

Reconfigurable Grid? FPGAs Versus DSPs for Power Electronics



Overview

▪ FPGA Applications in Power Electronics

- Detailed technical comparison to DSPs
- Control examples: Multilevel inverters, EMI reduction, Real-Time HIL Simulation

▪ VHDL Development Challenges

- Data on development cost for full custom design
- New data on graphical system design

▪ Designing at a higher level

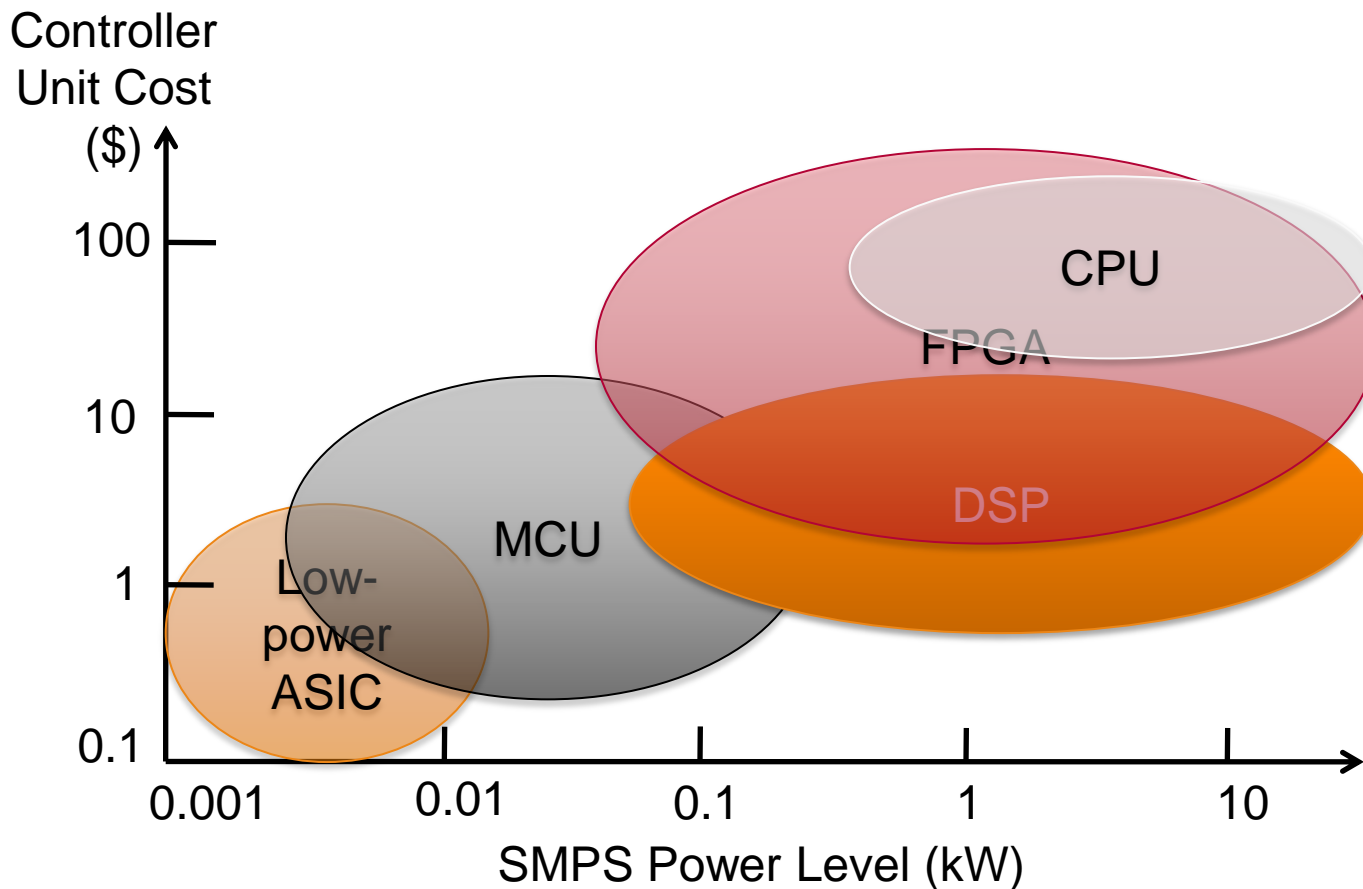
- New tools, new methodologies
- **Demonstrations!**
- Introducing the NI General Purpose Inverter Controller
- IP libraries available



Power Electronics Design Goals & Tradeoffs

- **Optimize for multiple design goals simultaneously, including:**
 - Energy efficiency
 - Cost
 - Component lifetime
 - Systematic reliability
 - Regulatory compliance
 - Smart grid ready
 - Differentiated features
 - ...
 - ...

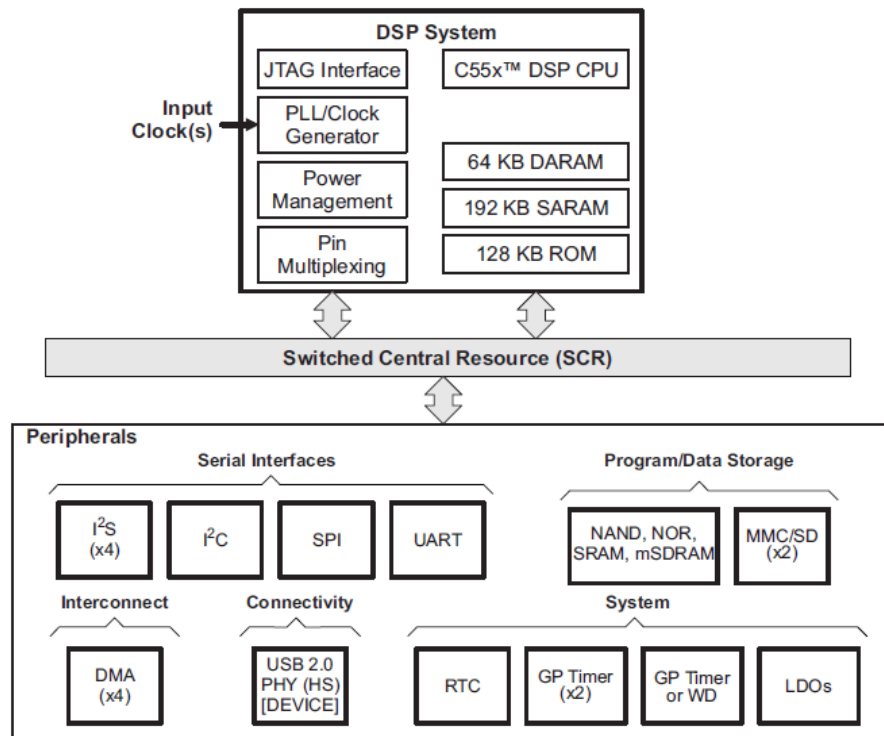
Spectrum of suitable control targets for high volume mass production



DSP and FPGA

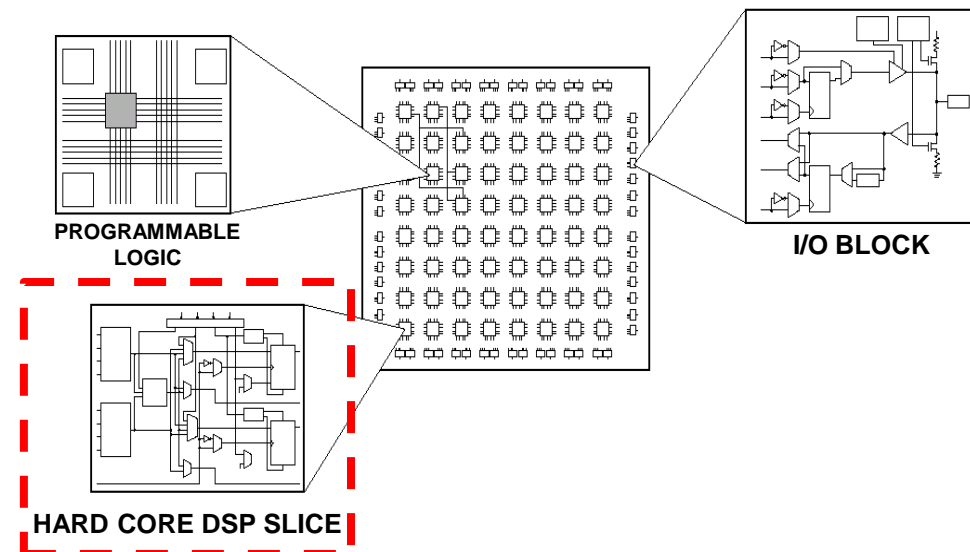
■ DSP

- Digital Signal Processor
- Special **Processor** optimized for fast operation

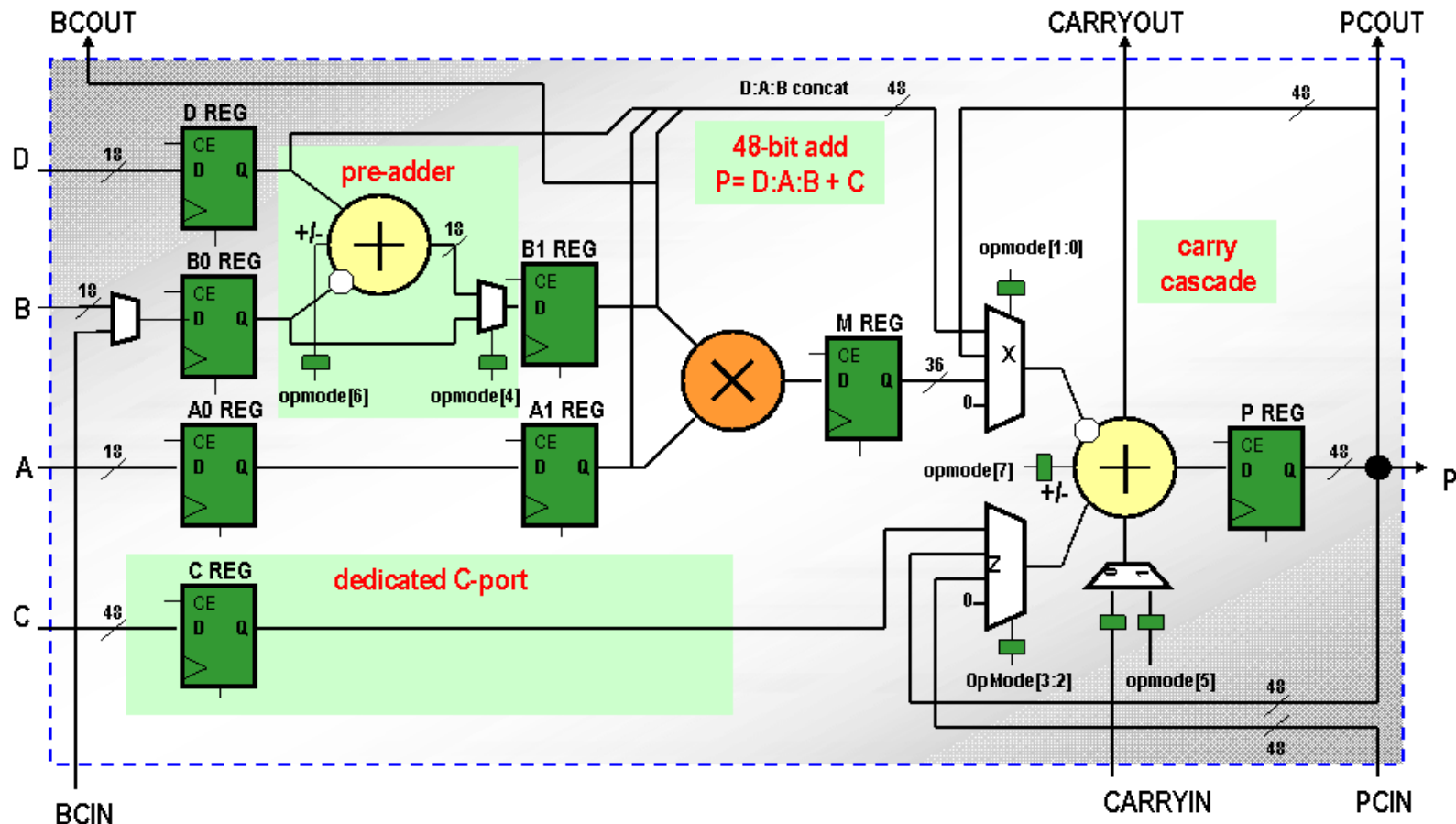


■ FPGA

- Field Programmable Gate Array
- Flexible hardware architecture with logic functions and DSP blocks



