.



General Block Diagram of a DSP System

Hardware tools:

DSP (DSKs), evaluation modules (EVMs) and other DSP boards

➔ For real-time DSP experiments, a DSK/EVM/Emu. is suitable along with a host system, which can be a typical PC.

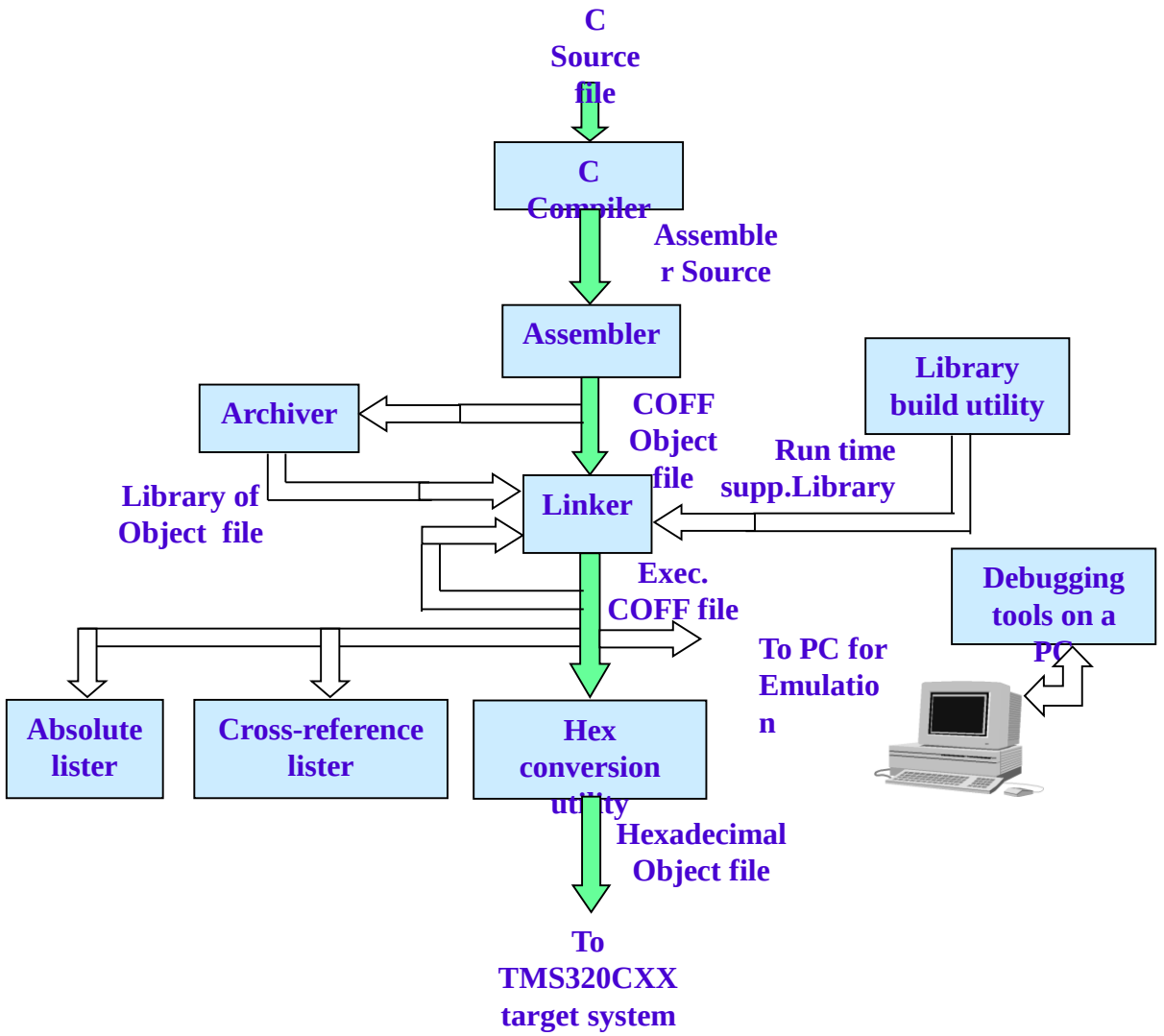
Software tools:

Assembly language tools, DSP simulator, C compiler and C source debugger.

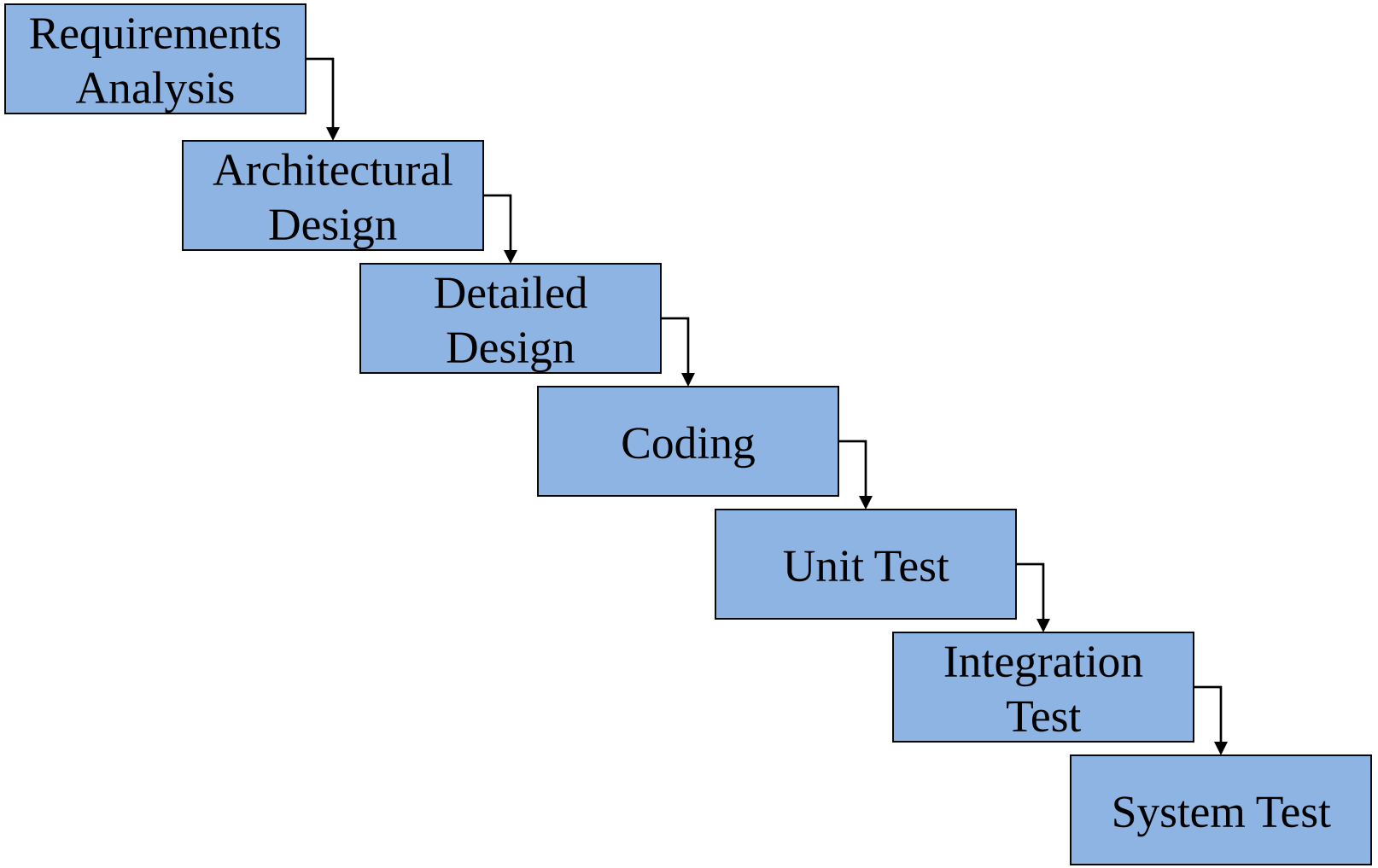
Code Composer Studio (CCS) ➔ IDE:

Simulates, C compiles and works with a DSK

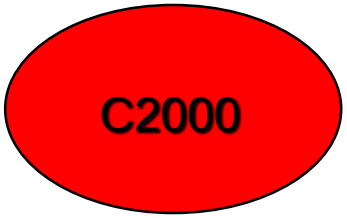
DSP Software Development flow

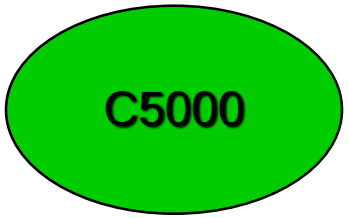


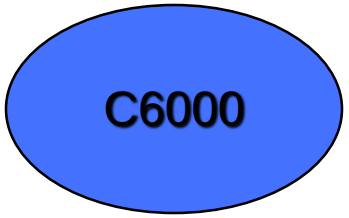
Software Life Cycle - Waterfall Method



Texas Instruments' TMS320 family







Lowest Cost

Control Systems

- Motor Control
- Storage
- Digital Ctrl Systems

Efficiency

Best MIPS per Watt / Dollar / Size

- Wireless phones
- Internet audio players
- Digital still cameras
- Modems
- Telephony
- VoIP

Performance & Best Ease-of-Use

- Multi Channel and Multi Function App's
- Comm Infrastructure
- Wireless Base-stations
- DSL
- Imaging
- Multi-media Servers
- Video

TMS320 DSP Families

C2000

>50 Products
ASP: $3 - $15

- World's most code-efficient DSP
- Advanced **embedded control applications**
- Leadership integration of analog and high-speed **Flash memory**
- **C28x** fully **code compatible**

C5000

>100 Products
ASP: $5 - $120

- World's most **power-efficient** DSP
- World's **most popular** DSP
- Heart of handheld solutions in Internet era
- **C55x** fully **code compatible**

C6000

>30 Products
ASP: $10 - $350

- World's **highest-performance** DSP
- Used in high-bandwidth comms and video equipment
- **C64x** fully **code compatible**

Floating vs. Fixed point processors

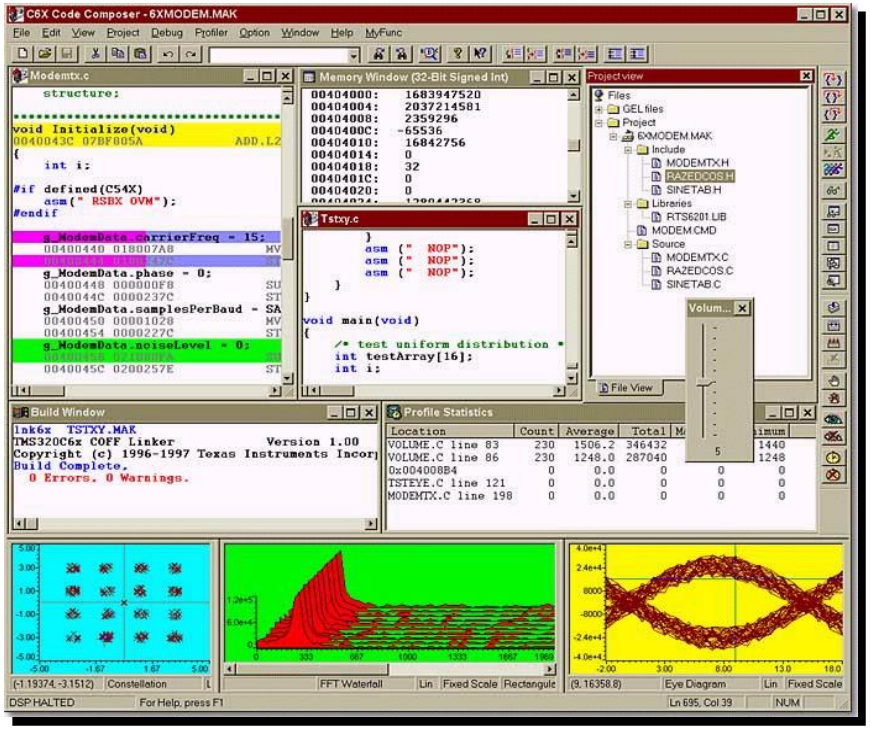
- Applications which require:
  - High precision.
  - Wide dynamic range.
  - High signal-to-noise ratio.
  - Ease of use.

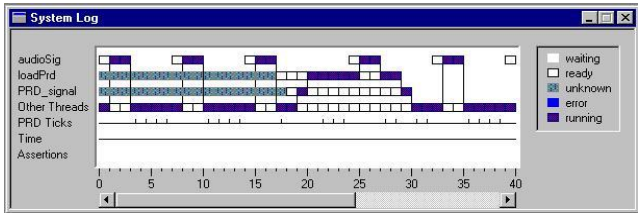
  Need a floating point processor.
- Drawback of floating point processors:
  - Higher power consumption.
  - Can be more expensive.
  - Can be slower than fixed-point counterparts and larger in size.

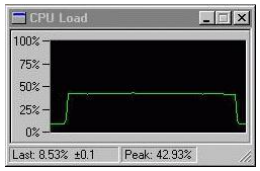
Floating vs. Fixed point processors

- It is the application that dictates which device and platform to use in order to achieve optimum performance at a low cost.
- For educational purposes, use the floating-point device (C6713) as it can support both fixed and floating point operations.
- Fixed point processors:
- TMS320c2X, TMS320c5X and TMS320c62X
- (Modulators, demodulators, carrier and clock recovery etc.,)
- Floating point processors:
- TMS320c3X and TMS320c67X
- (Speech processing, control systems, equalization etc.,)

