System On a Programmable Chip (SOPC) based Power Electronic Controller

By

AJEESH A
CDAC-Thiruvananthapuram
ajeesh@cdac.in
Some Calculations

- A Processor with one million switching devices
- Use Vacuum tubes (Triodes), not MOS Transistors
Technical issues

Area

Assume the triode occupies an area of 5cm X 5cm including some space between adjacent tubes

- Total area of 50 X 50 = 2500 m²

- With multilayer area = 22m X 22m
Technical issues

Power consumption

Assume

- Taking 5W per triode
- Half of the triodes are conducting at any time

- The total power consumed = 0.5 \times 10^6 \times 5
  = 2.5 MW
Technical issues

Heat dissipation

The electric Power consumed

Heat
Technical issues

Heat dissipation
Technical issues

Reliability

Let ‘T’ – Life of a transistor in hours

> Replace

on an average one triode every $\frac{T}{10^6}$ hr

Grant $T = 10^6$

Then we need to replace a triode every “one hour”

24X7 maintenance
The system
Technical issues

Speed

Below 1 MHz due to parasitic wiring capacitance and Inductance

Longer development time
Summary

- Area/Size
- Power Consumption
- Reliability
- Speed
- Time to market
- Cost
FPGAs

1980s

- PLDs
  - Configurable
  - Fast design and modification time
  - Could not support large or complex functions

The Gap

- ASICs
  - Support large or complex functions
  - Expensive and time-consuming
  - Frozen in Silicon

C-DAC ©All Rights Reserved
FPGAs

1984s

- Highly configurable
- Fast design and modification time
- Could not support large or complex functions

- Support large or complex functions
- Expensive and Time consuming
- Frozen in Silicon

Xilinx Introduced FPGAs in Market
CMOS – SRAM Based
FPGAs

Programmable logic Blocks
- A 3 input Lookup table (LUT)
- Flip-Flop or Latch
- A Multiplexer

Modern FPGAs Contains complex versions of “Programmable logic Blocks”

Using SRAM every logic block in the device can be programmed
PLB

- a
- b
- c
- d
- clk
- LUT
- MUX
- Flip-Flop
- y
- q
Configuring LUT

Required Function

\[ y = (a \text{ AND } b) \text{ OR } \neg c \]

Truth Table

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>c</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Programmed LUT
FPGA Structure

Complex architecture today
FPGA Design Flow

1. Functional Spec:
2. HDL (VHDL, Verilog, Bluespec)
3. Synthesis
4. Place and Route
5. Download and verify ckt
6. Timing
PE Control System Design

- Analog – Using OP-AMPS, Resistors, Capacitors etc.

- Digital – Using Processors and software programming
  
  Logics and numerical methods – DSPs
“What if have a Processor so that user defines Instructions, Internal Modules, Speed, Memory, pins etc. ???”

*Obsolescence free design!!!*
“Building embedded system into a single programmable integrated circuit like FPGAs”

Generally involves utilization of a large FPGA

- Processor cores (Soft / Hard)
- Memory (RAM and ROM)
- Intellectual property (IP), Custom hardware blocks/IPS

“have all (or majority) of the components on a single programmable chip”
Need for Development

**SOPC PE Controller**

**Long term support for Hardware and software**

- Industrial PE systems, Power system controls etc. (5 -10Y)
- Railways, Defense Application (10-30Y)

- *Processor Obsolescence* in 4 - 6 years

- *Next generation Processor*: Software porting overhead according to Processor Hardware changes

  *soft processor IP core in general purpose FPGA*

**Single chip Solution:**

- Difficult to find a processor with all interfaces/reconfigurable for the application

  *Custom made IPs with the CPU core*

- *Single core to multi core*: with increased cost, may require hardware redesign

  *same FPGA Hardware for single to multi core application*
Need for Development
DSP Revenue

Nobody is going to use a mobile more than 20 Years
Need for Development

**Faster Controller: Sampling of 5-10 microsec or less**

- **Use Parallel processing** – multi-core processor integration
- **Use Hardware accelerators** - Control blocks in dedicated hardware to perform functions faster than a CPU (PI, Filters, PLL, PWM)
- **C2H Compilers:** Inspect the application ‘C’ code and try to convert time consuming software algorithms into equivalent hardware implementation
  
  **Hardware performs faster than software**

- **Hardware Parallel Processing than Software pipelining**
  
  8201 NIOS II cycles equivalent to 24603 TigerSHARC cycles
  
  (200 MHz vs 600 MHz)

**Better Time to Market criteria**

- Same FPGA Board
- Proven CPU and IP core Set

*handles a wide range of applications*
Need for Development

**Higher reliability**

- Higher level of integration and resources utilization of a single FPGA

**Board Design**

On chip interconnections: *Industry standard (AVALON, AMBA AXI)*

Internal PLLs: *Remove the need to distribute high speed clocks round a PCB*

**Reduced EMI**

“PCB can be smaller, consume less power, be easier to get through EMC tests”

**Board Testing**

“Dedicated test designs can be loaded into the FPGA to automate Board testing”
Conventional Controller Design

CONVENTIONAL CONTROLLERS

AC Drive Control Board

FPGA
(DSP offload and Glue logic)

DSP or MCU

PWM

Power Stage

A/D Converters

Off-chip Interconnection

M

Encoder

Conventional Controller Design
Conventional Controller Design

AC Drive Control Board

- FPGA
  (DSP offload and Glue logic)
- DSP or MCU
- PWM
- A/D Converters
- Power Stage
- Encoder