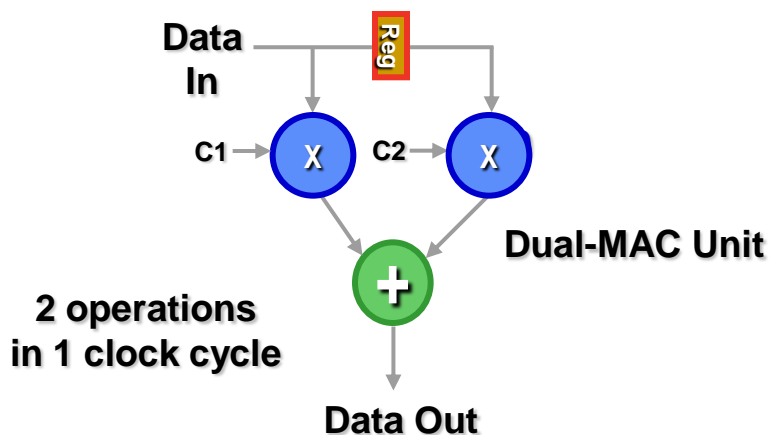
DSP Cores Embedded in FPGA FABRIC



- Upto 58 DSP48A1 slices embedded in a Spartan 6 LX45 FPGA
 - 250MHz implementation
 - Fast 18-bit multiplier & 48-bit adder
 - ASIC-like performance

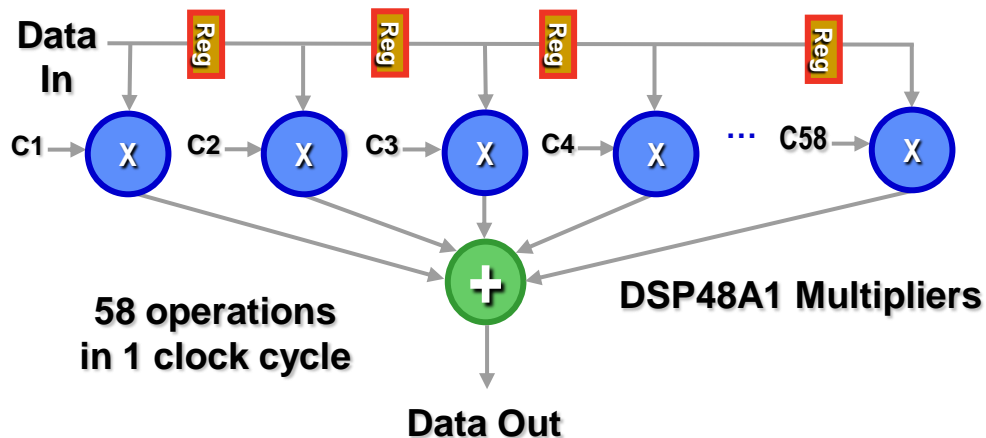
Modern FPGAs utilize hard-core DSP processing elements integrated in fabric

Standard DSP Processor (Dual Core DSP)



$$\frac{300 \text{ MHz} * 2 \text{ Cores}}{1 \text{ clock cycle}} = 600 \text{ MMACS}$$

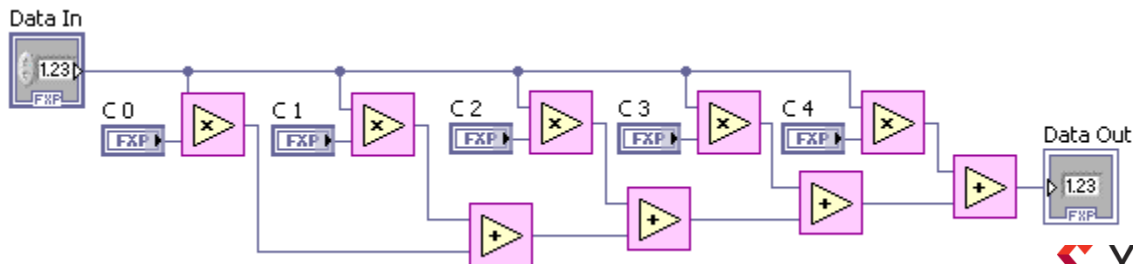
FPGA - Fully Parallel Implementation (Spartan-6 LX45 FPGA)



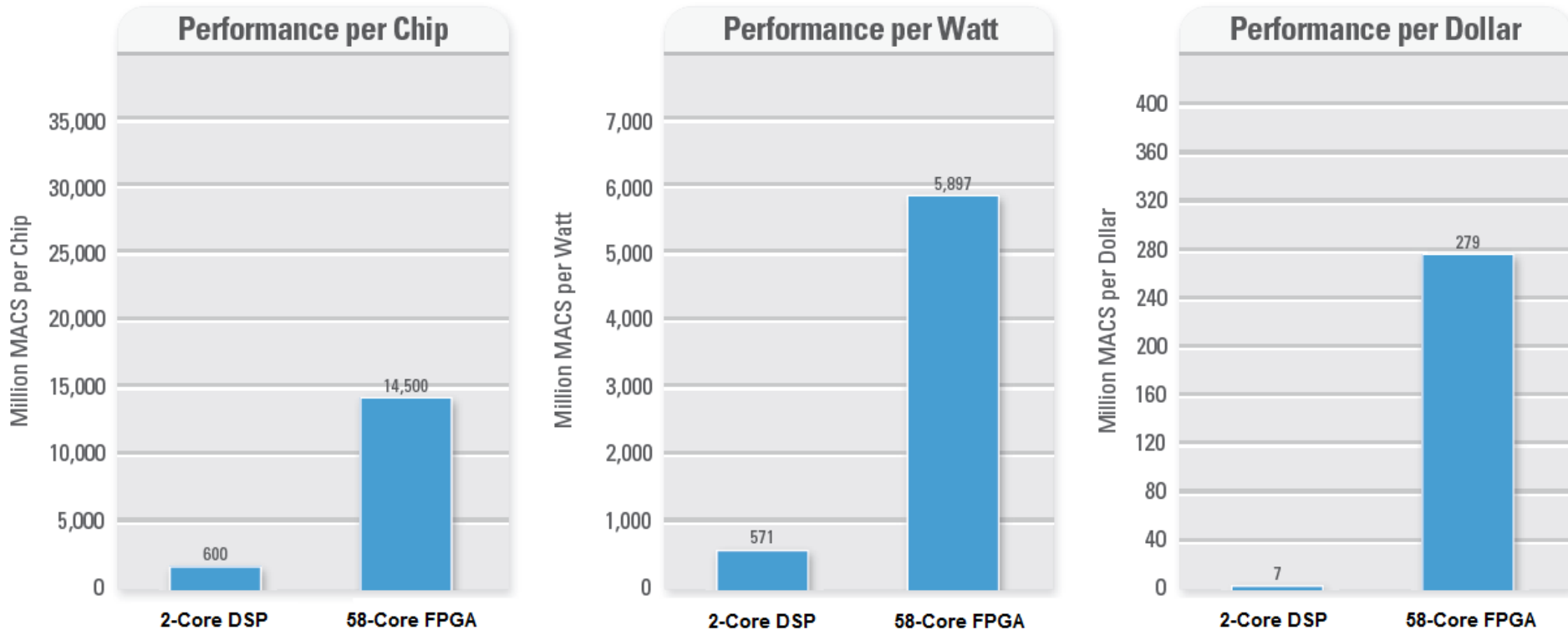
$$\frac{250 \text{ MHz} * 58 \text{ Cores}}{1 \text{ clock cycle}} = 14,500 \text{ MMACS}$$

NI LabVIEW FPGA

- High Level Programming
- True Parallel Execution



Dual-Core DSP versus Spartan-6 FPGA



	Dual-Core DSP	Spartan-6 LX45 FPGA	Performance Ratio (FPGA/DSP)
Million MACS per Chip	600	14,500	24
Million MACS per Watt	571	5,897	10
Million MACS per Dollar	7	279	40

FPGA versus DSP for Power Electronics

Features	DSPs	FPGA
Performance	600 MMACS	14,500 MMACS
Peripheral	Fixed	Flexible
Processor cores	Single (or Double)	Multi
Modulation Scheme	Fixed PWM	Flexible
Loop Response Time	~50us	~2.5 us
Silicon Gate Level (SGL) Reconfigurability	-	✓
Design Environment	C (Higher level of Abstract)	VHDL & Verilog (RTL)

FPGA Pros/Cons

– Pros

- Nanosecond control of timing, completely parallel execution
- Now contain dozens to hundreds of mini hard-core DSPs, emerging FPGAs have hard-core floating point processors
- Orders of magnitude higher DSP performance per dollar and per watt compared to single core DSPs
- Field reconfigurable at the silicon gate level (SGL)

– Cons

- Orders of magnitude more complex to program
- PCB layout is becoming increasingly complex and costly due to fine-pitch parts
- 70% of software development cost is I/O interface development (rather than algorithms)

FPGA based control reduces Total Cost of Ownership

Increase power efficiency by advanced modulation schemes
(Space Vector Modulation, RPFM)

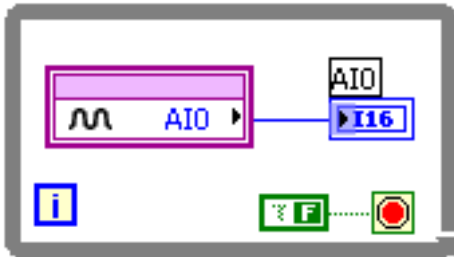
Reduce board BOM by integration of industrial networking