Code Composer Studio





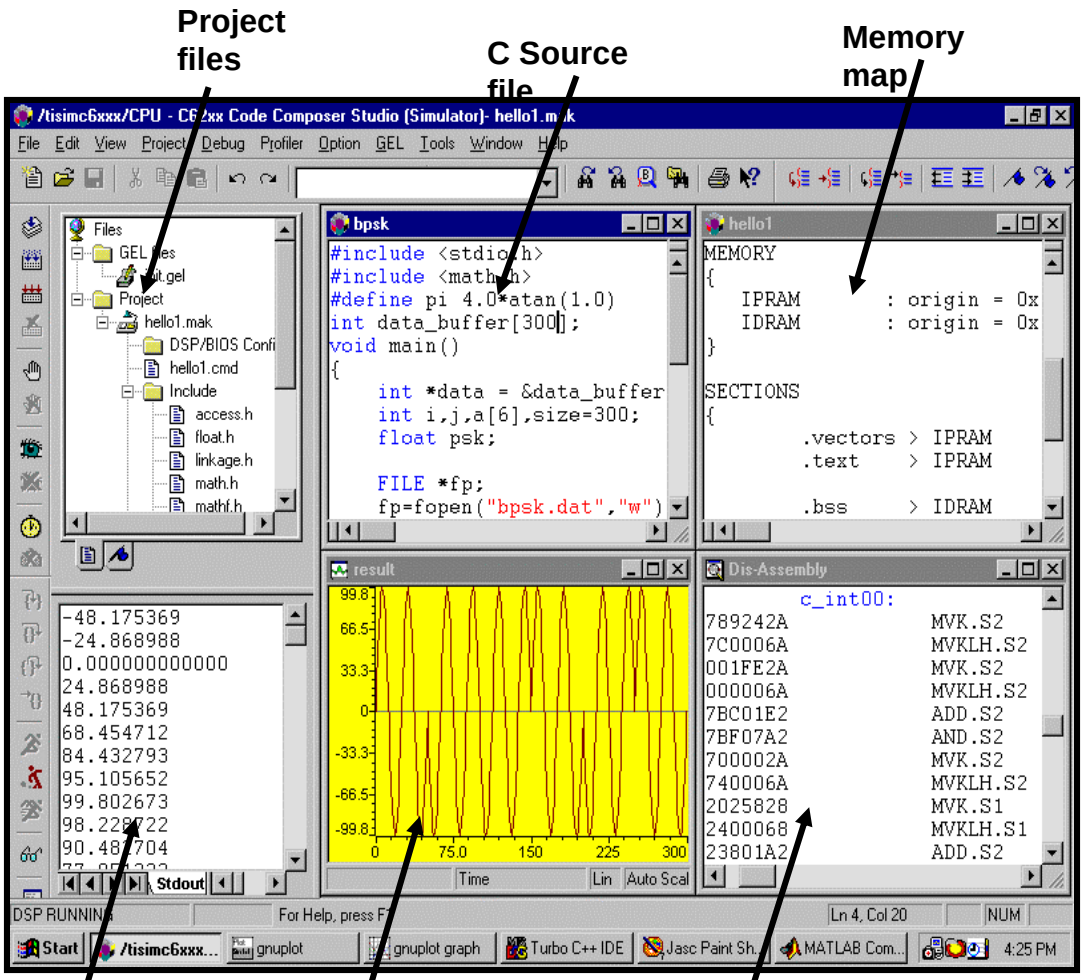


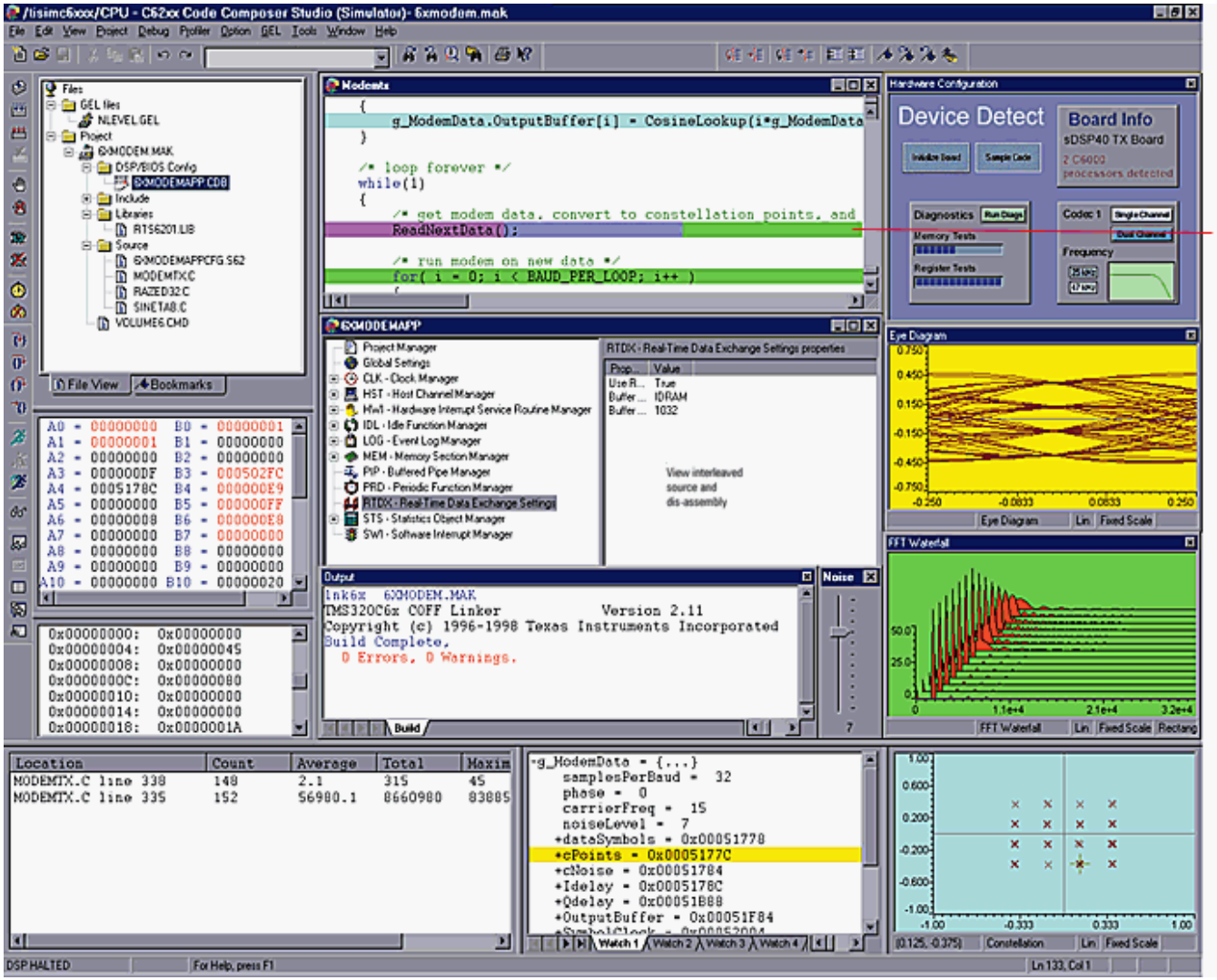
- DSP industry's first comprehensive, open Integrated Development Environment (IDE)
- Advanced visualization
- Intuitive ease-to-use
- Third-party plug-ins
- Visualization without stopping the processor

The CCS is an integrated suite of DSP software development tools

efficient 'C6000 C compiler, Assembly Optimizer with the Code Composer IDE, Advanced Data Visualization, **standard open APIs,** ***DSP/BIOS and Real-Time Data Exchange(RTDX)***

- **Optimizing C compiler ➔** ***fully exploits the architecture's instruction-level parallelism and orthogonal instruction set***
- **Assembly optimization ➔** ***supports automatic scheduling, optimizing and separation of parallel tasks from linear assembly code***
- **Debugger ➔** ***Conditional or hardware breakpoints are based on full C-expressions, local variables or CPU register symbols*****.**
- **Real-Time Analysis ➔*****Using RTDX technology, DSP/BIOS provides a real-time window into the target system***


Project files
C Source file
Memory map
/tisimc6xxx/CPU - C6xxx Code Composer Studio (Simulator)- hello1.mak
File Edit View Project Debug Profiler Option GEL Tools Window Help
Files
GEL Files
init.gel
Project
hello1.mak
DSP/BIOS Confi
hello1.cmd
Include
access.h
float.h
linkage.h
math.h
mathf.h
bpsk
#include <stdio.h>
#include <math.h>
#define pi 4.0*atan(1.0)
int data_buffer[300];
void main()
{
int *data = &data_buffer
int i,j,a[6],size=300;
float psk;
FILE *fp;
fp=fopen("bpsk.dat","w")
hello1
MEMORY
{
IPRAM : origin = 0x
IDRAM : origin = 0x
}
SECTIONS
{
.vectors > IPRAM
.text > IPRAM
.bss > IDRAM
-48.175369
-24.868988
0.000000000000
24.868988
48.175369
68.454712
84.432793
95.105652
99.802673
Stdout
result
99.8
66.5
33.3
0
-33.3
-66.5
-99.8
0
75.0
150
225
300
Time
Lin
Auto Scal
Dis-Assembly
c_int00:
789242A MVK.S2
7C0006A MVKLH.S2
001FE2A MVK.S2
000006A MVKLH.S2
7BC01E2 ADD.S2
7BF07A2 AND.S2
700002A MVK.S2
740006A MVKLH.S2
2025828 MVK.S1
2400068 MVKLH.S1
23801A2 ADD.S2
DSP RUNNING
For Help, press F1
Ln 4, Col 20
NUM
Start
/tisimc6xxx...
gnuplot
gnuplot graph
Turbo C++ IDE
Jasc Paint Sh...
MATLAB Com...
4:25 PM
Data display
Graphics Display
Dis-Assembly window (Assembly source)



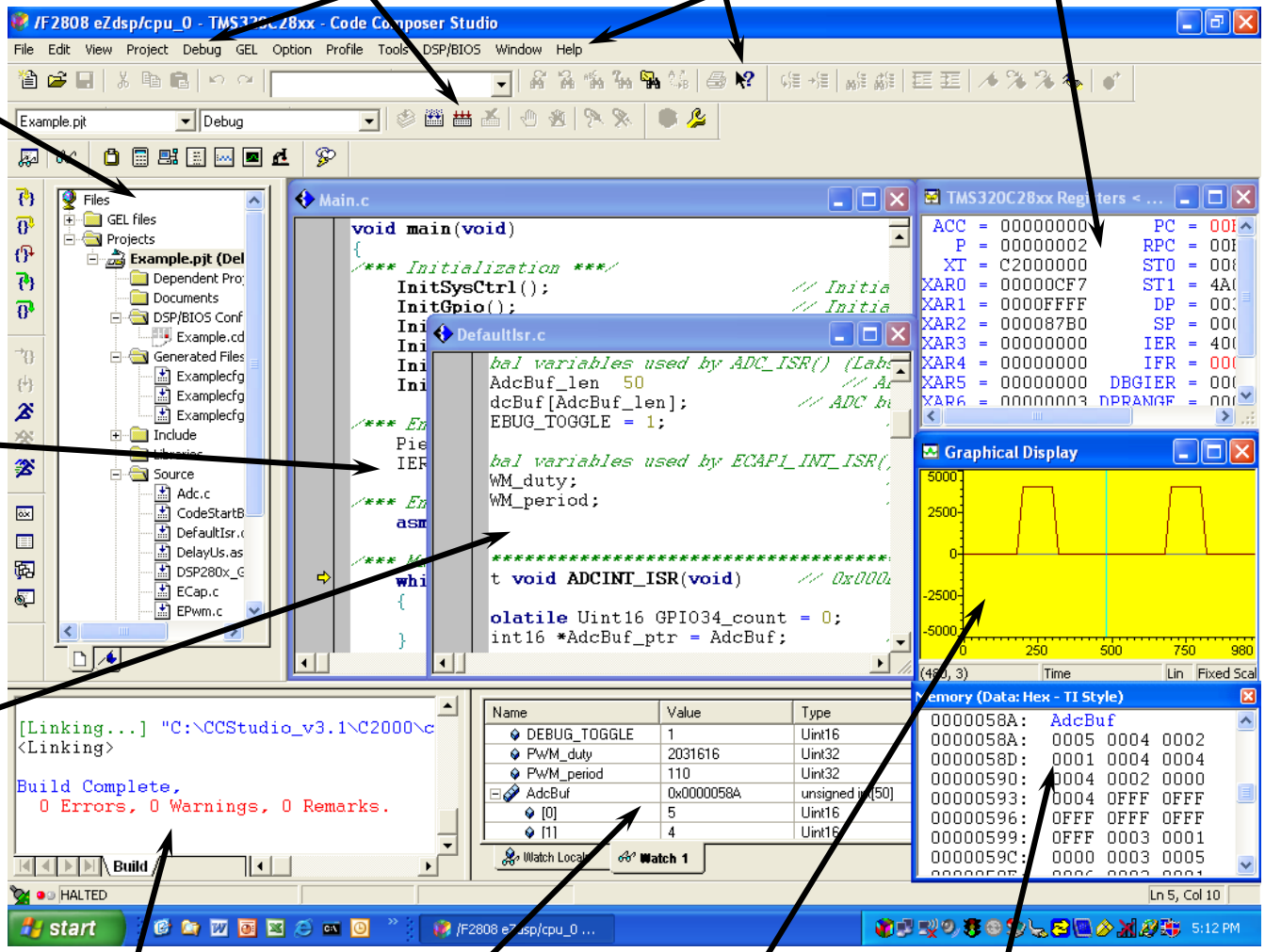
View interleaved source and dis-assembly

Code Composer Studio

Project Manager:
- Source & object files
- File dependencies
- Compiler, Assembler & Linker build options

Full C/C++ & Assembly Debugging:
- C & ASM Source
- Mixed mode
- Disassembly (patch)
- Set Break Points
- Set Probe Points

Editor:
- Structure Expansion

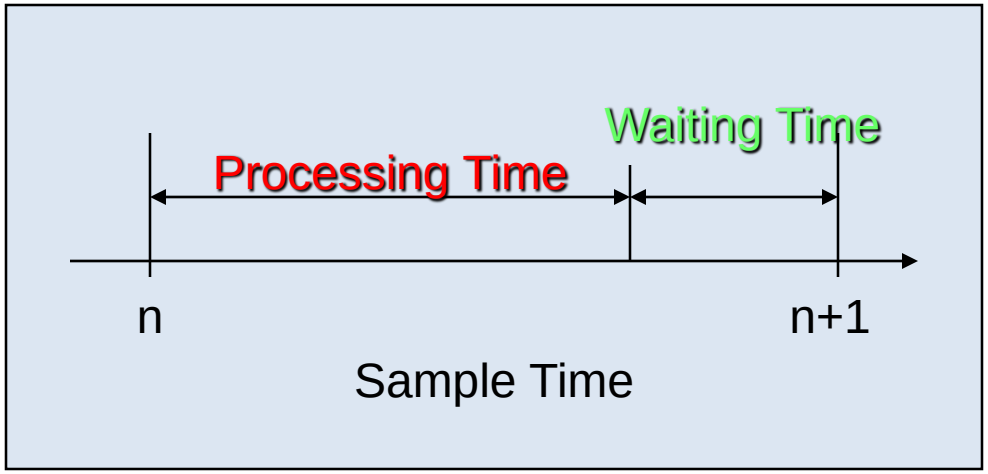


Real-Time Processing

- Real-time processing means:
  - The processing of a particular sample must occur within a given time period or the system will not operate properly.
  - Real-time DSP is inherently an ***interrupt*** driven process. The input samples should only be processed using interrupt service routines (ISR).
- Hard real-time system
  - The system will fail if the processing is not done in a timely manner.
- Soft real-time system
  - The system will tolerate some failures to meet real-time targets and still continue to operate, but with some degradation in performance.

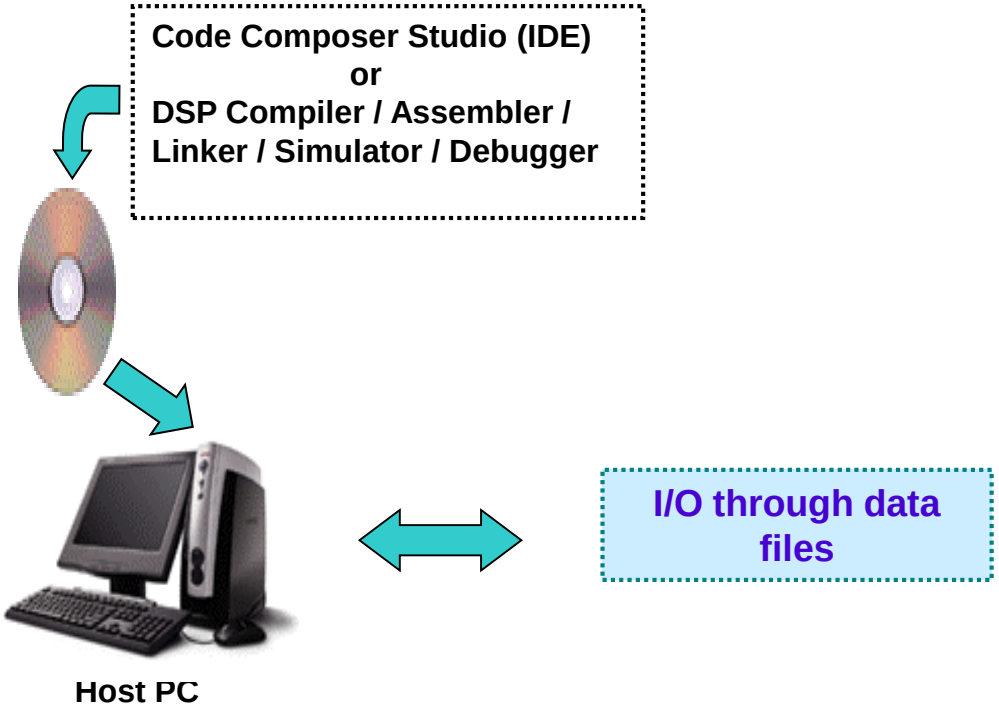
- The performance demands and power constraints of real-time systems often mandate specialized hardware.
  - That may include the digital signal processor (DSP), programmable logic devices, application specific integrated circuits (ASIC), and etc.