FPGA-based control offers more than 15x performance

Loop bandwidth of uC based ASSPs is ~55usec

FPGA-based Field Oriented Control easily achieve less than
2.5usec

LabVIEW FPGA vs



-- © 2003, 2005, 2006 National Instruments Corporation

```
library(MASS)
use.mass.sld<-1
use.mass.sld<-1

-- release 1

cov11 and so
geom11 ("
  WIDTH OUT
  WIDTH IN "
```

```

3 |
port(
  otk
  read
  min_W : 1
  min_out :
  min_oth
  min_oth
  min_oth
  3 |
  out min_W;

```

```

module.exports = {
  type: 'state',
  signal: 'Shift',
  signal: 'Click',
  signal: 'relat',
  signal: 'Shift',
  signal: 'data',
  signal: 'Clear',
  signal: 'Knob1',
  signal: 'Knob1',
  signal: 'vib'
}

```

```

sigval min_0
attribute op
attribute op
begin
    min_out[16] <
    min_out[0]
    ClearKradle
    KradleIn
    min_out <

```

```
OutputProc :
process fork,
begin
  if reset =
    CntCount
  elsif not
    if (Clea
      CntCou
```

3

```

else
    if c1 == c2
        c1 == c2
    else
        c1 == c2
    and if;
    and if;
    and if;
    and process;

```

```
process(ok, a)
begin
  if total = 0
    WriteCoord
    WriteCoord
    data
    WriteCoord
    ok out
    all done
  else rising
    if clearW
      WriteCo
```

[illegible]

```

end IF
IF (N)
  WRITE
  WRITE
end IF
IF (N)
  GOTO
  GOTO
end IF
IF (N)
  GOTO
  GOTO

```

```

    11 2
    12.
    13.
    14.
    15.
    16.
    17.
    18.
    19.
    20.
    21.
    22.
    23.
    24.
    25.
    26.
    27.
    28.
    29.
    30.
    31.
    32.
    33.
    34.
    35.
    36.
    37.
    38.
    39.
    40.
    41.
    42.
    43.
    44.
    45.
    46.
    47.
    48.
    49.
    50.
    51.
    52.
    53.
    54.
    55.
    56.
    57.
    58.
    59.
    60.
    61.
    62.
    63.
    64.
    65.
    66.
    67.
    68.
    69.
    70.
    71.
    72.
    73.
    74.
    75.
    76.
    77.
    78.
    79.
    80.
    81.
    82.
    83.
    84.
    85.
    86.
    87.
    88.
    89.
    90.
    91.
    92.
    93.
    94.
    95.
    96.
    97.
    98.
    99.
    100.
    101.
    102.
    103.
    104.
    105.
    106.
    107.
    108.
    109.
    110.
    111.
    112.
    113.
    114.
    115.
    116.
    117.
    118.
    119.
    120.
    121.
    122.
    123.
    124.
    125.
    126.
    127.
    128.
    129.
    130.
    131.
    132.
    133.
    134.
    135.
    136.
    137.
    138.
    139.
    140.
    141.
    142.
    143.
    144.
    145.
    146.
    147.
    148.
    149.
    150.
    151.
    152.
    153.
    154.
    155.
    156.
    157.
    158.
    159.
    160.
    161.
    162.
    163.
    164.
    165.
    166.
    167.
    168.
    169.
    170.
    171.
    172.
    173.
    174.
    175.
    176.
    177.
    178.
    179.
    180.
    181.
    182.
    183.
    184.
    185.
    186.
    187.
    188.
    189.
    190.
    191.
    192.
    193.
    194.
    195.
    196.
    197.
    198.
    199.
    200.
    201.
    202.
    203.
    204.
    205.
    206.
    207.
    208.
    209.
    210.
    211.
    212.
    213.
    214.
    215.
    216.
    217.
    218.
    219.
    220.
    221.
    222.
    223.
    224.
    225.
    226.
    227.
    228.
    229.
    230.
    231.
    232.
    233.
    234.
    235.
    236.
    237.
    238.
    239.
    240.
    241.
    242.
    243.
    244.
    245.
    246.
    247.
    248.
    249.
    250.
    251.
    252.
    253.
    254.
    255.
    256.
    257.
    258.
    259.
    260.
    261.
    262.
    263.
    264.
    265.
    266.
    267.
    268.
    269.
    270.
    271.
    272.
    273.
    274.
    275.
    276.
    277.
    278.
    279.
    280.
    281.
    282.
    283.
    284.
    285.
    286.
    287.
    288.
    289.
    290.
    291.
    292.
    293.
    294.
    295.
    296.
    297.
    298.
    299.
    300.
    301.
    302.
    303.
    304.
    305.
    306.
    307.
    308.
    309.
    310.
    311.
    312.
    313.
    314.
    315.
    316.
    317.
    318.
    319.
    320.
    321.
    322.
    323.
    324.
    325.
    326.
    327.
    328.
    329.
    330.
    331.
    332.
    333.
    334.
    335.
    336.
    337.
    338.
    339.
    340.
    341.
    342.
    343.
    344.
    345.
    346.
    347.
    348.
    349.
    350.
    351.
    352.
    353.
    354.
    355.
    356.
    357.
    358.
    359.
    360.
    361.
    362.
    363.
    364.
    365.
    366.
    367.
    368.
    369.
    370.
    371.
    372.
    373.
    374.
    375.
    376.
    377.
    378.
    379.
    380.
    381.
    382.
    383.
    384.
    385.
    386.
    387.
    388.
    389.
    390.
    391.
    392.
    393.
    394.
    395.
    396.
    397.
    398.
    399.
    400.
    401.
    402.
    403.
    404.
    405.
    406.
    407.
    408.
    409.
    410.
    411.
    412.
    413.
    414.
    415.
    416.
    417.
    418.
    419.
    420.
    421.
    422.
    423.
    424.
    425.
    426.
    427.
    428.
    429.
    430.
    431.
    432.
    433.
    434.
    435.
    436.
    437.
    438.
    439.
    440.
    441.
    442.
    443.
    444.
    445.
    446.
    447.
    448.
    449.
    450.
    451.
    452.
    453.
    454.
    455.
    456.
    457.
    458.
    459.
    460.
    461.
    462.
    463.
    464.
    465.
    466.
    467.
    468.
    469.
    470.
    471.
    472.
    473.
    474.
    475.
    476.
    477.
    478.
    479.
    480.
    481.
    482.
    483.
    484.
    485.
    486.
    487.
    488.
    489.
    490.
    491.
    492.
    493.
    494.
    495.
    496.
    497.
    498.
    499.
    500.
    501.
    502.
    503.
    504.
    505.
    506.
    507.
    508.
    509.
    510.
    511.
    512.
    513.
    514.
    515.
    516.
    517.
    518.
    519.
    520.
    521.
    522.
    523.
    524.
    525.
    526.
    527.
    528.
    529.
    530.
    531.
    532.
    533.
    534.
    535.
    536.
    537.
    538.
    539.
    540.
    541.
    542.
    543.
    544.
    545.
    546.
    547.
    548.
    549.
    550.
    551.
    552.
    553.
    554.
    555.
    556.
    557.
    558.
    559.
    560.
    561.
    562.
    563.
    564.
    565.
    566.
    567.
    568.
    569.
    570.
    571.
    572.
    573.
    574.
    575.
    576.
    577.
    578.
    579.
    580.
    581.
    582.
    583.
    584.
    585.
    586.
    587.
    588.
    589.
    590.
    591.
    592.
    593.
    594.
    595.
    596.
    597.
    598.
    599.
    600.
    601.
    602.
    603.
    604.
    605.
    606.
    607.
    
```


```

        EnableOut    <= '1' ;
        data(15) <= not data(15) ;
    end if;
end if;
end if;
end if;
end process;
```

```
StateTransProc :
process(ok, reset)
begin
    if reset = '1' then
        state <= idle;
    elsif rising_edge(ok) then
        if clearStrobe = '1' then
            state <= idle;
        elsif CMCover = 0 then
            state <= rotate;
        end if;
    end if;
```

```

end if;
end process;

HwCtrlIdleProc :
process(state, KbdIn, MailBox)
begin
    case state is
        when Idle is =>
            if KbdIn = '1' then
                state <= Chg;
            end if;
        when

```

```

        rotate <= Mile.at;
    and if;
    when Conv.at <=
        if MailComplete = '1' then
            rotate <= Mail.at;
        else
            rotate <= Conv.at;
    and if;
    when Mail.at <=
        rotate <= Mail.at;
    when Mail.at <=
        rotate <= Mail.at;

```

```

return <> send;
when Mail <= 0
  if MailComplete = '1' then
    return <> Done;
  else
    return <> Mail;
  end if;
when Done <= 0
  return <> Done;
end when;
end state;
end all;

```

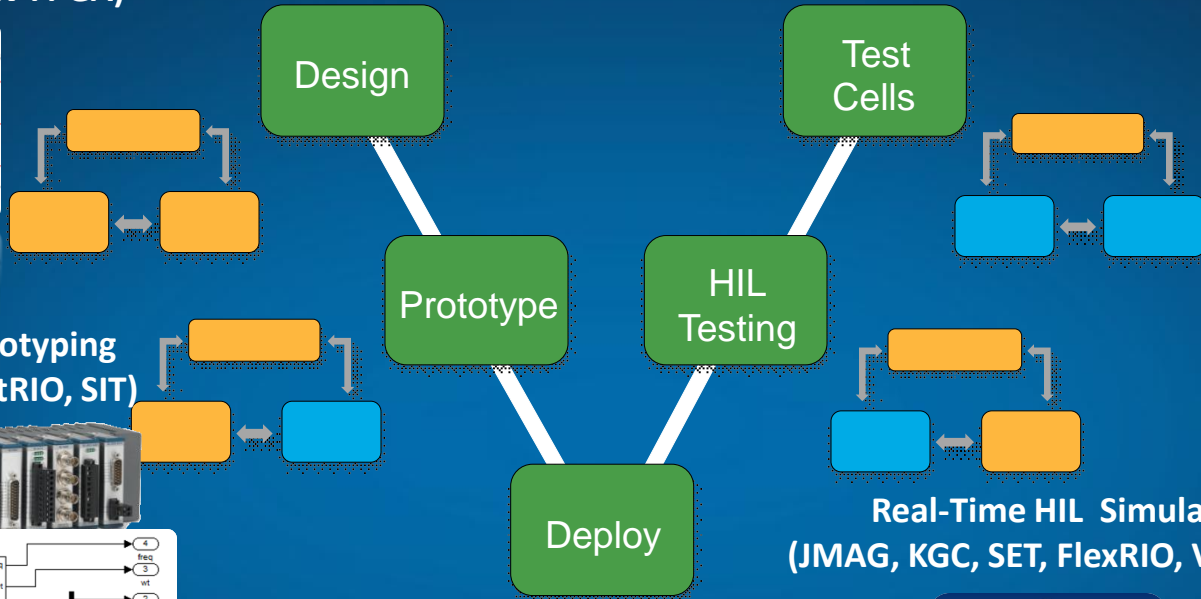
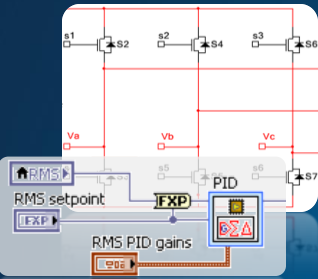
100

Analog I/O

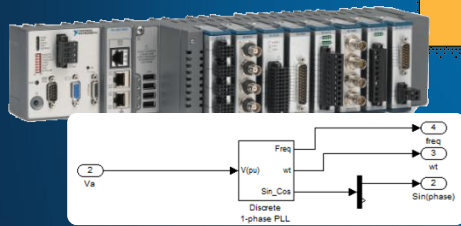
Designing at a Higher Level

Graphical Co-Simulation
(Multisim, LabVIEW FPGA)

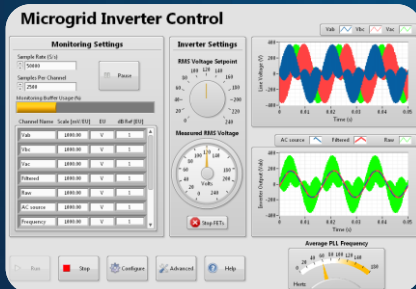
Power Electronics Testing
(Bloomy Energy, PXI)



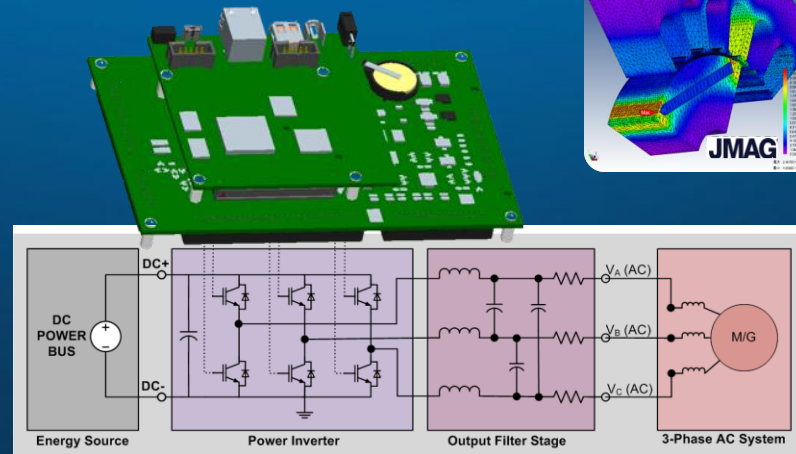
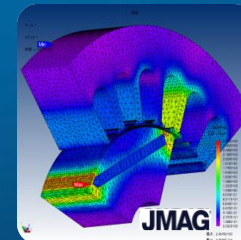
Rapid Control Prototyping
(Multicore CompactRIO, SIT)



Commercial Deployment
(General Purpose Inverter Controller)



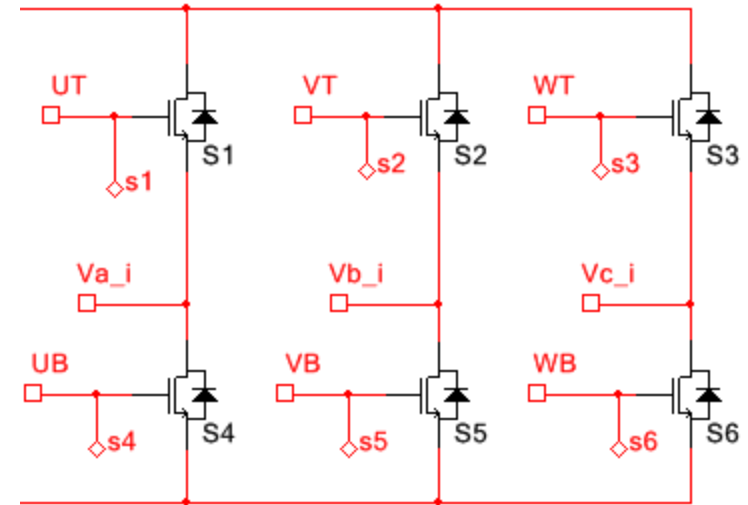
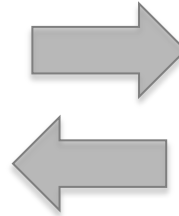
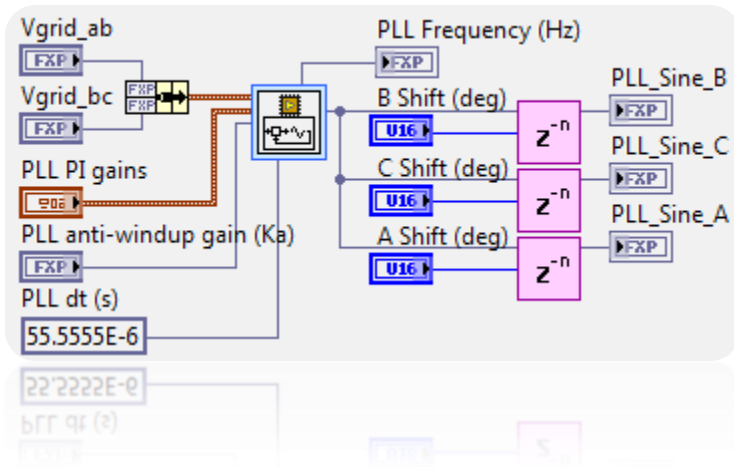
Real-Time HIL Simulation
(JIMAG, KGC, SET, FlexRIO, Veristand)



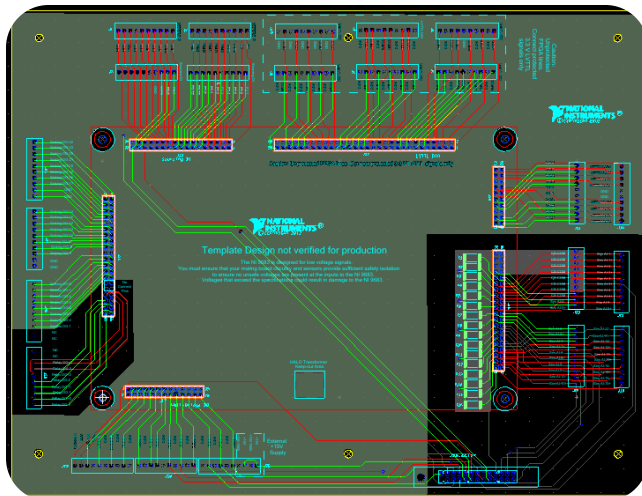
Development Methodology

1. Co-Simulation, 2. Interface Board Design, 3. Commercial Deployment

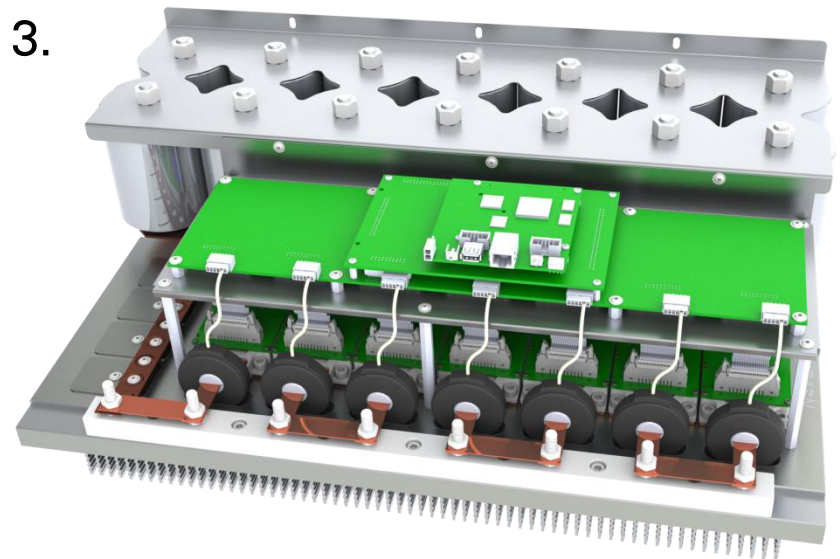
1.



2.

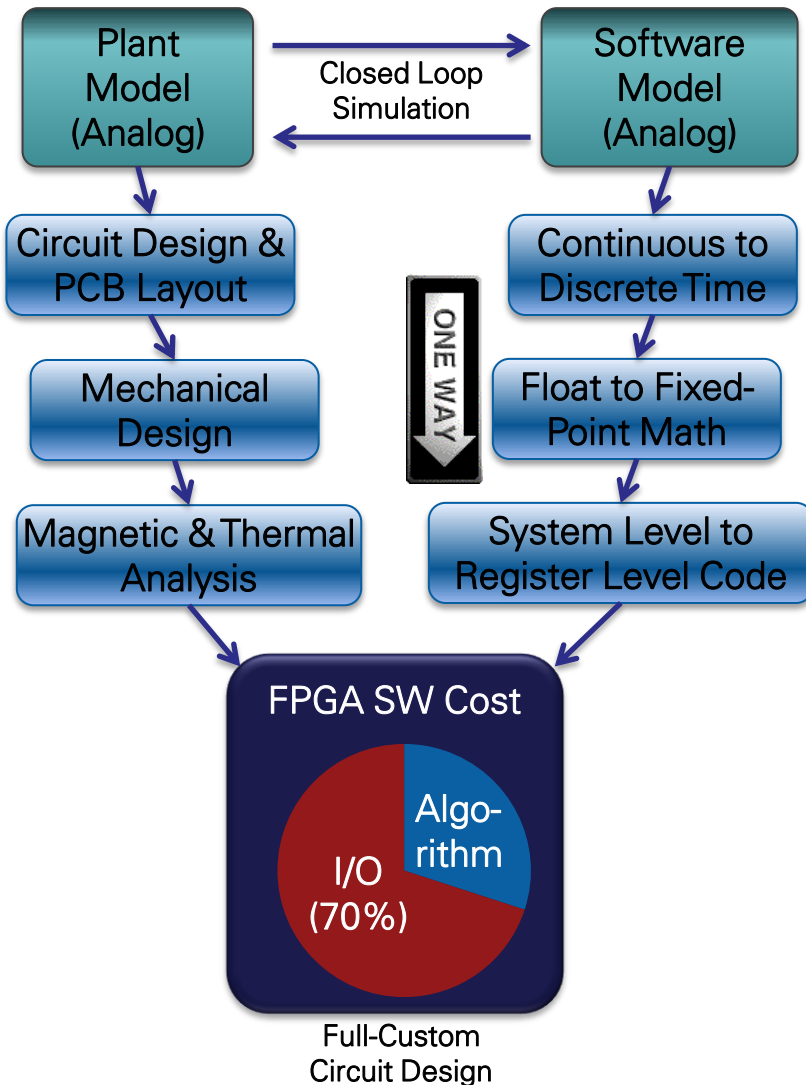


3.