Real-time processing



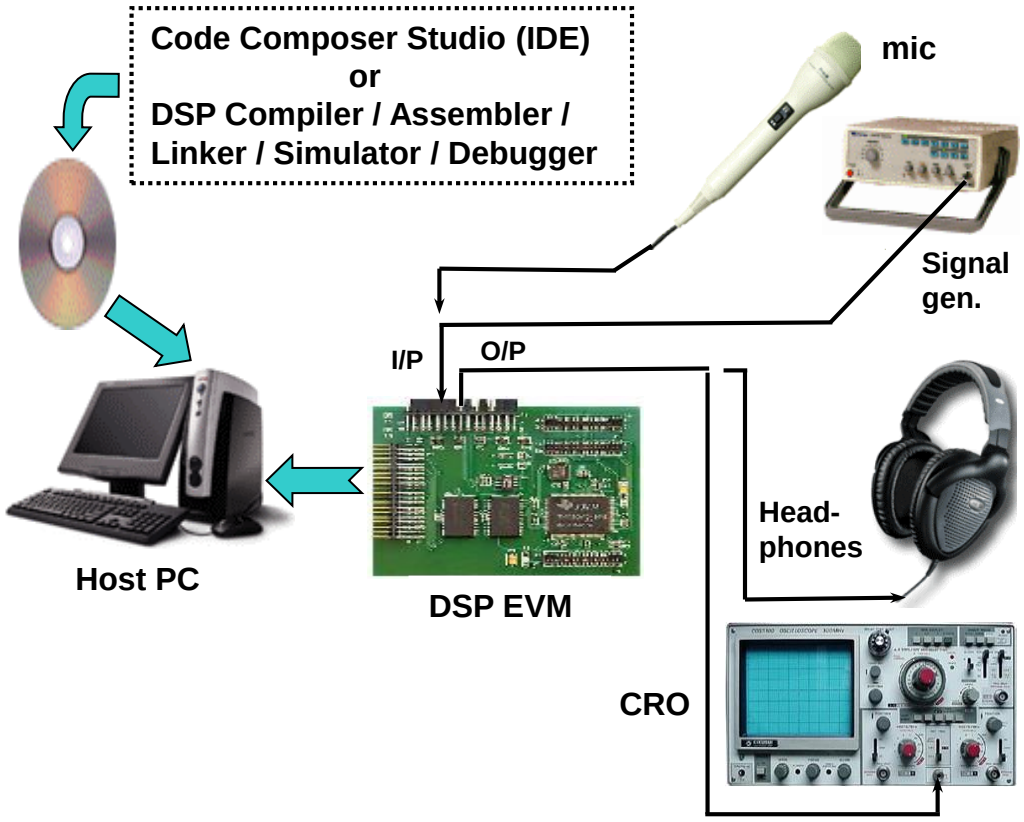
- We can say that we have a real-time application if:
  - Waiting Time ≥ 0
- DSP processors have to perform tasks in real-time, so how do we define real-time?
- The definition of real-time depends on the application.

A Setup for Non-real-time Experiments



Assembly language code and implementation flavor is present, but real-time experiments cannot be carried out using this setup.

A Setup for Real-time Experiments



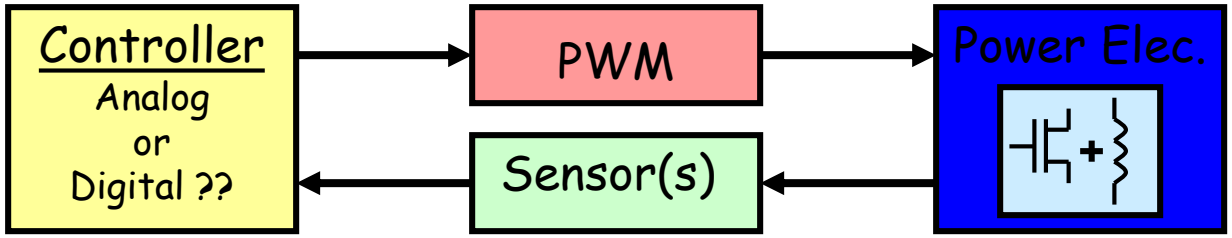
Assembly language code and implementation flavor is present.
Real-time experiments can be carried out using this setup.

Hardware vs. Microcode multiplication

- DSP processors are optimized to perform multiplication and addition operations.
- Multiplication and addition are done in hardware and in one cycle.
- Example: 4-bit multiply (unsigned).

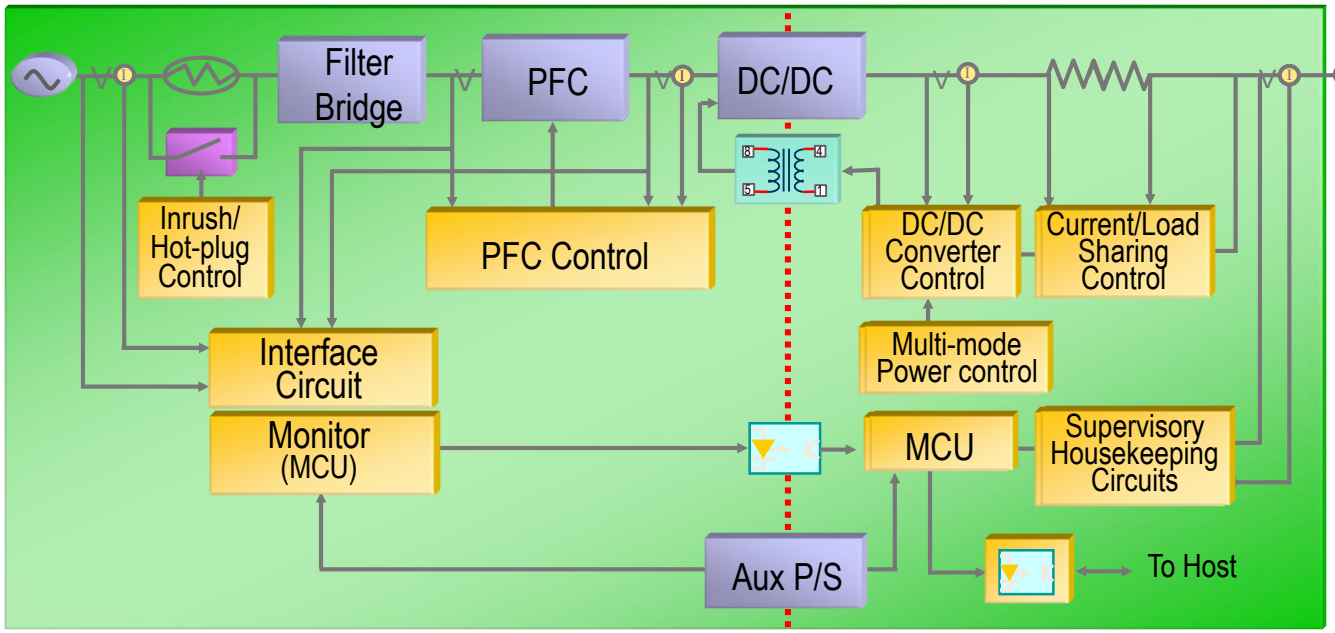
Hardware	Microcode	
1011	1011	
x 1110	x 1110	
10011010	0000	Cycle 1
	1011.	Cycle 2
	1011..	Cycle 3
	1011...	Cycle 4
	10011010	Cycle 5

Why Digital Control Techniques?



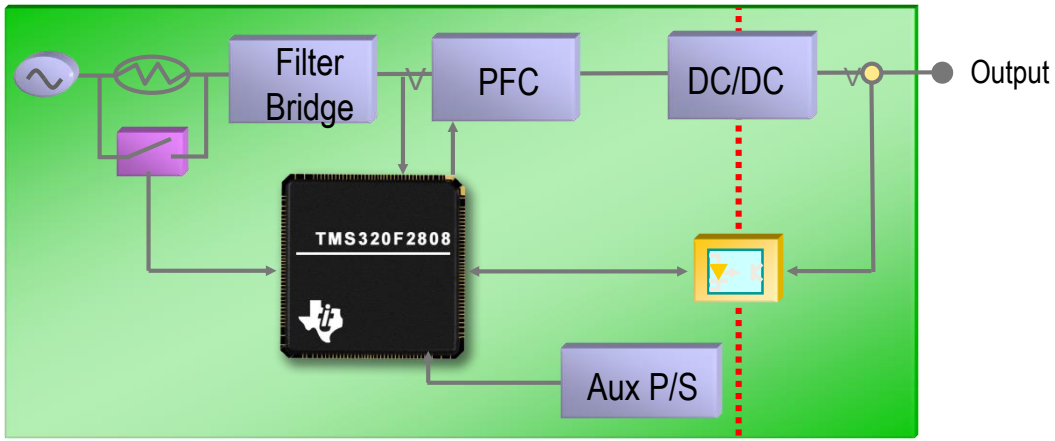
	Analog Controller	Digital Controller
+	◆ High bandwidth ◆ High resolution ◆ Easy to understand / use ◆ Historically lower cost	◆ Insensitive to environment (temp, drift,…) ◆ S/w programmable / flexible solution ◆ Precise / predictable behavior ◆ Advanced control possible (non-linear, multi-variable) ◆ Can perform multiple loops and “other” functions
—	◆ Component drift and aging / unstable ◆ Component tolerances ◆ Hardwired / not flexible ◆ Limited to classical control theory only ◆ Large parts count for complex systems	◆ Bandwidth limitations (sampling loop) ◆ PWM frequency and resolution limits ◆ Numerical problems (quantization, rounding,…) ◆ AD / DA boundary (resolution, speed, cost) ◆ CPU performance limitations ◆ Bias supplies, interface requirements

Benefits of Digital Control



Traditional Analog Power Supply

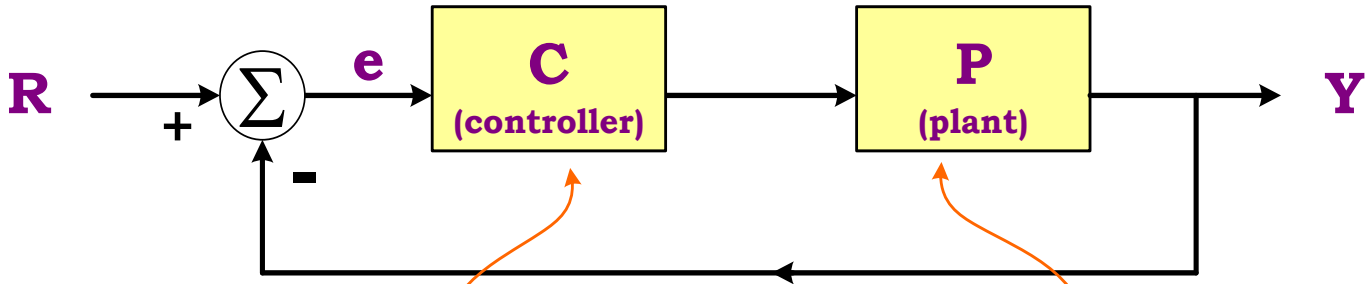
- Multiple chips for control
- Micro-controller for supervisory
- Dedicated design



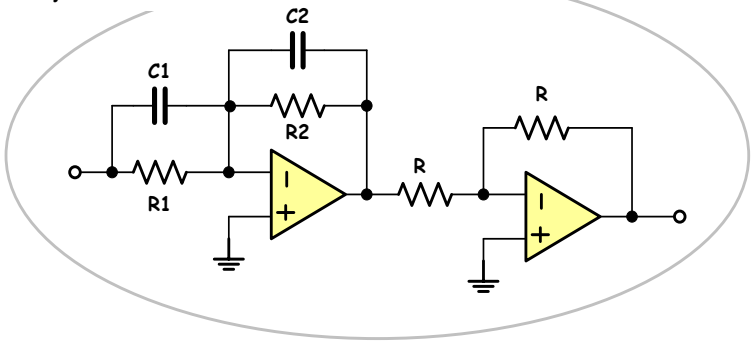
- ✓ Eliminate Components
- ✓ Reduce Manufacturing Cost
- ✓ Better Performance Across Corners
- ✓ One Design, Multiple Supplies
- ✓ Failure Prediction
- ✓ One Device, Multiple DC Outputs
- ✓ Variable DC Output

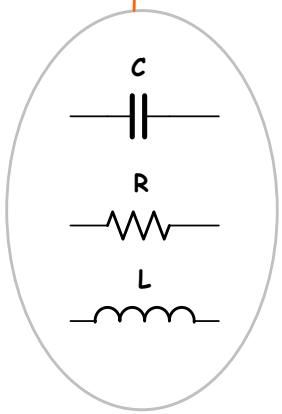
Digital controller enables multi-threaded applications

Analog Control System



"Analog Computation"
Differential equations





Energy Storage Elements

$$C(s) = \frac{R_2}{R_1}\left(\frac{1+R_1C_1s}{1+R_2C_2s}\right)$$

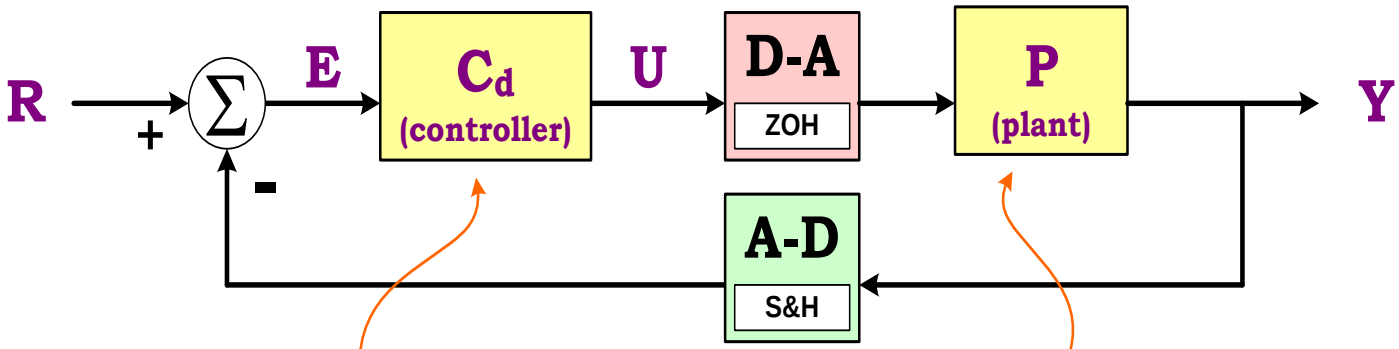
Need to find:
R_1, R_2, C_1, C_2

$$\frac{d^3y(t)}{dt^3} + k_2\frac{d^2y(t)}{dt^2} + k_1\frac{dy(t)}{dt} + k_0y(t) = f(t)$$

Differential equations
1st, 2nd, 3rd,...order

Laplace Transform

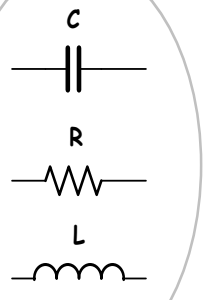
Digital Control System



Difference equation

$$U(n) = a_2 \cdot U(n-2) + a_1 \cdot U(n-1) + b_2 \cdot E(n-2) + b_1 \cdot E(n-1) + b_0 \cdot E(n)$$

$$where \ldots E(n) = R(n) - Y(n)$$



Energy Storage Elements

Need to find:
a_1, a_2, b_0, b_1, b_2

$$\frac{d^3y(t)}{dt^3} + k_2\frac{d^2y(t)}{dt^2} + k_1\frac{dy(t)}{dt} + k_0 y(t) = f(t)$$