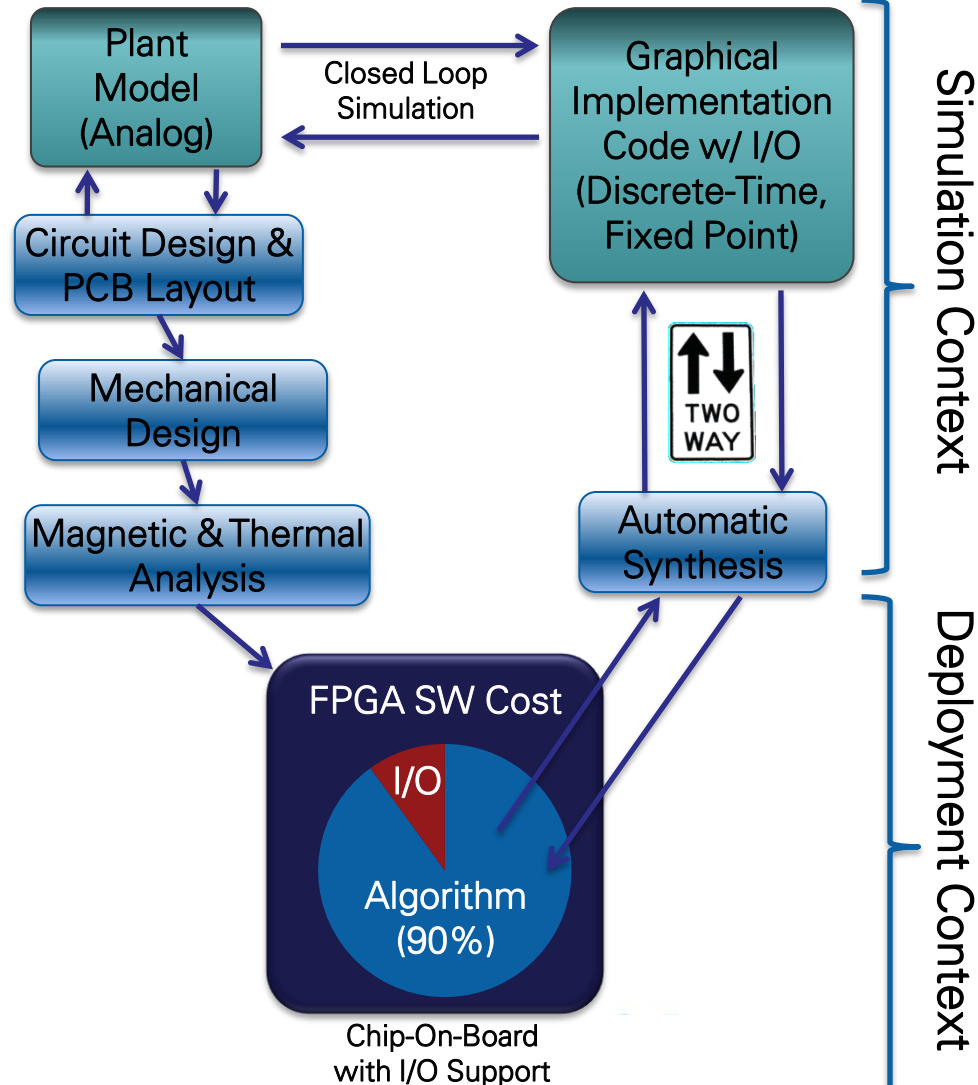


The Evolution of System Level Design

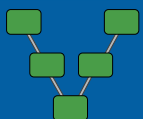
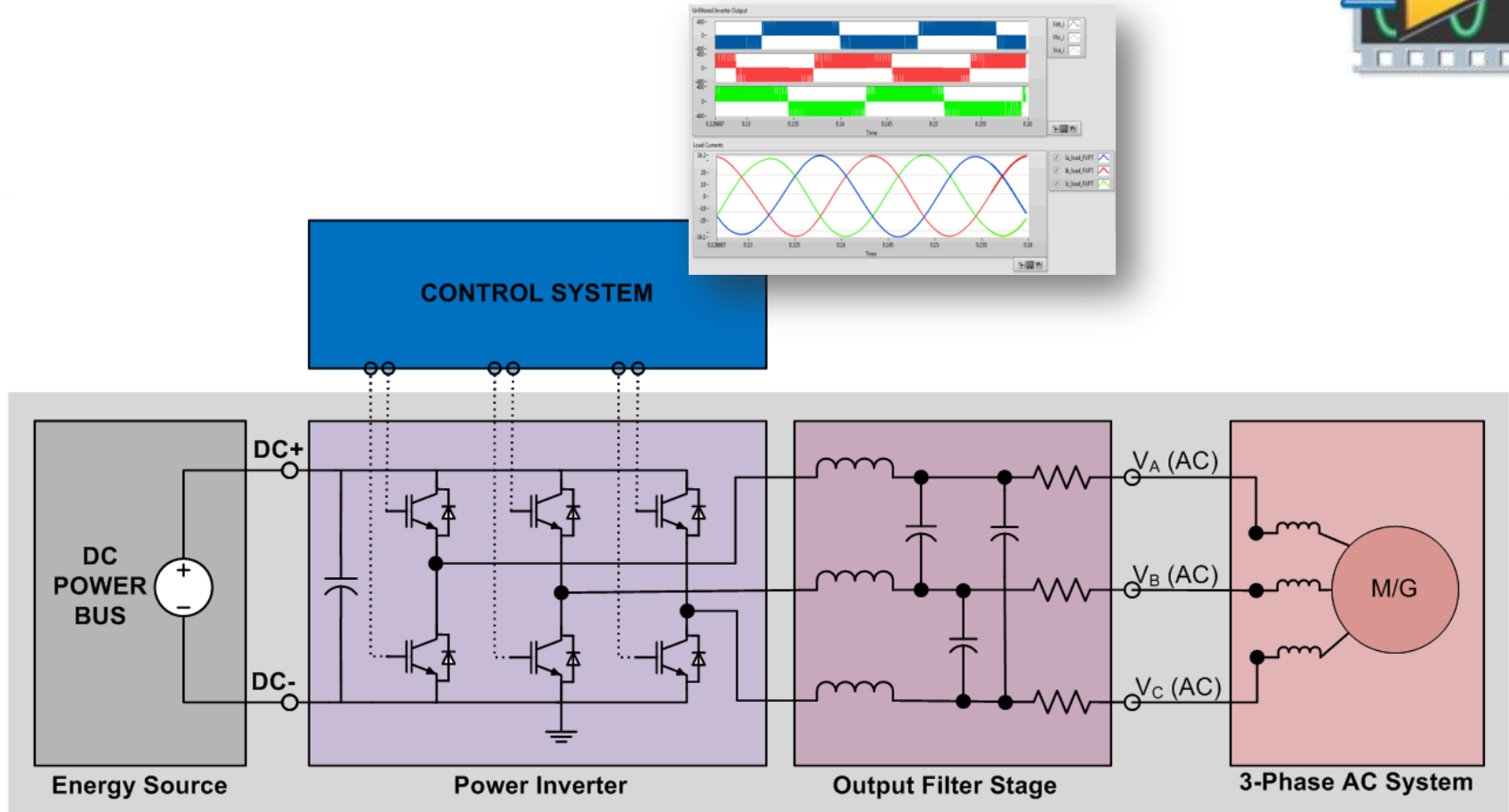
Traditional Methodology



Proposed Methodology



Demos



New data on LabVIEW graphical system design approach

■ Wilson Research Study

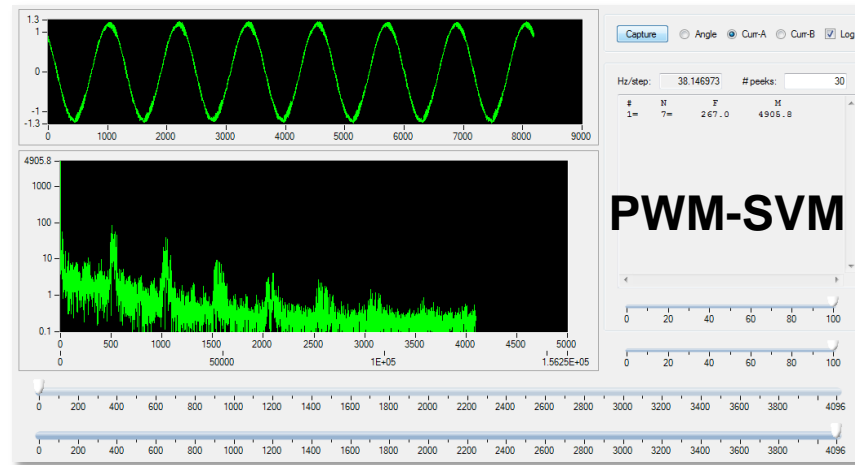
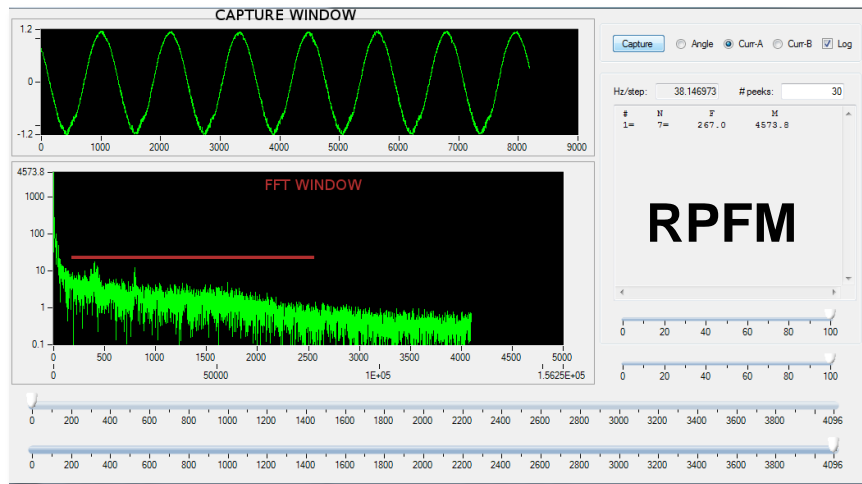
	NI Embedded Customers (2012) ¹	EETimes Overall Embedded Market (2012) ²	Ratio
Average Development Team Size (HW, SW, Firmware Engineers)	4.8	11.5	2.4
Average Months to Complete Project	6.2	12.5	2.0
Average Person-Months to Complete Project	30	144	4.8 (average of 114 person-month savings per design)
Average Development Cost (assuming \$100k/person/year with overhead)	\$248,000	\$1,198,000	4.8 (average \$950,000 cost savings per design)
Percent of Projects Completed On or Ahead of Schedule	58% of NI customers	42% of embedded market	0.7
Percent of Projects Completed Behind Schedule/Late	38% of NI customers	55% of embedded market	1.4

NOTE1: The overall embedded market study was a global Email/web study including over 1,700 responses from embedded engineers from Americas, Europe and Asia

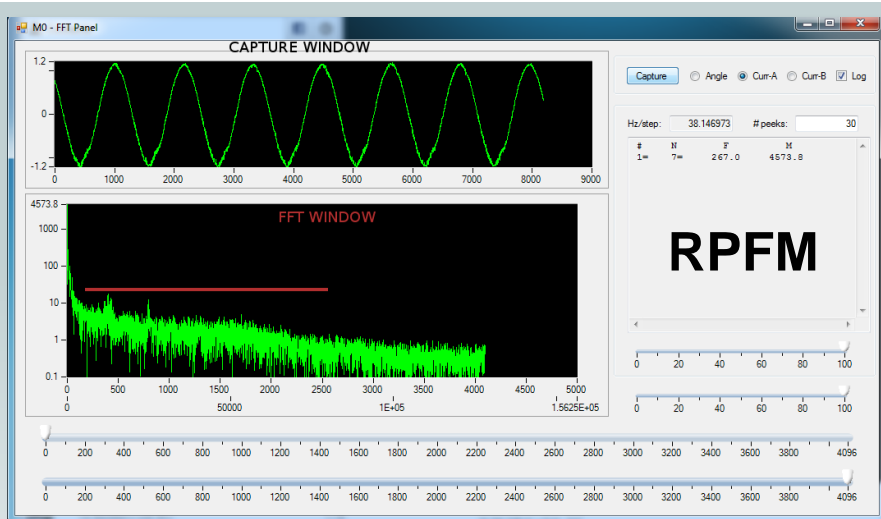
NOTE2: The study of NI embedded customers was a global Email/web study including over 1,100 responses NI embedded customers from Americas, Europe and Asia

Achieving Performance with FPGA-based Control

Methods	Disadvantage	Advantage	DSP	FPGA
6 step PWM	<ul style="list-style-type: none"> - Rough motor operation - Degrades motor life 	<ul style="list-style-type: none"> - Simple to implement - Highest Power Output 	✓	✓
Sinusoidal PWM	<ul style="list-style-type: none"> - Not power efficient 	<ul style="list-style-type: none"> - Smooth Motor operation - Relatively simple to design 	✓	✓
Space Vector PWM	<ul style="list-style-type: none"> - Complex Algorithm - Requires Processing 	<ul style="list-style-type: none"> - Smooth Motor operation - 15% more efficient vs sinusoidal 	Some	✓
Frequency Modulation	<ul style="list-style-type: none"> - Requires Processing 	<ul style="list-style-type: none"> - Less harmonics - Switching loss reduced 		✓