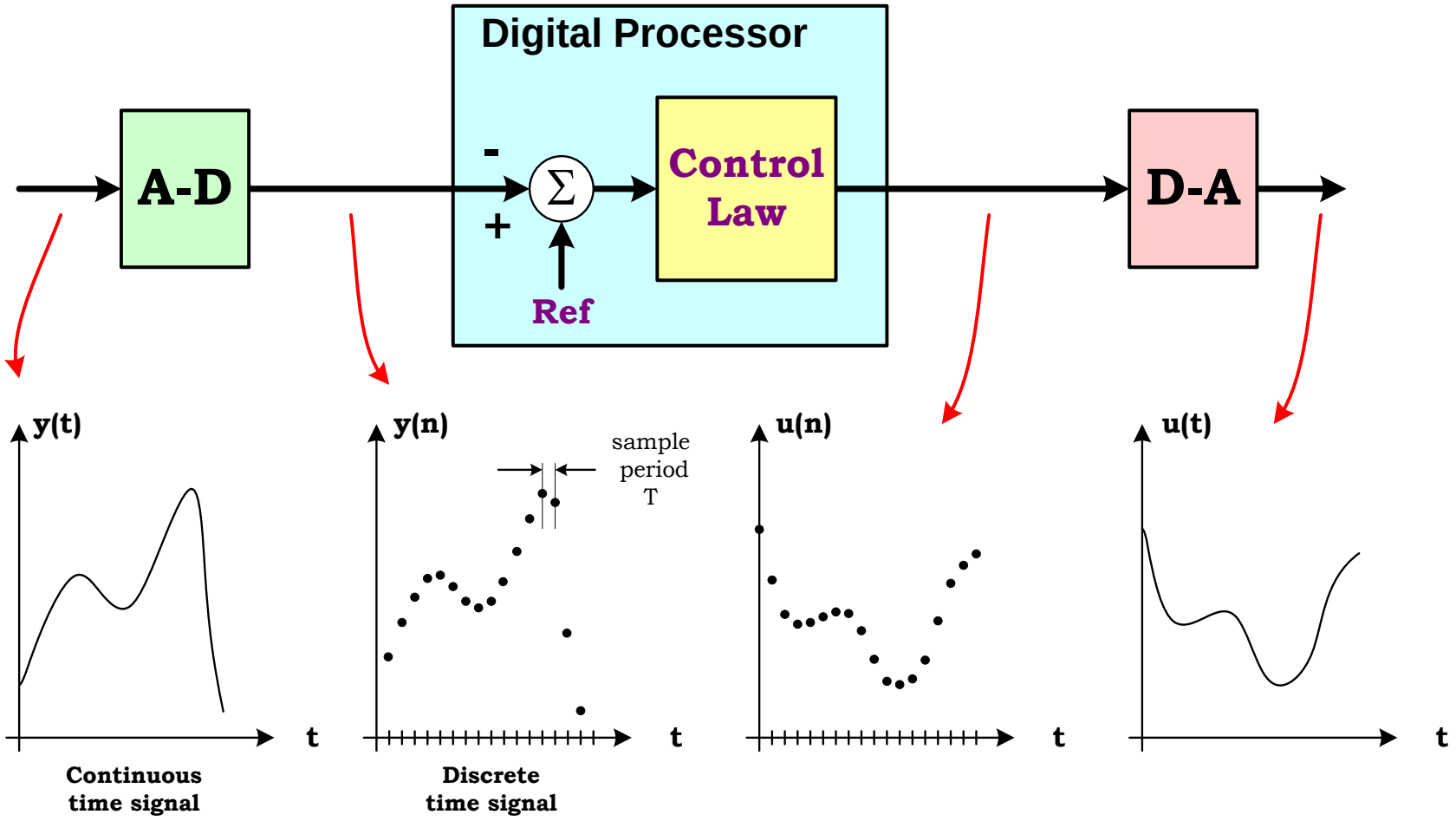
Differential equations
1st, 2nd, 3rd,...order

OR

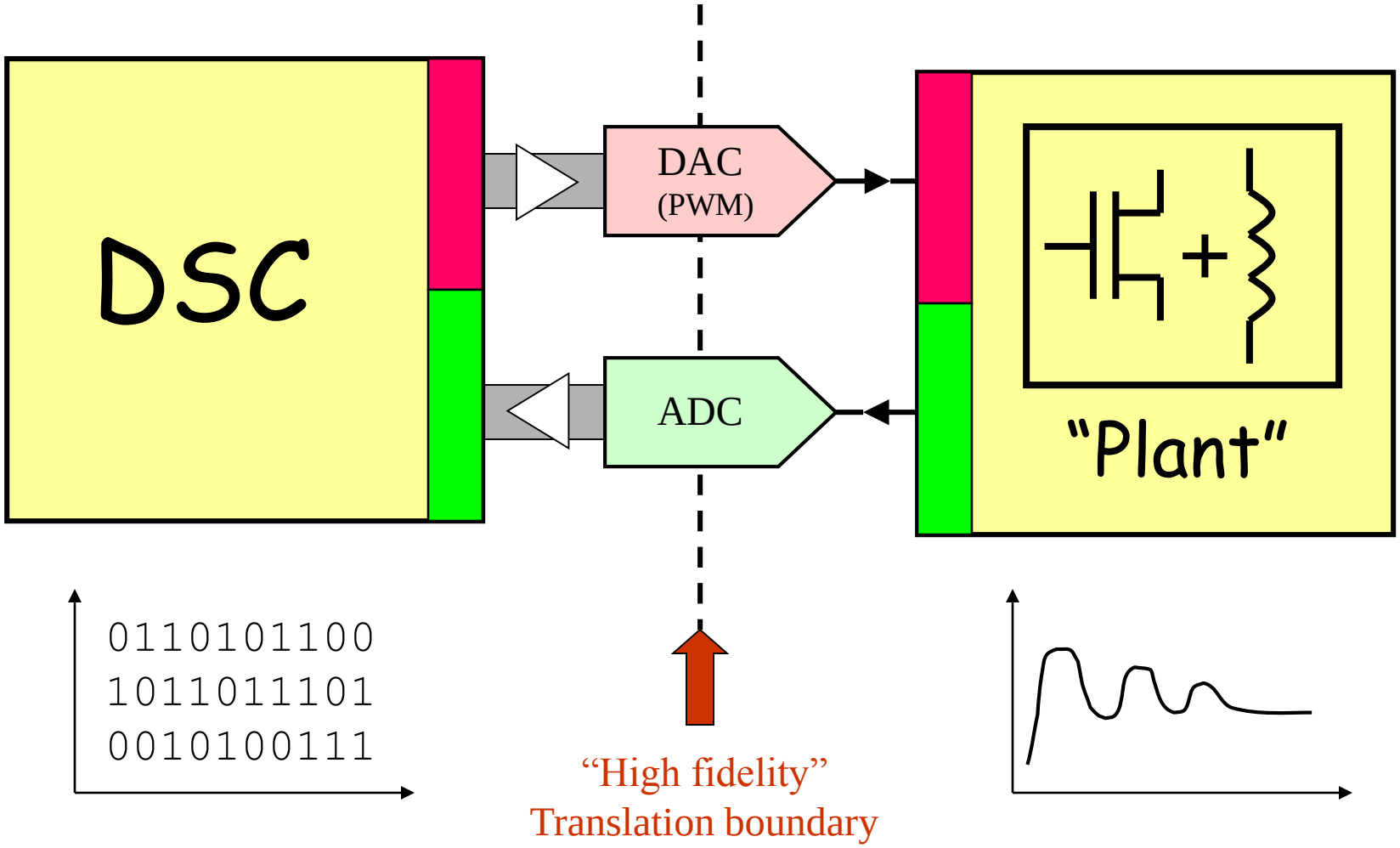
Laplace Transform

Z Transform

Time Sampled Systems

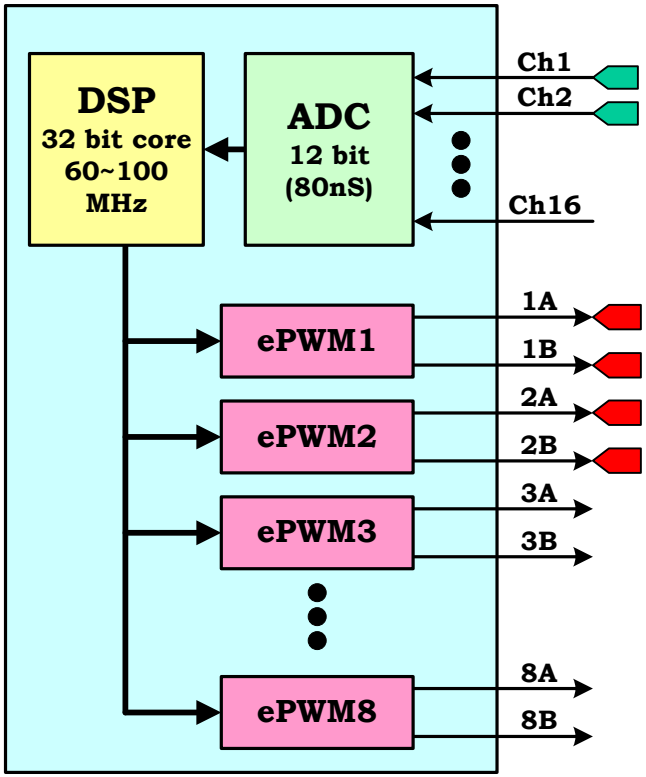


Digitally Controlled Power Supply

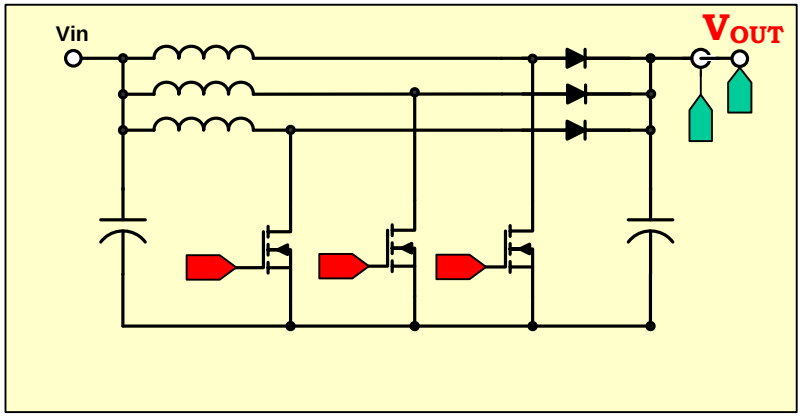


System Mapping

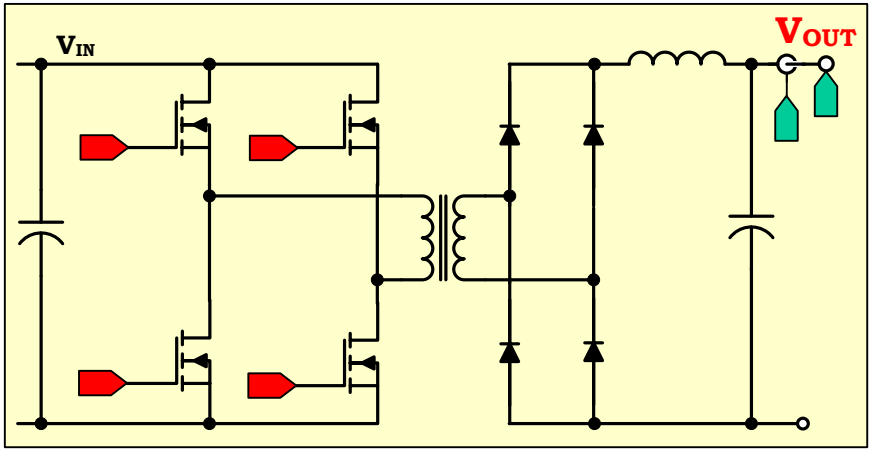
F280xx



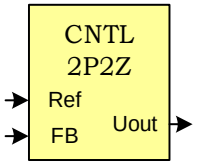
PFC – 3ph Interleaved



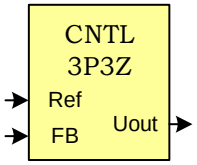
Phase-Shifted Full Bridge



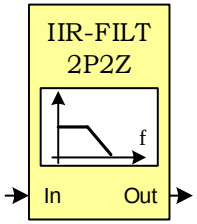
Software Library Approach



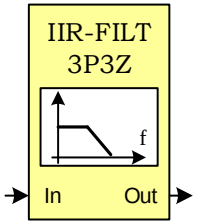
Control 2-pole / 2-zero



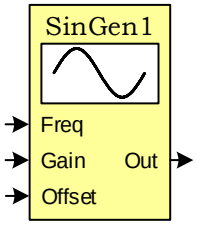
Control 3-pole / 3-zero



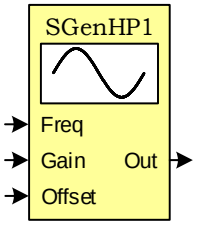
2nd order IIR filter



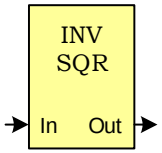
3rd order IIR filter



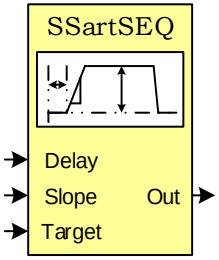
Sine Wave generator



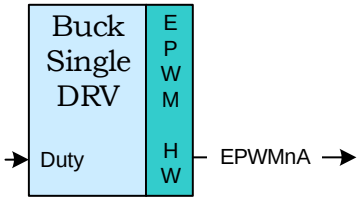
High precision Sine Gen



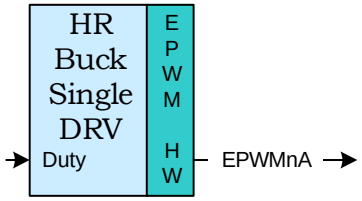
Inverse Square function



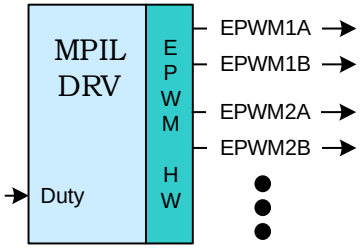
Soft Start and Sequencing



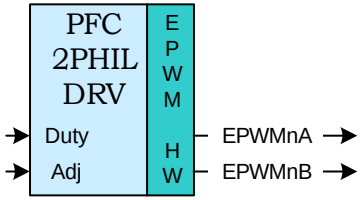
Buck Single Output



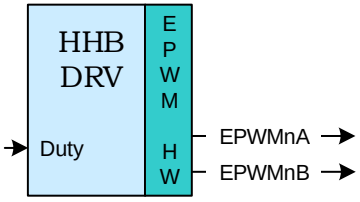
High Resolution Buck



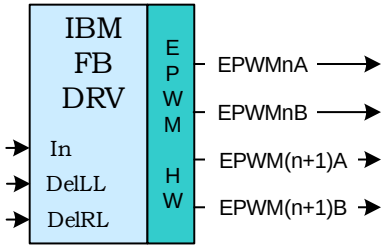
Multi-Phase Interleaved



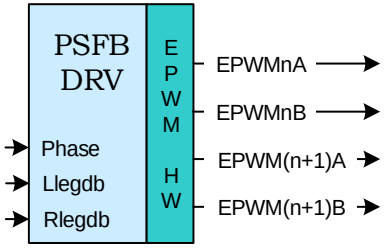
Power Factor 2-phase Interleaved



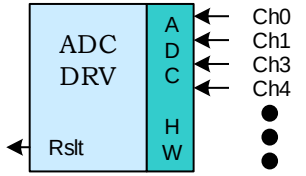
Half H-Bridge



IBM method Full Bridge



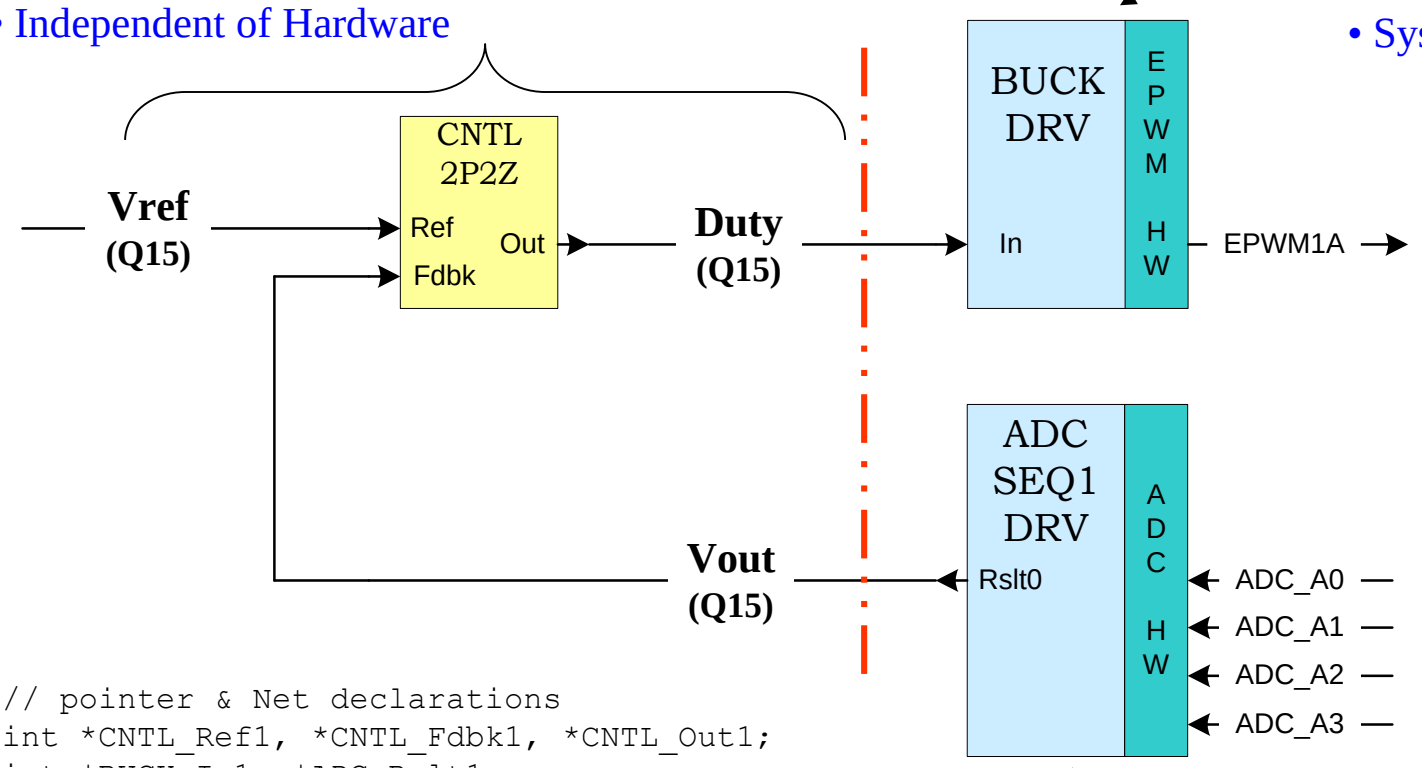
Phase Shifted Full Bridge



Analog-Digital Converter driver

Peripheral Drivers

CPU dependency only:
- Math / algorithms
- Per-Unit math (0-100%)
- Independent of Hardware