RPFM and Space Vector Modulation

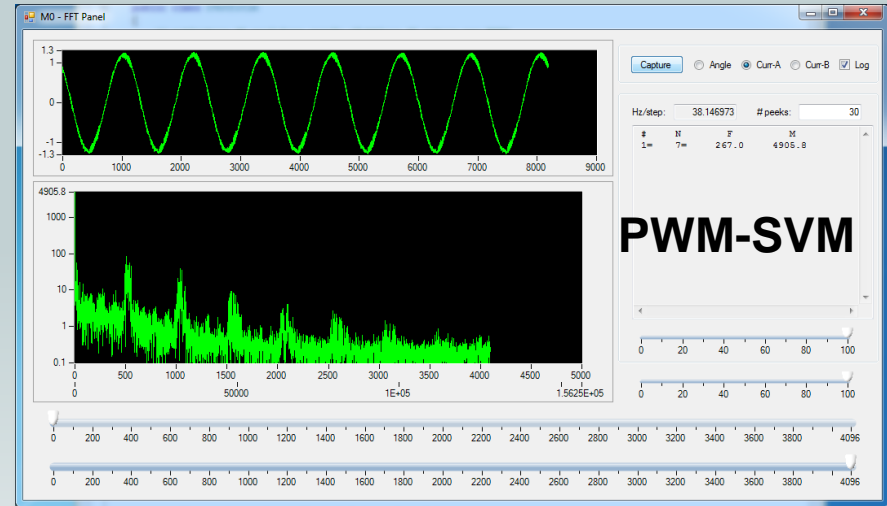


RPFM

Motor Dunkermotoren BG65x50

Regenerative Pulse Frequency Modulation

- Less harmonics
- Less switching



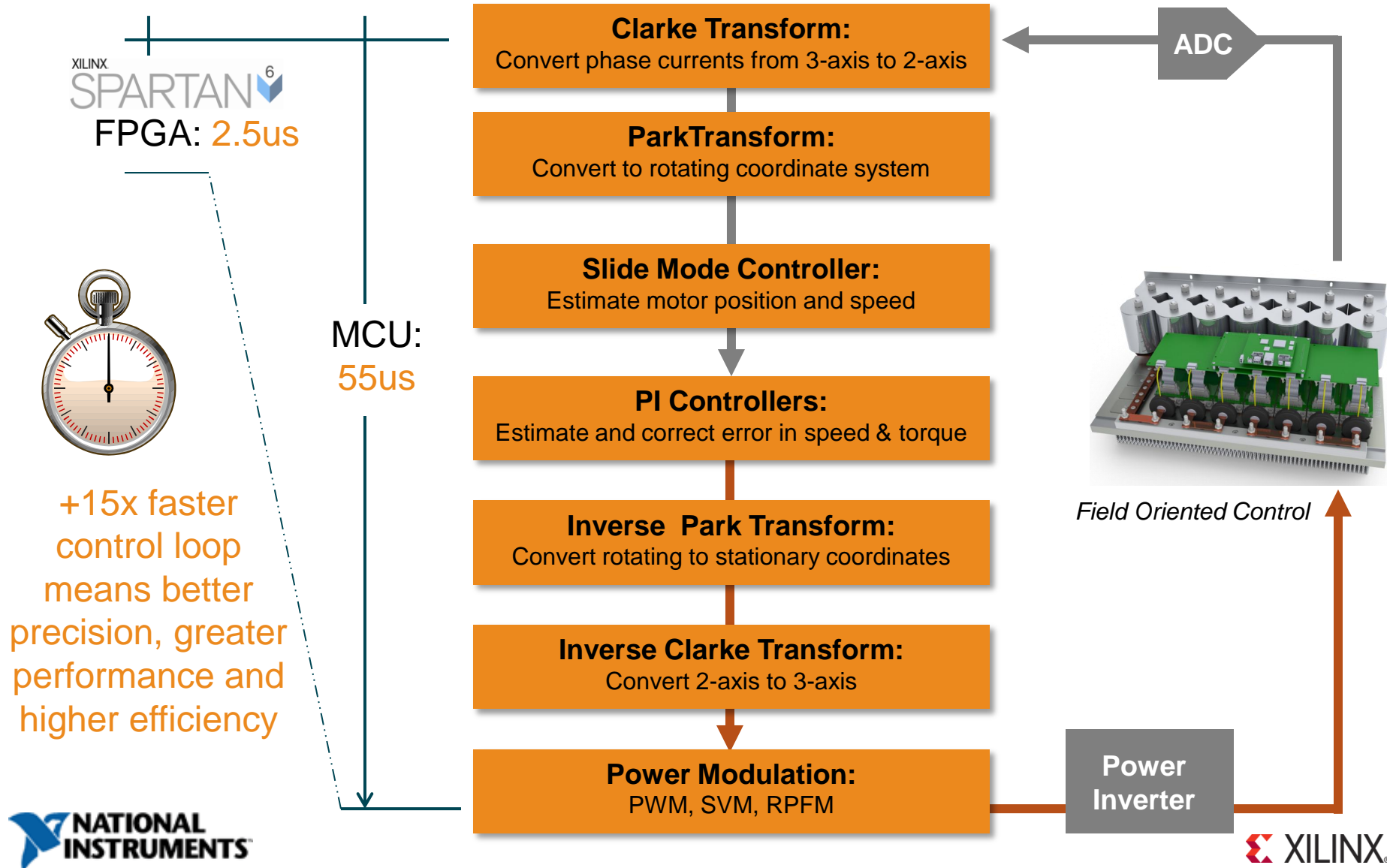
PWM-SVM

Pulse Width Space Vector Modulation

- Spectral peaks on harmonics

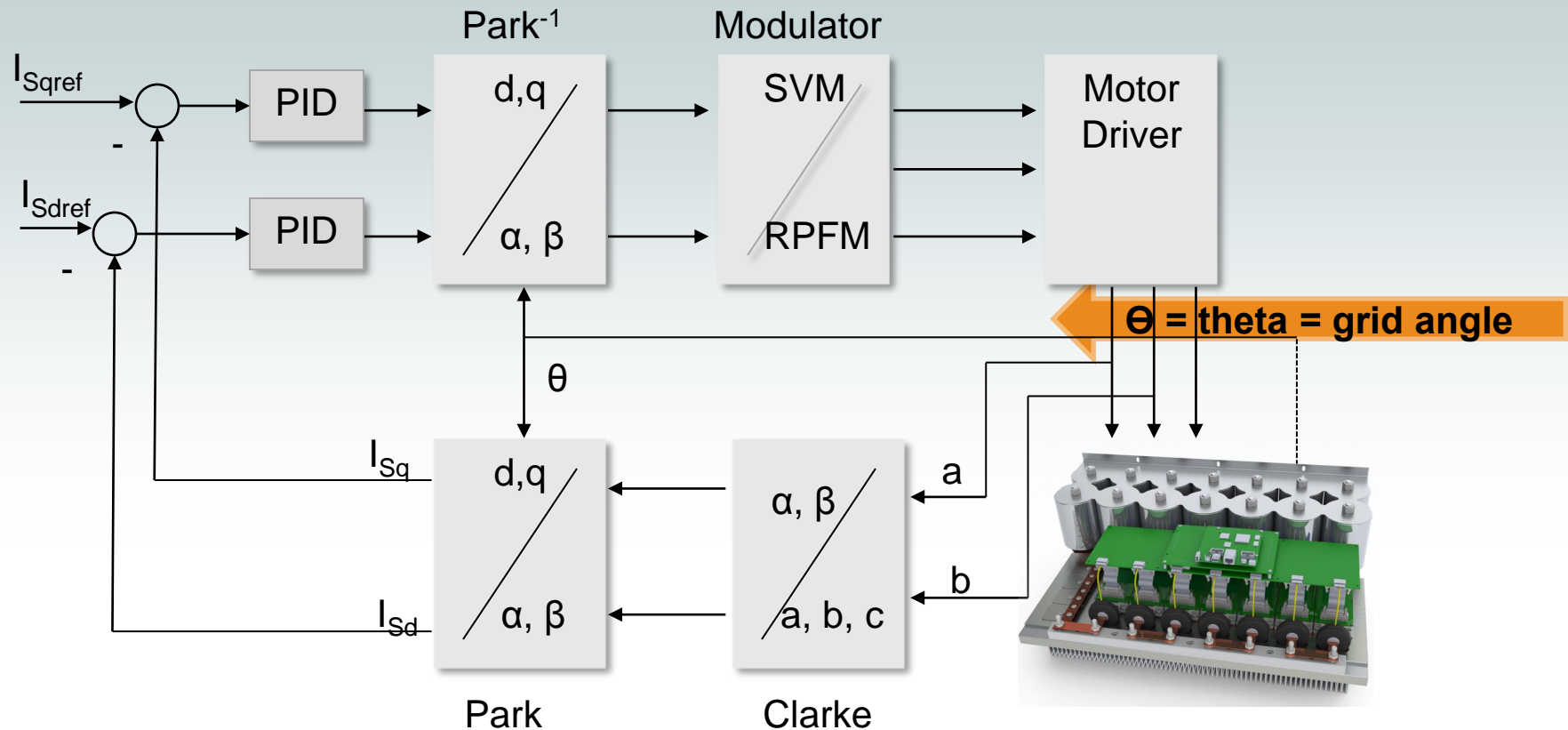
Excellent Noise Reduction = Less EMI

FPGA Performance Advantage Example



Field Oriented Control

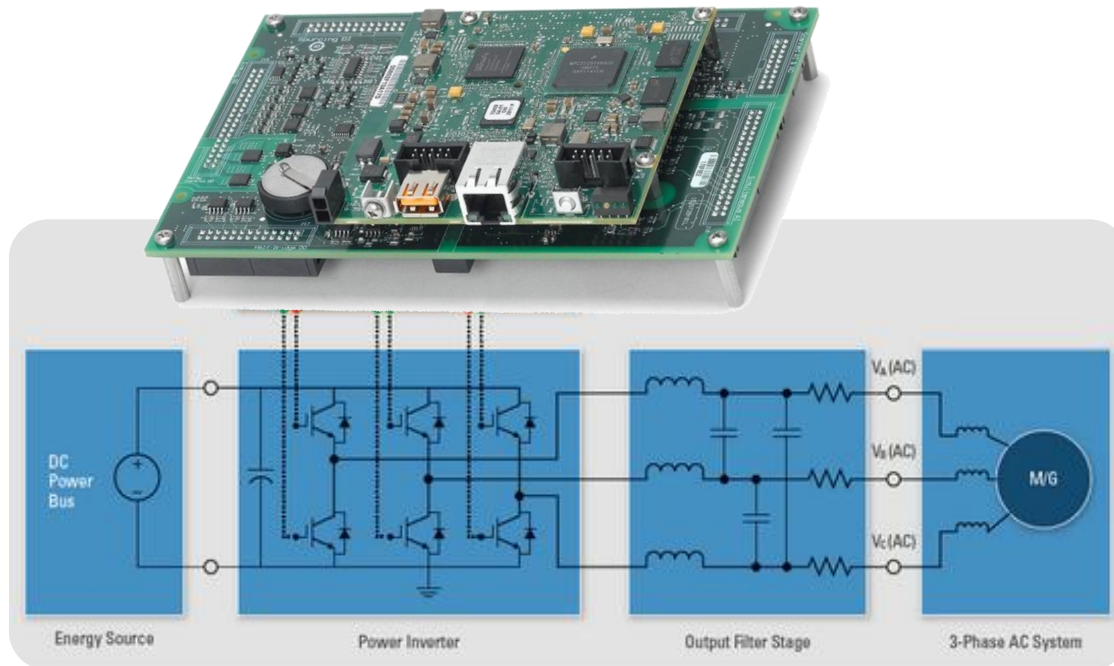
In a Nutshell



Measure Theta = FOC

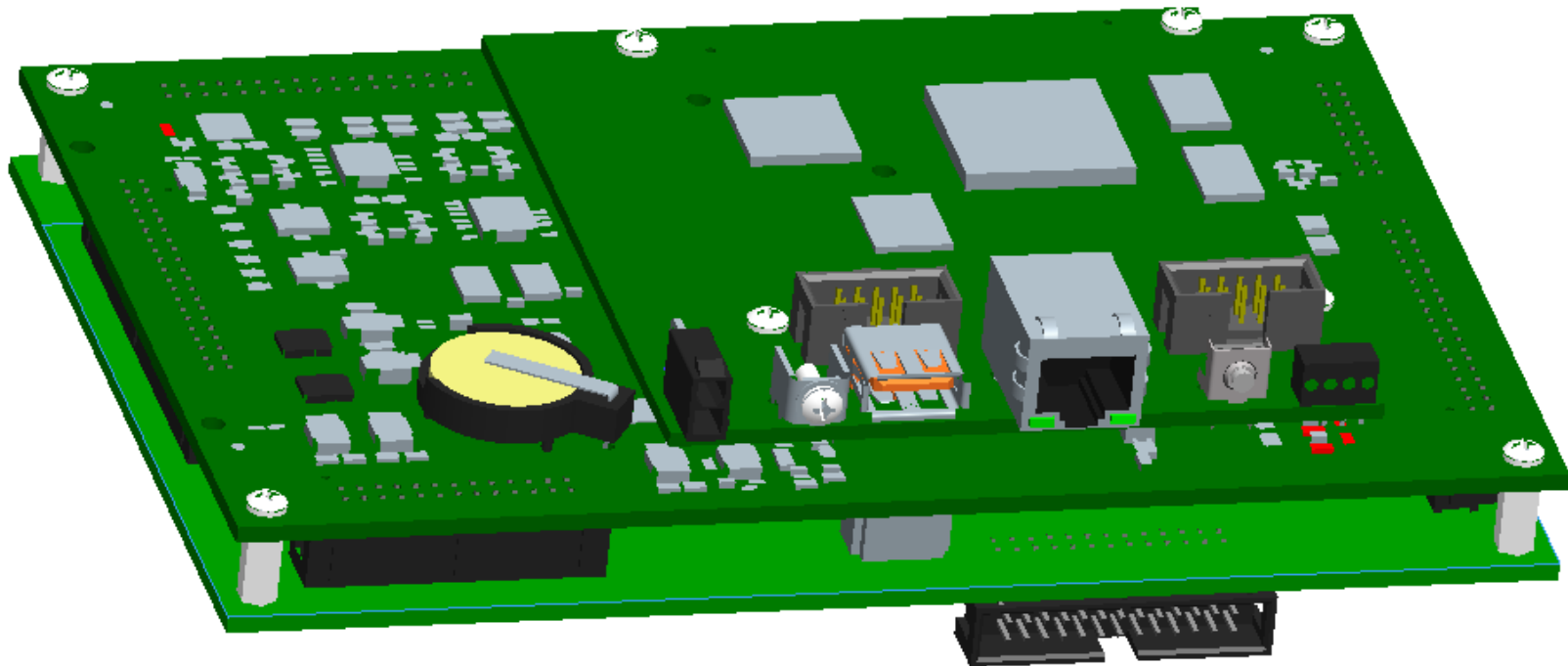
NI Single-Board RIO General Purpose Inverter Controller (NI GPIC)

- Industry-proven NI LabVIEW RIO architecture and cutting-edge co-simulation tools
- Deployment-ready for high volume commercial applications
- Multicore FPGA delivers 40x higher performance per dollar than traditional DSPs



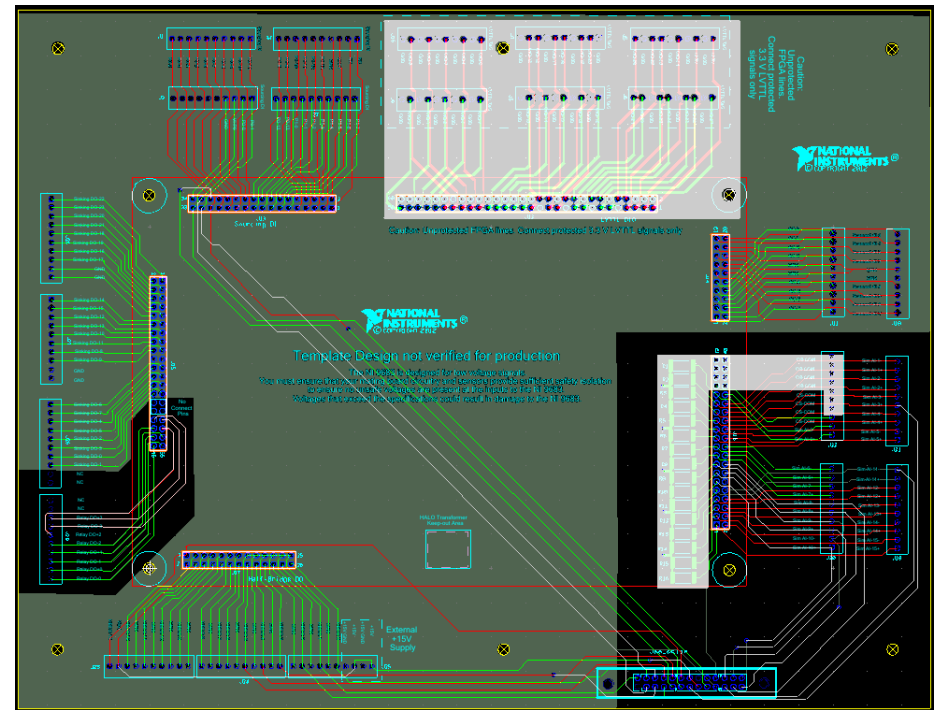
Typical Stack

1. NI Single-Board RIO sbRIO-9606
2. NI GPIC RIO Mezzanine Card (bottom orientation connectors)
3. Custom interface or gate drive PCB (not provided by NI)