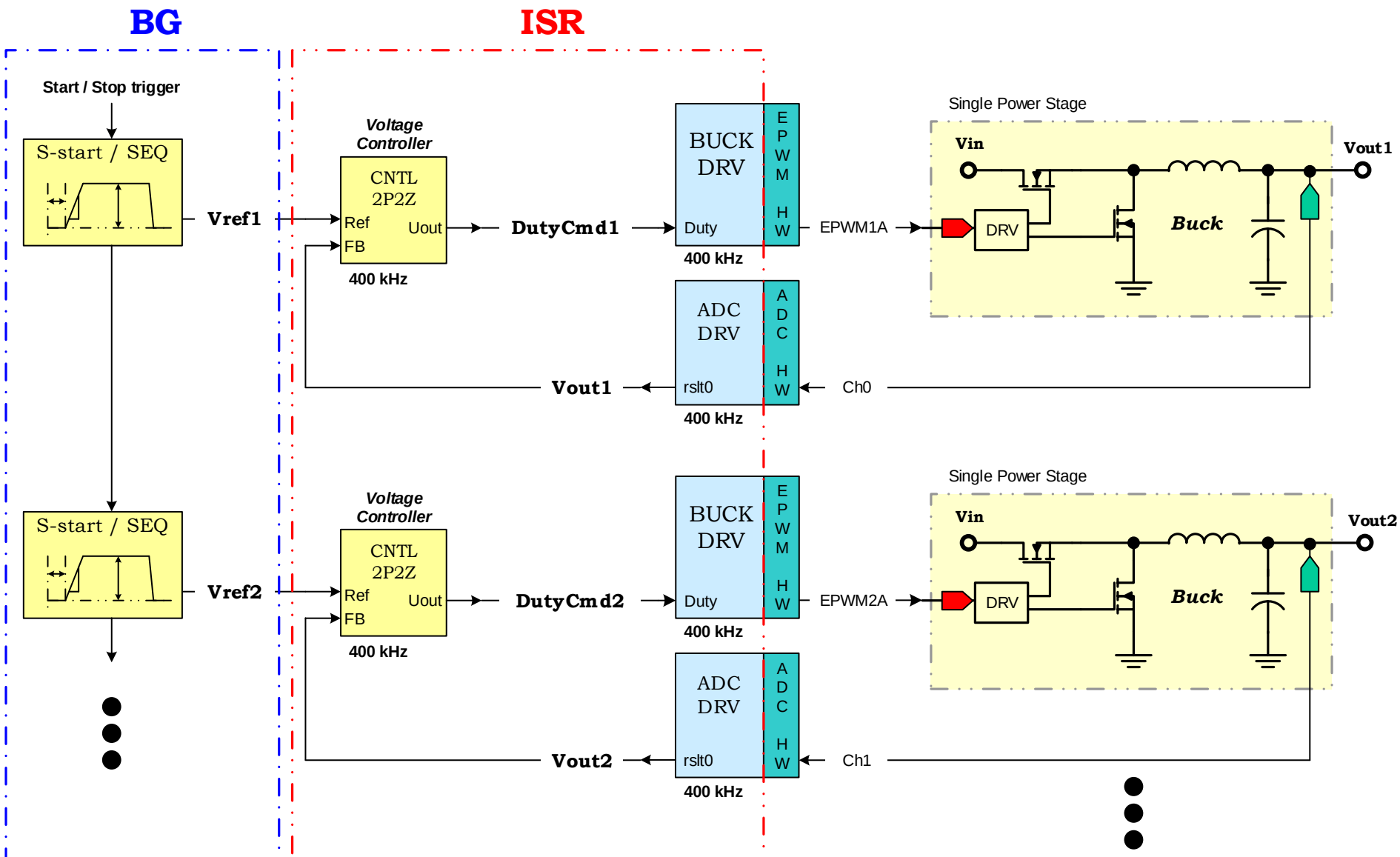


Depends on:
- PWM frequency
- System clock frequency

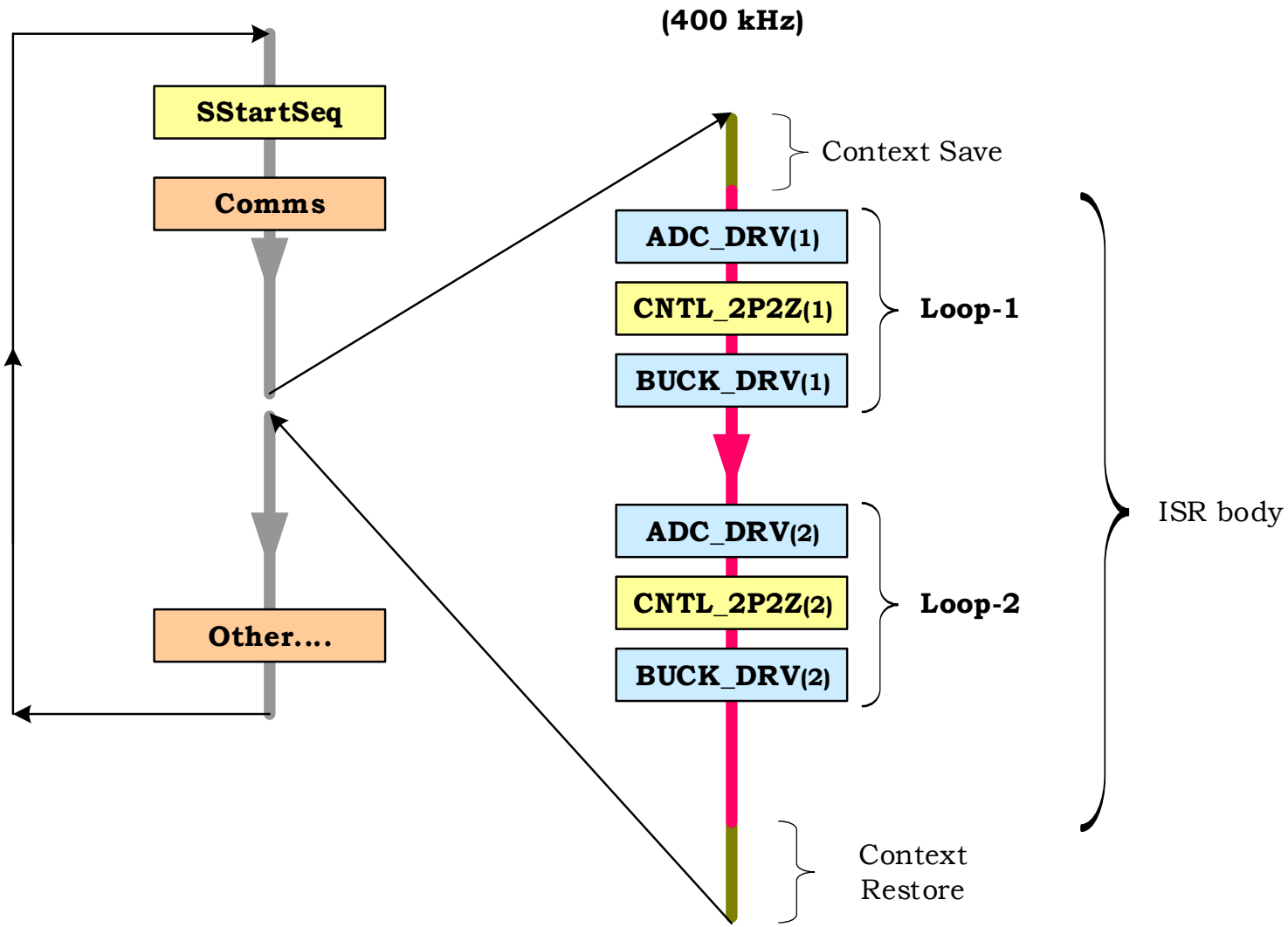
```
// pointer & Net declarations
int *CNTL_Ref1, *CNTL_Fdbk1, *CNTL_Out1;
int *BUCK_In1, *ADC_Rslt1;
int Vref, Duty, Vout;

// "connect" the modules
CNTL_Ref1 = &Vref;
CNTL_Out1 = &Duty; BUCK_In1 = &Duty;
CNTL_Fdbk1 = &Vout; ADC_Rslt1 = &Vout;
```

Depends on:
- # ADC bits (10 / 12 ?)
- Unipolar, Bipolar ?
- Offset ?

Dual Buck Example



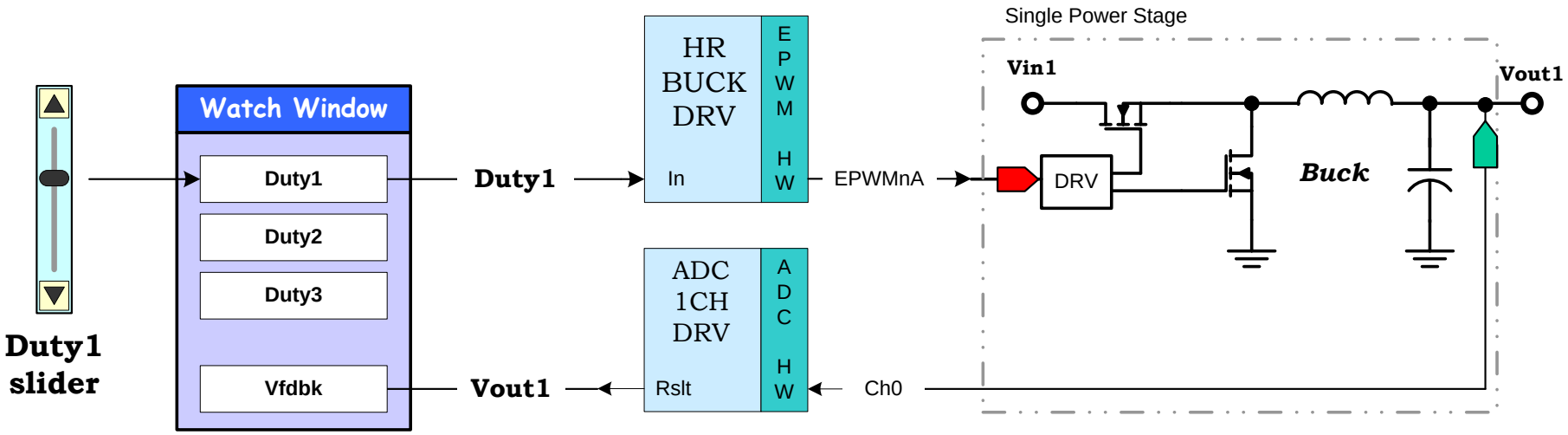
Software Block Execution



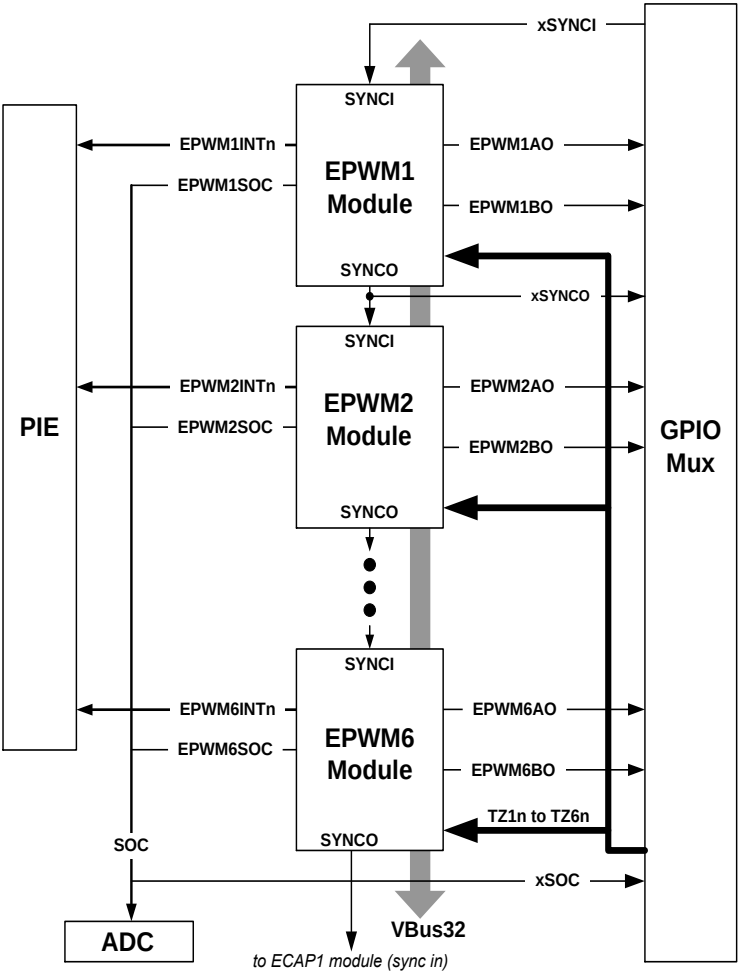
Driving the Power Stage with PWM Waveforms

- Open-Loop System Block Diagram
- Generating PWM using the ePWM Module
- Power Stage Topologies and Software Library Support

Simple Open-Loop Diagram

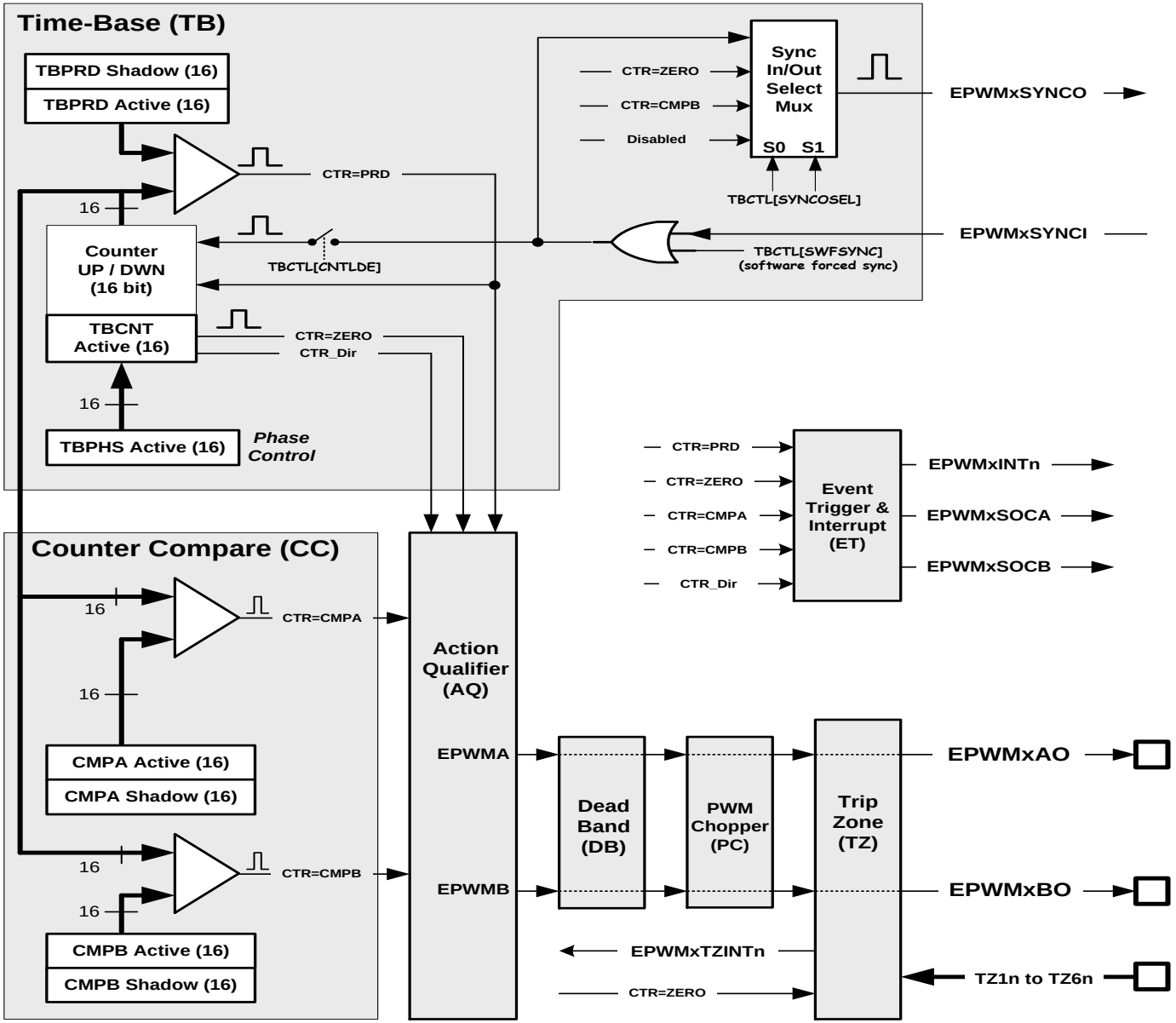


Scaleable PWM Peripherals

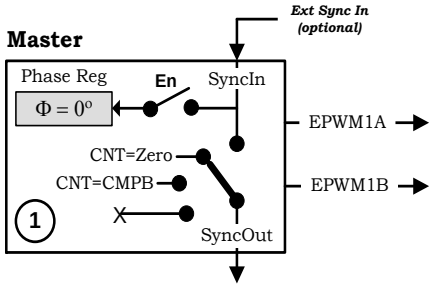


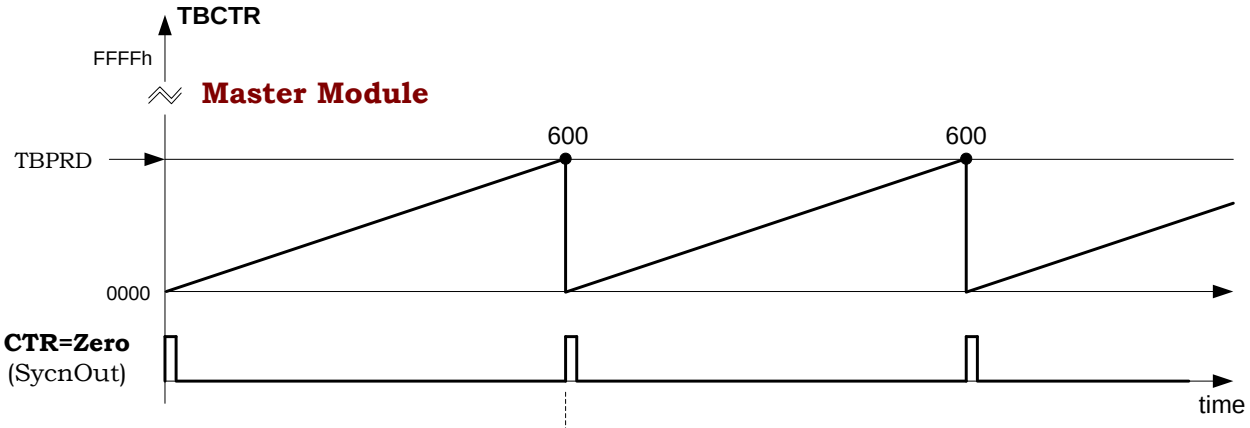
- Resources allocated on a per channel basis
- Each channel (module) supports 2 independent PWM outputs (A&B)
- # Channels easily scaleable – software reuse
- Time-base synch feature for all channels
- 6 modules (12 PWM outputs) on F2808
- Key features:
  - Phase & edge control
  - New counting modes
  - Independent deadband
  - Flexible trip-zones
  - High frequency chopper mode

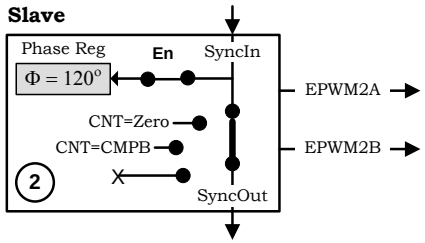
ePWM Module Block Diagram

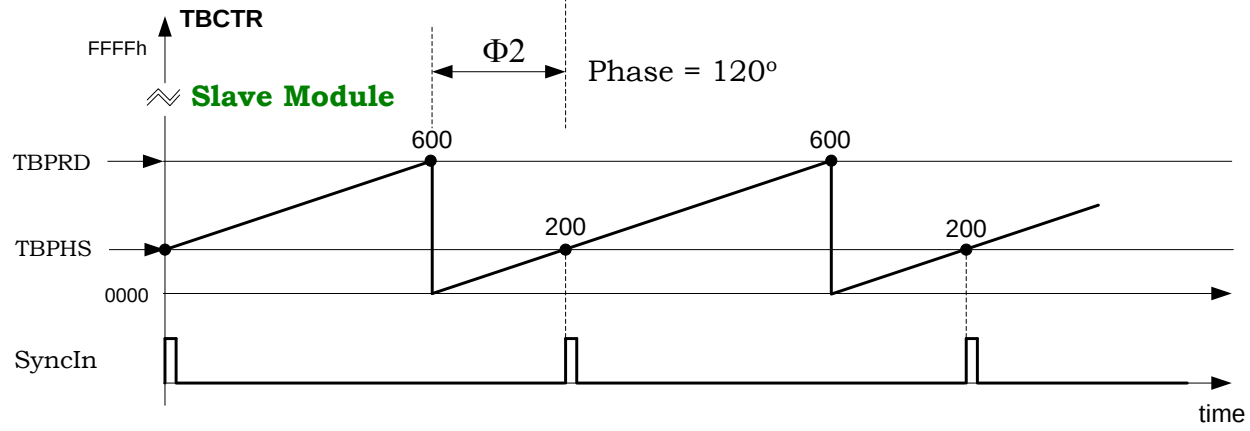


Module Sync and Phase Control





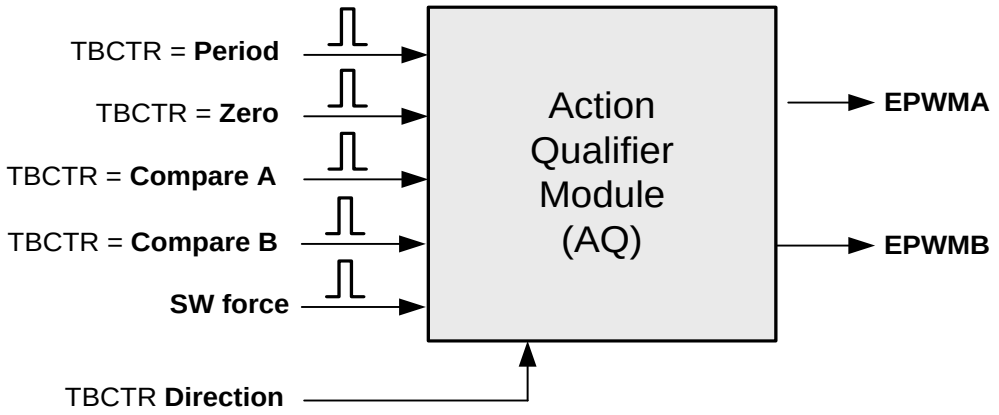




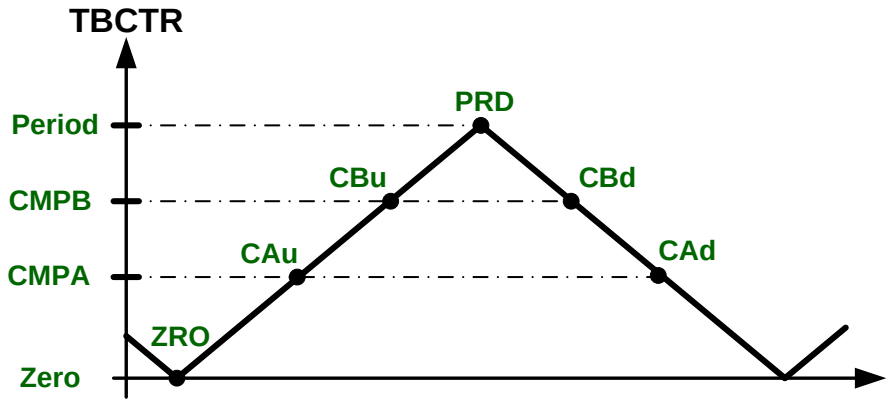
Action Qualifier Module (AQ)

Key Features

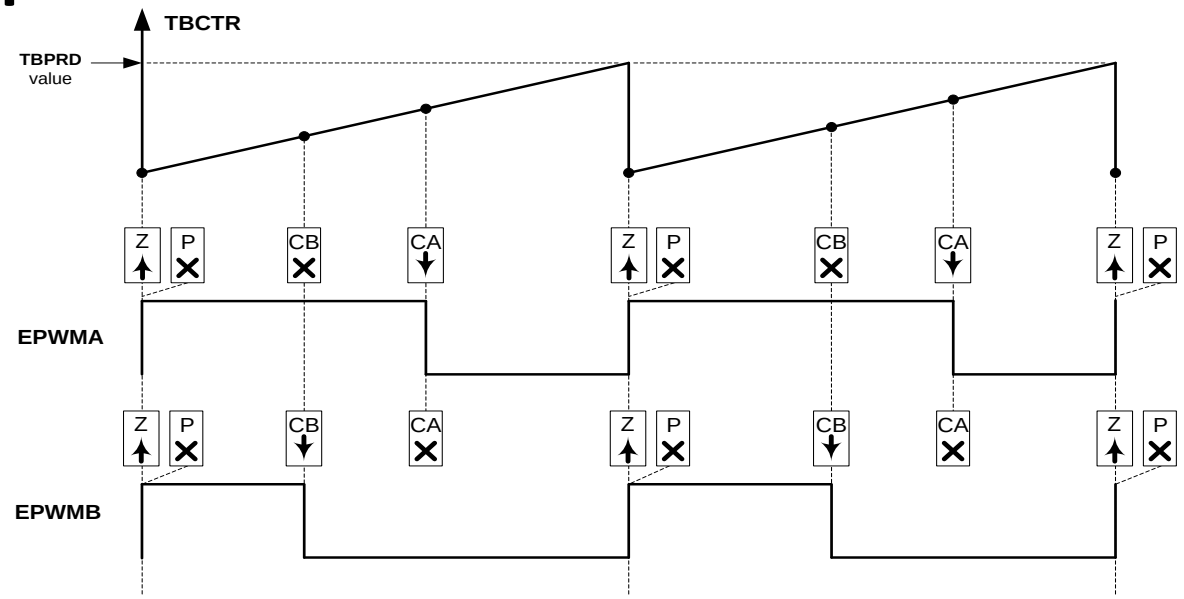
- Multi event driven waveform generator
- Events drive outputs A and B independently.
- Full control on waveform polarity
- Full transparency on waveform construction
- S/W forcing events supported
- All events can generate interrupts & ADC SOC

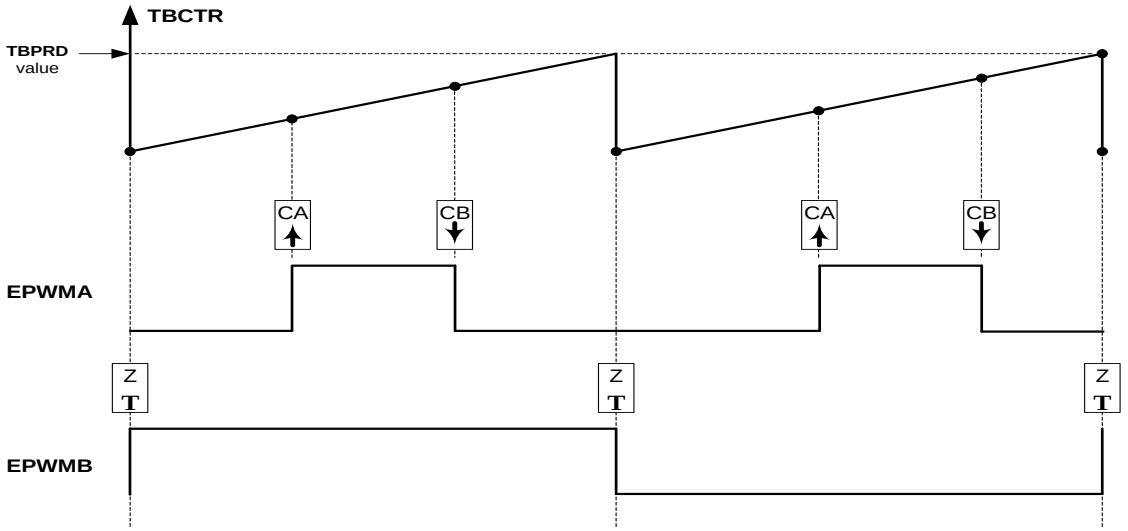


Events		Actions			
		Nothing	Clear Lo	Set Hi	Toggle
Zero (ZRO)		Z ✕	Z ↓	Z ↑	Z T
TBCTR (Up) equals:	CMPA (CAu)	CA ✕	CA ↓	CA ↑	CA T
	CMPB (CBu)	CB ✕	CB ↓	CB ↑	CB T
Period (PRD)		P ✕	P ↓	P ↑	P T
TBCTR (Down) equals:	CMPA (CAd)	CA ✕	CA ↓	CA ↑	CA T
	CMPB (CBd)	CB ✕	CB ↓	CB ↑	CB T
S/W force		SW ✕	SW ↓	SW ↑	SW T