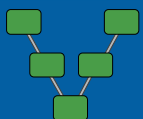
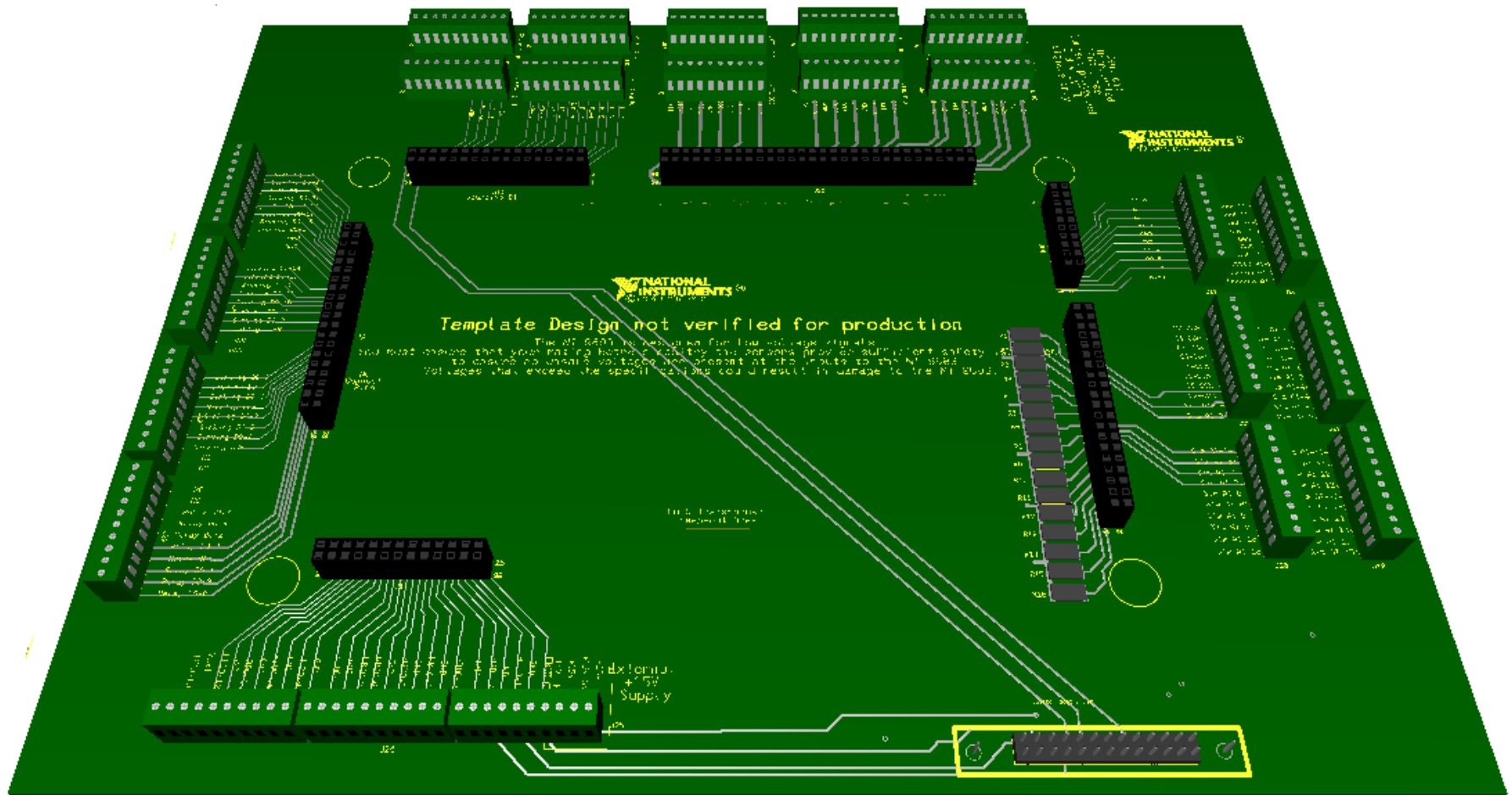


To Gate Driver
or IPM

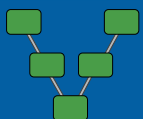
Diagram illustrating a tree structure with 5 nodes and 4 edges. The root node is at the bottom, branching into two nodes, which each branch into one more node, resulting in a total of 5 nodes and 4 edges. The text "ni.com" is displayed to the right of the diagram.



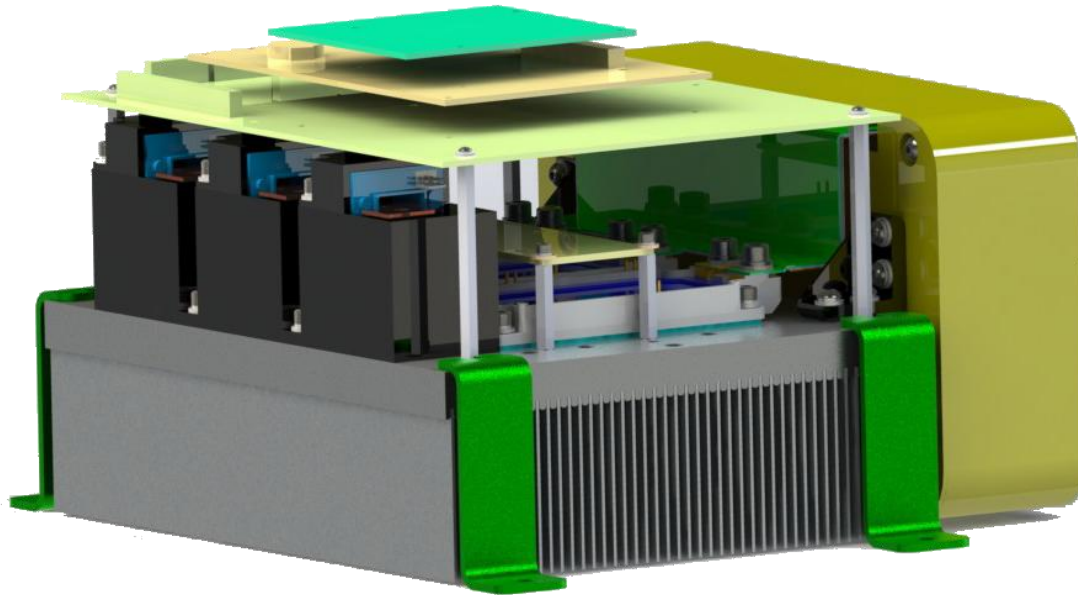
GPIC Mating PCB Template (NI Ultiboard)



GPIC Mating PCB Template (NI Ultiboard)



SmartPower Stack™ Inverter



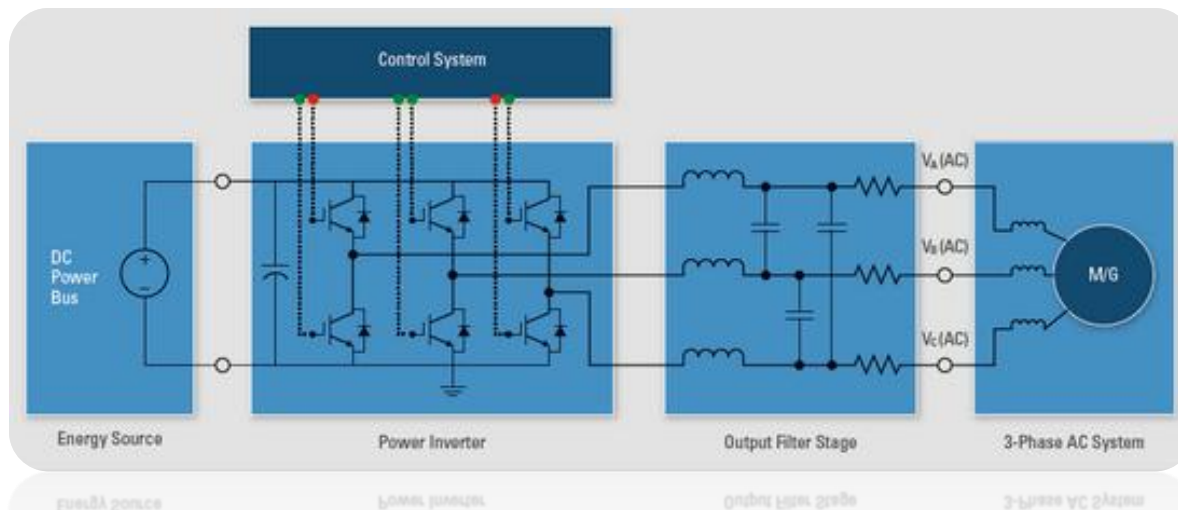
 AgileSwitch™

 Fuji Electric

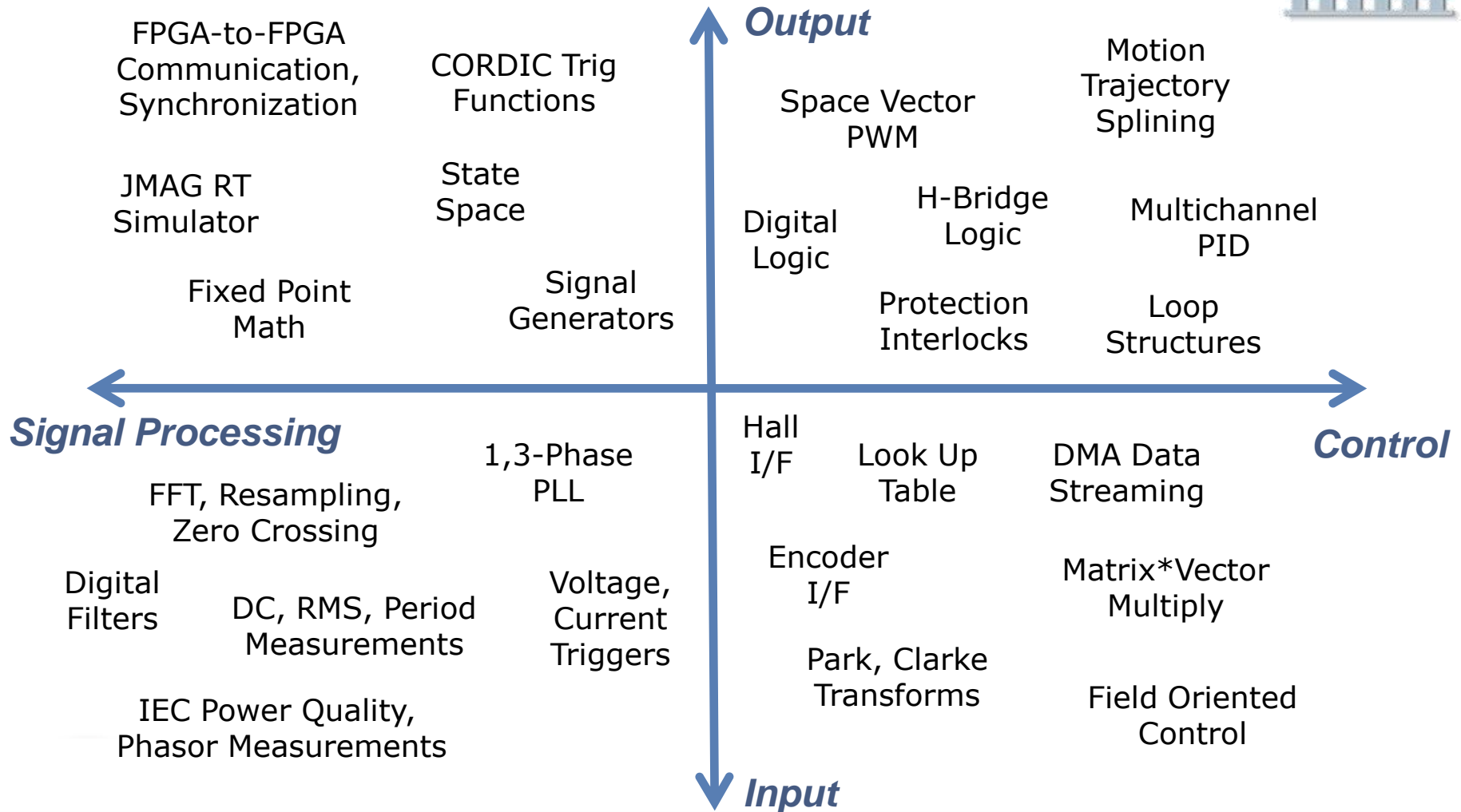
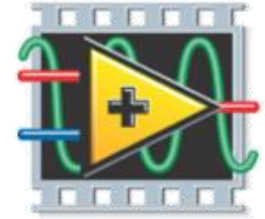
 METHODE ELECTRONICS, INC.

 NATIONAL
INSTRUMENTS™

 SBE



LabVIEW FPGA Power Electronics IP



Next Steps

Visit
ni.com/powerdev
for more info

Download the Power
Electronics Design Guide

Visit
ni.com/gpic

Order the NI GPIC
Evaluation Kit

Q & A