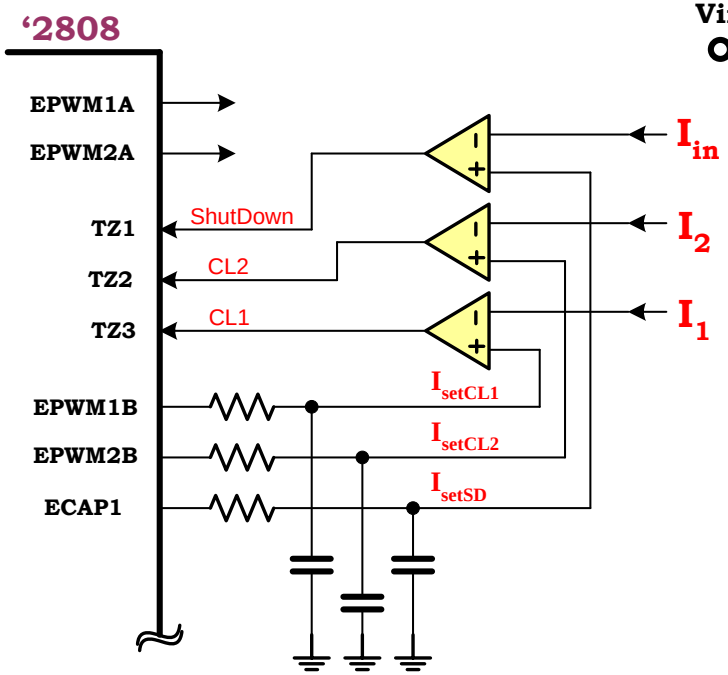


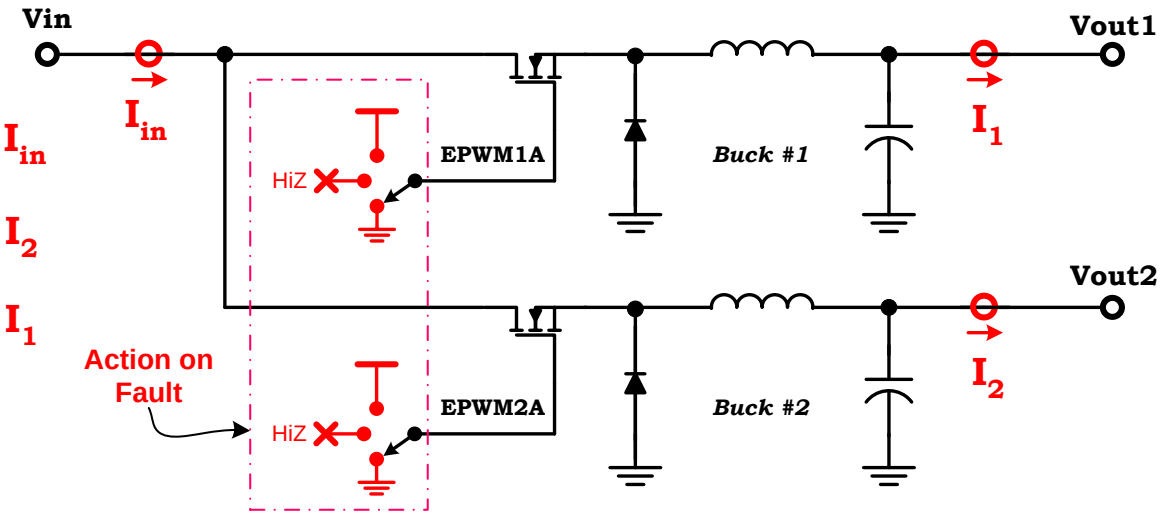
Simple Waveform Construction

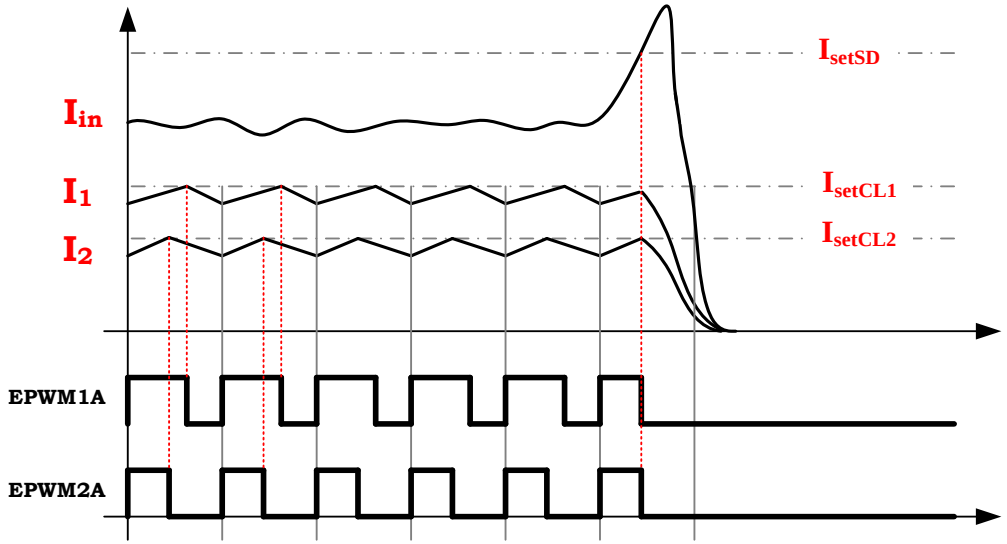




Fault Management Support







Trip Zones:

6 independent zones (TZ1~TZ6)

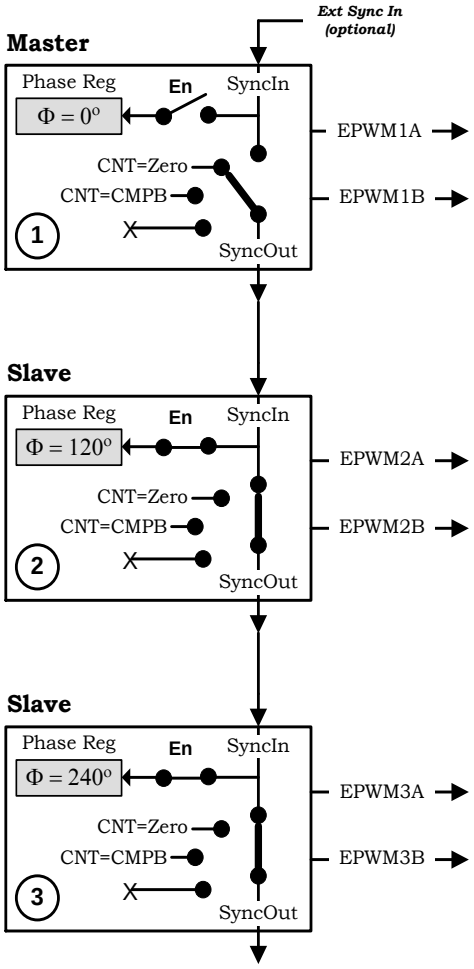
Force High, Low or HiZ on trip

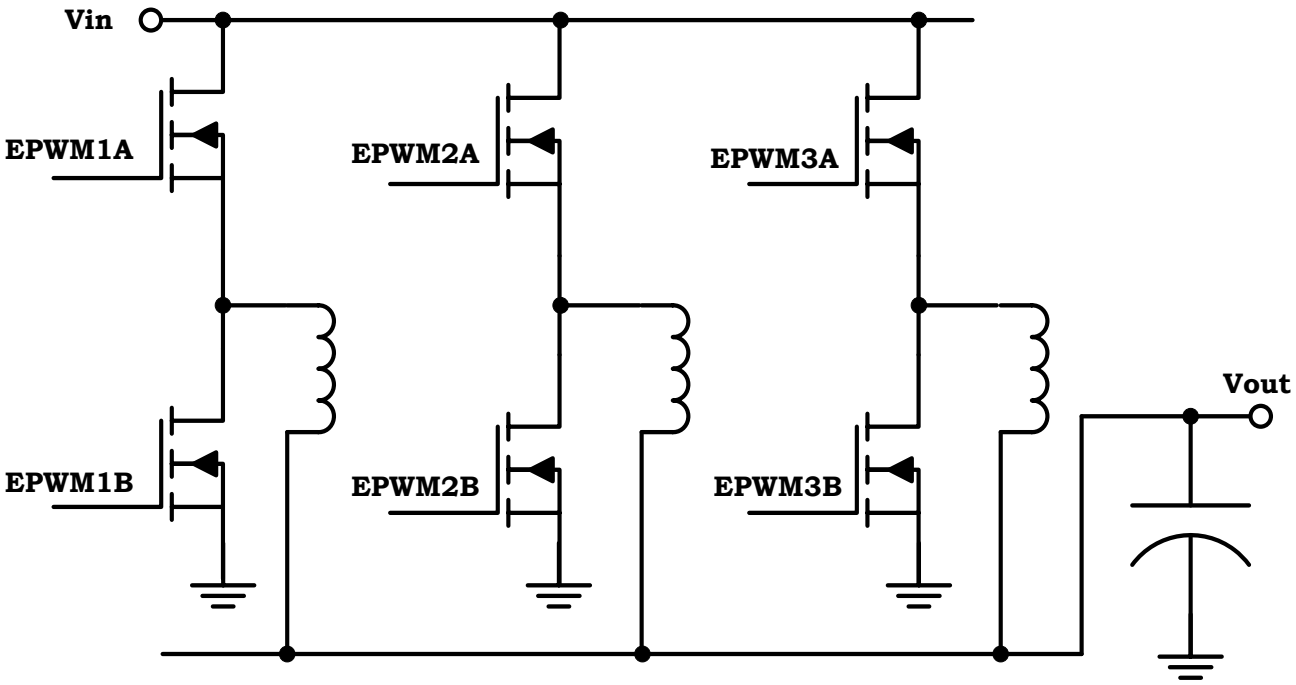
One-time trip → catastrophic failure

Cycle-by-cycle → current limit mode

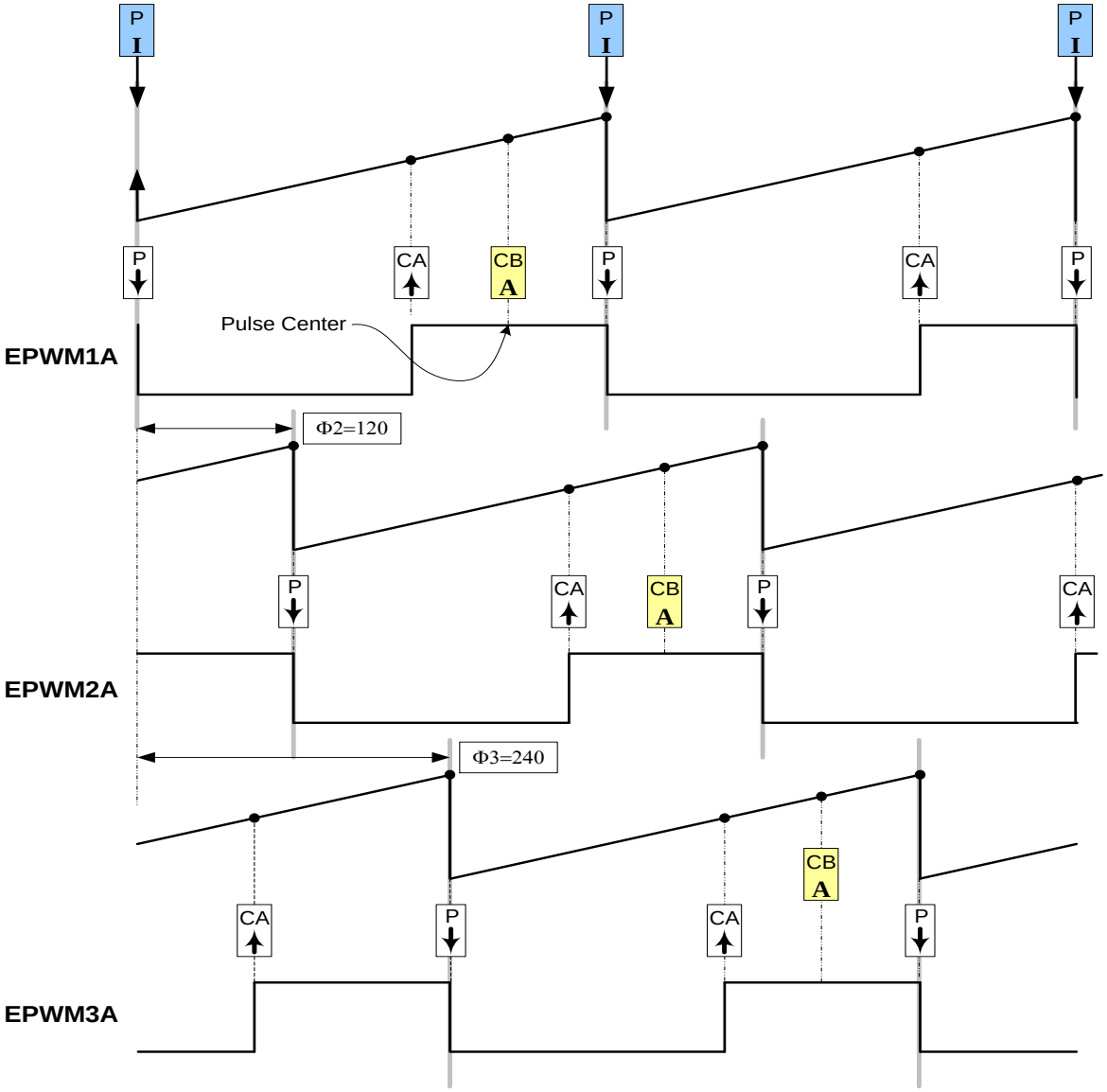
TZ1~TZ6 can trigger interrupt

Multi-Phase Interleaved (MPI)





Switching Requirements – MPI



- Asymmetrical PWM case
- Complementary output generated by dead-band unit
- CMPB triggers ADC SOC

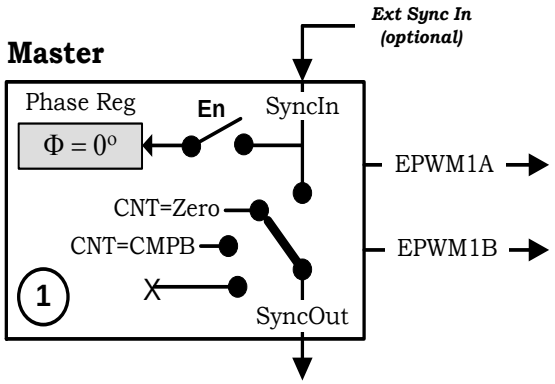
INIT-time

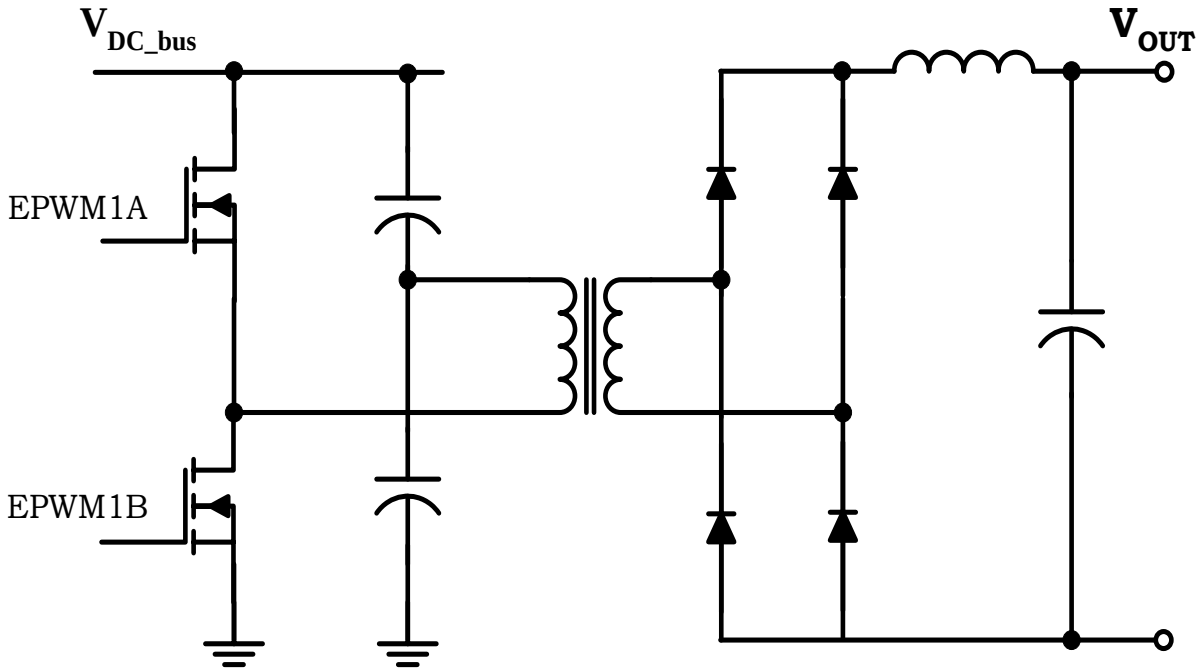
- Period (1,2,3)
- CAu Action (1,2,3)
- PRD Action (1,2,3)
- Phase (2,3)
- PRD Interrupt (1)
- CBu ADC SOC (1,2,3)
- Dead-band

RUN-time

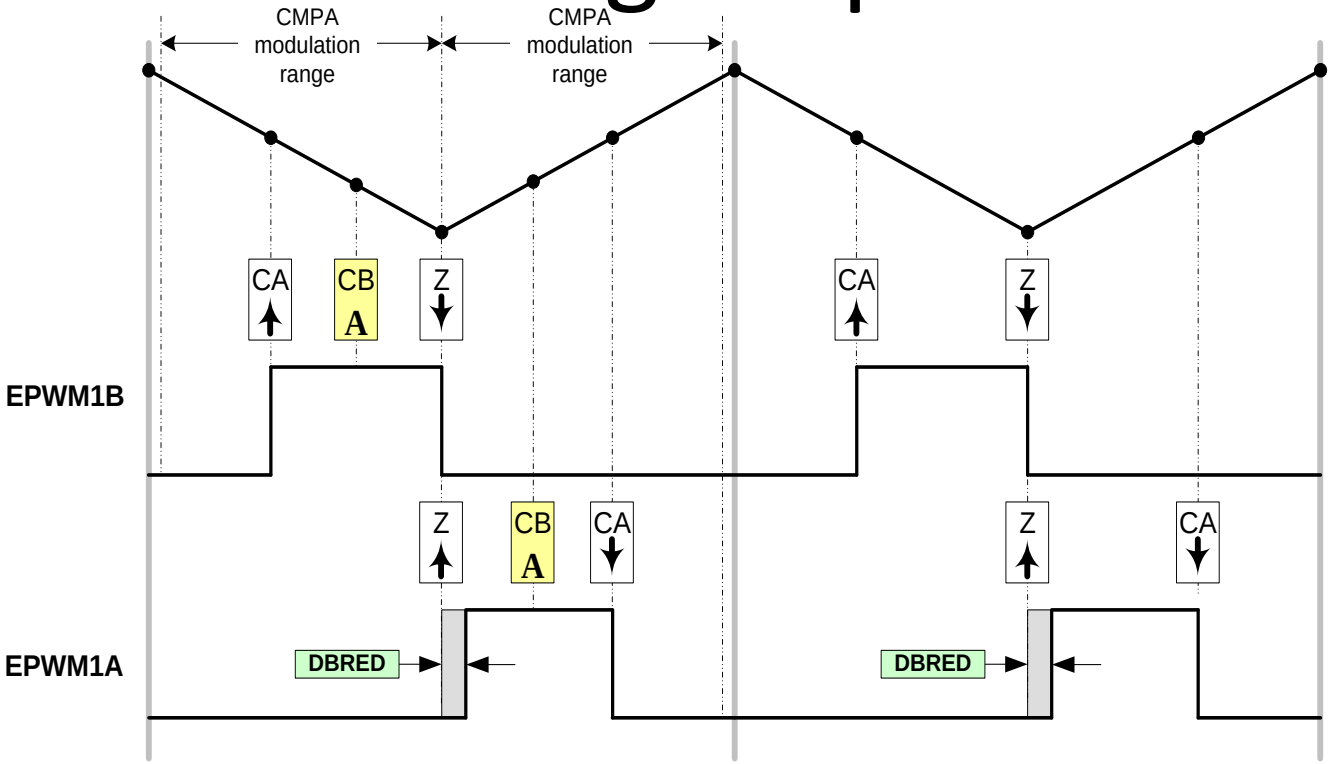
- CMPA (1,2,3)
- CMPB (1,2,3)

Half H-Bridge (HHB)





Switching Requirements – HHB



- Up/Down Count
- Asymmetrical PWM
- dead-band on A only
- 50 % max Modulation (controlled by CMPA)

INIT-time

- ZRO Action (A,B)
- CAd Action
- CAu Action
- CBd ADC trigger
- CBd ADC trigger
- DBRED

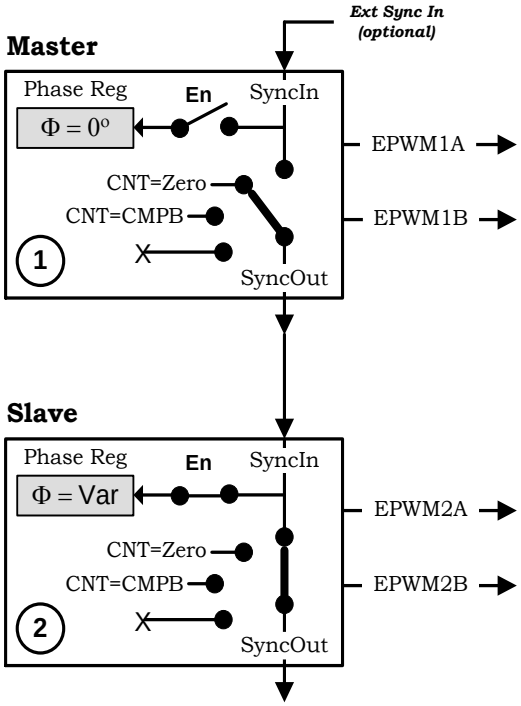
RUN-time

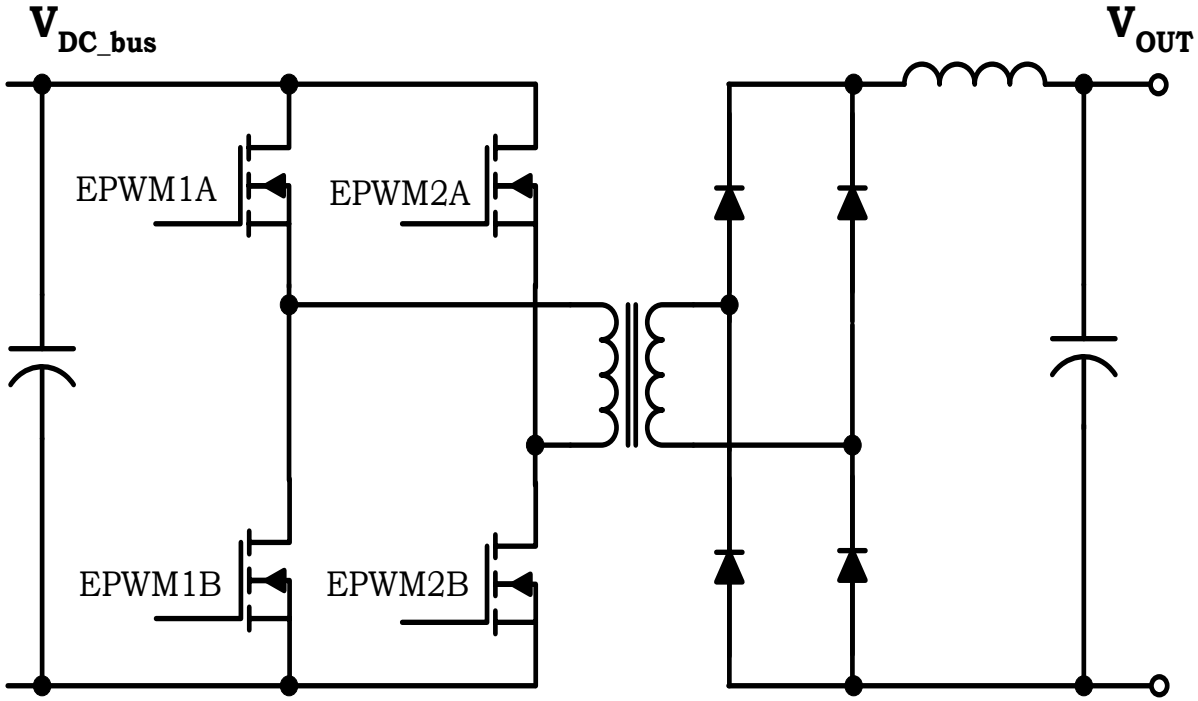
- CMPA
- CMPB (optional)

Compare A modulation range:

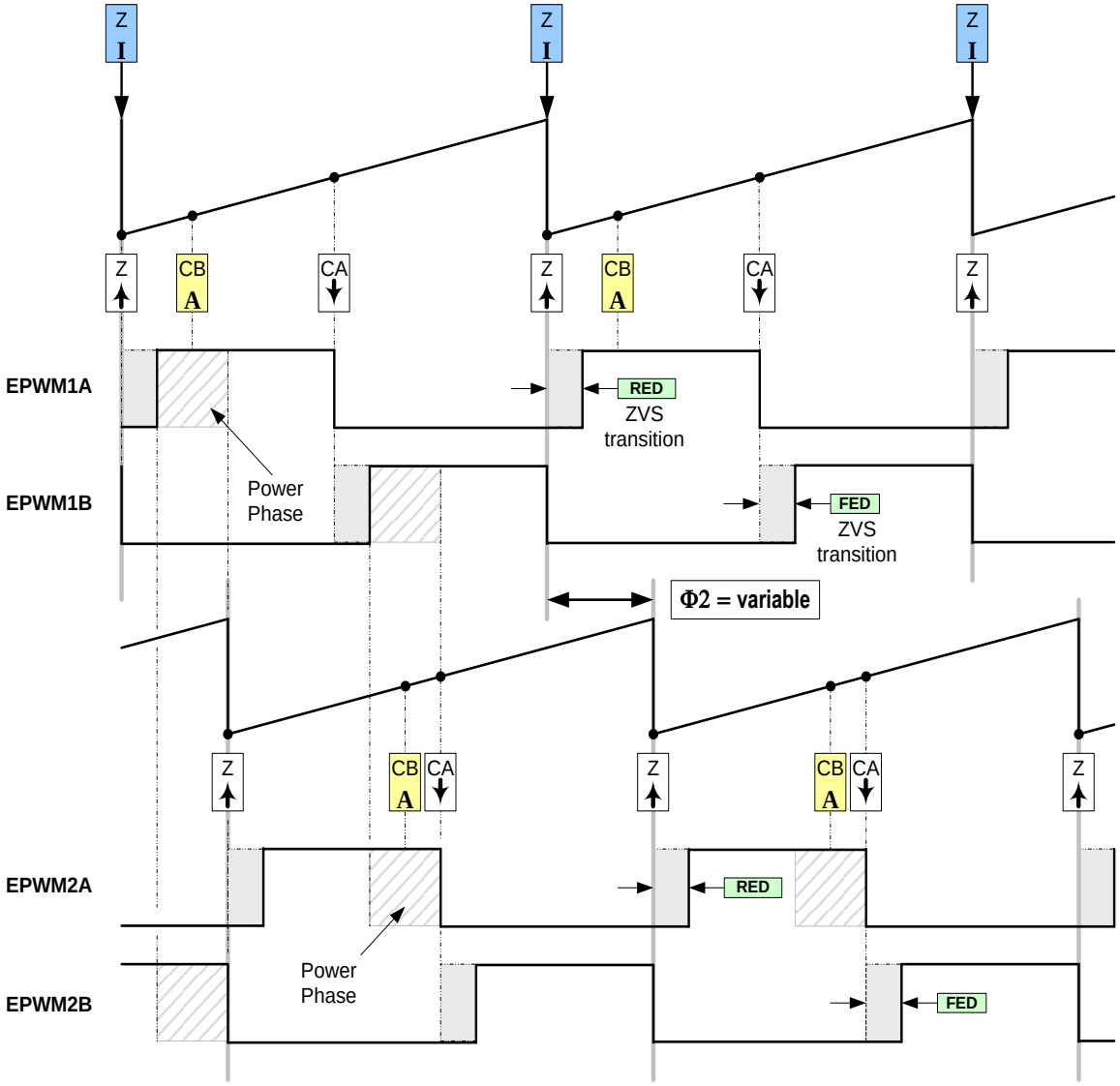
$0 < CMPA < (PRD – ½ \times DBRED)$

Phase Shifted Full Bridge (PSFB)





Switching Requirements – PSFB



- Asymmetrical PWM
- Using dead-band module
- Phase (Φ) is the control variable
- Duty fixed at ~ 50%
- RED / FED control ZVS trans.
  i.e. via resonance
- CMPB can trigger ADC SOC

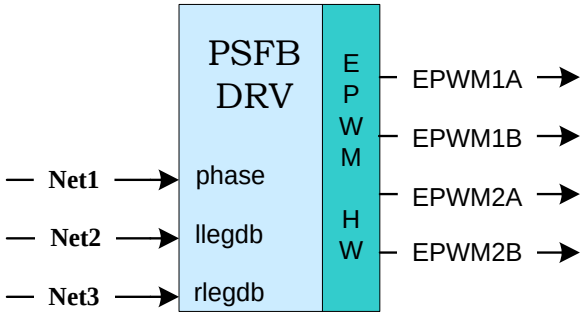
INIT-time

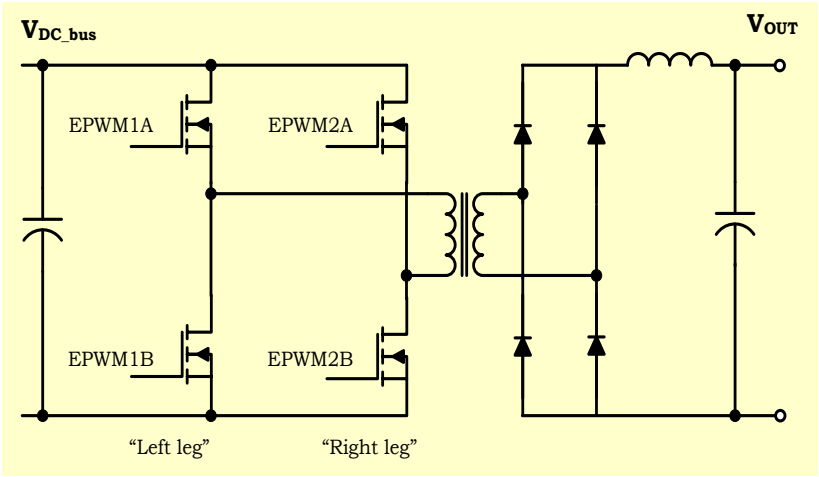
- Period (1,2)
- CMPA (1,2) ~ 50%
- CAu action (1,2)
- ZRO action (1,2)
- CBu trigger for ADC SOC

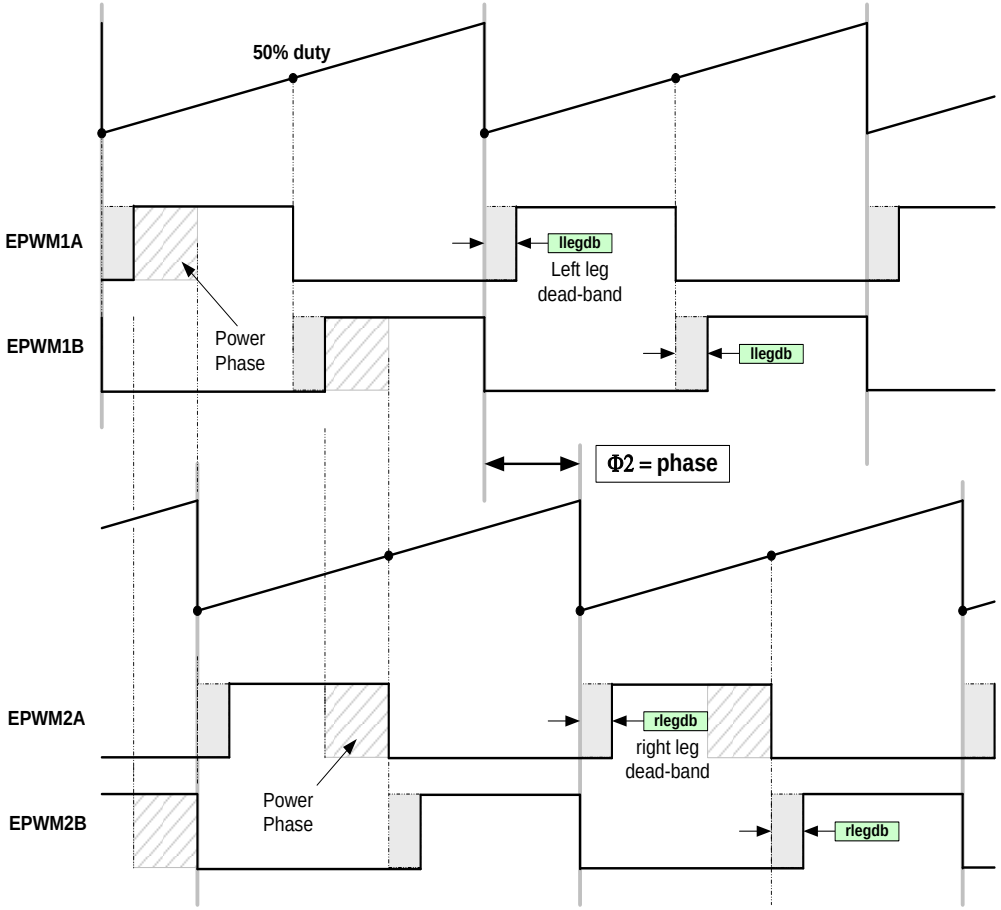
RUN-time

- Phase (2) – every cycle
- FED / RED (1,2) – slow loop

Software Driver Module – PSFB







Software Driver Module – PFC2PHIL