Off-chip Interconnection

Communication Board

- FPGA/MCU/ASSP
- BRIDGE
- PHY

Industrial Ethernet

CONVENTIONAL CONTROLLERS
Resource Utilization

AC Drive Control Board

- FPGA: 10%
- DSP or MCU: 60%
- Power Stage
- PWM
- A/D Converters
- Encoder

Communication Board

- FPGA/MCU/ASSP: 20%
- BRIDGE
- PHY
- Industrial Ethernet
SOPC PE Controller

AC Drive Control Board

- PWM Generator
- CPU (soft/Hard)
- MAC
- FPGA

Power Stage

- A/D Converters
- Digital Encoder

Encoder

Industrial Ethernet

FPGA as a Drive Controller Chip

HDL Motor Control Logic, Space Vector Mod, IGBT Control etc.
SOPC PE Controller Board V_1.1

Specifications

- **FPGA**: Cyclone III EP3C25E144C8N (24,624 LEs)
- **On chip Memory**: 64 kBytes (Inside FPGA)
- **Flash memory**: 2 MB
- **Digital I/Os**: 56 No.s 3.3-V LVTTL - configurable for ADC/DAC, PWM, I2C interfaces
- **Host interface**: JTAG
- **Supply voltage**: 3.3 V

- This Generic Board can handle Most of the PE applications
- The Hardware inside the FPGA is Configurable depending on PE Application
- The I/Os are Configurable
Peripheral Interface Card

Specifications

- Analog Input: +10V (ADC 13 bit 8 Channel, SPI interface)
- Analog Output: +3.3V (DAC 12 bit 8 Channel, SPI interface)
- Digital input: +5V
- Digital Output: +5V
- Supply voltage: 24 V
- Stack Connector for SOPC_CONTROLLER Card

Targeted Application

- DC-DC converter control
- AC drive applications
- Front end converter etc.

“CUSTOM DESIGN FOR SPECIFIC APPLICATION IS POSSIBLE”
SOPC Configuration

High Performance Drive

- Rectifier
- Inverter
- Rectifier Control
- Inverter Control
- MVB Controller
- Motor Model Estimates The speed from Voltage and currents
- DSP/Micro controller implementation, the motor model and other control blocks will be implemented as a software program.
- In SOPC architecture, control blocks are realized in Hardware using HDL called IPs
Control blocks now part of the CPU – **Hardware Modules**
Other similar PE applications with out redesign - “Reuse”
Can be accessed Using **Custom Instructions/Functions**
### IPs Required for The application

<table>
<thead>
<tr>
<th>SOPC Configuration for <strong>DC-DC Control</strong></th>
<th>SOPC Configuration for <strong>Inverter Control</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>IP cores for Basic control</td>
<td></td>
</tr>
<tr>
<td>Soft Processor core</td>
<td>Soft Processor core</td>
</tr>
<tr>
<td>PI Controller</td>
<td>PI Controller</td>
</tr>
<tr>
<td>PWM(2 Channel)</td>
<td>Motor Dynamic Model</td>
</tr>
<tr>
<td>Subtraction</td>
<td>PWM(6 Channel)</td>
</tr>
<tr>
<td>Limiter</td>
<td>Phase Transformers</td>
</tr>
</tbody>
</table>

**Multi processor core Implementation in one FPGA Possible**
Processors in FPGAs

- Soft or Hard CPU cores

- Soft Processor
  - Processor implemented in VHDL, Verilog, etc., and downloaded onto FPGA hardware
  - Example: NIOS II, MicroBlaze, ARM Cortex-M1 etc
  - Highly Reconfigurable, a schematic (or code like software).

“More than 20 soft core processors are available”
LEON 3, S1 core (64Bit), ARM Cortex-M1, DPUVA-16 etc
Hard processors on FPGA

- SoC FPGA
- FPGA + Hard Processor System (HPS)
- Can’t alter Placement, Routing, and Area
- Cyclone V SoC from Altera
- Optimized for Higher speed of operation 925 MHz
- Costly
ALTERA NIOS II CPU

- Full 32-bit instruction set, data path, and address space
- 32 external interrupt sources
- Single-instruction 32 x32 multiply and divide producing a 32-bit result.
- Access to a variety of on-chip peripherals, and interfaces to off-chip memories and peripherals
- Speed: 20 MHz to 430 MHz
The top level entity of the IPs should meet AVALON Specification for NIOS.
For ARM, AMBA Bus specification.
The Control blocks should be designed in HDL.

Functionality and timing are properly monitored using test bench or IP Verification methods.

*Time consuming process but one time process*
Technology Developed

- PE IP Library
  - PE Specific IPs
- Design procedure for the IP Developer
- Drivers for PE IP library
- Design procedure for SOPC user
- SOPC Controller Card
- SOPC Interface Card
- PE Application Evaluation Report
Application Development in ALTERA FPGAs

“Faster Concept to System”

- **Tools**
  - Altera QUARTUS II V9.1 or higher with SOPC Builder/Qsys
  - NIOS II Eclipse IDE for software development
  - Model sim for Simulation

- **Hardware**
  - SOPC_PE CONTROLLER_V1 card
  - SOPC Peripheral Board
SOPC DESIGN FLOW

1. **Analyze system requirements**
2. **SOPC Builder**
   - Application software
   - BSP Project
3. **Hardware Project**
   - Off-the-shelf IP Cores
   - Nios II Eclipse IDE
   - Hardware information file (.sopcinfo)
   - FPGA config File
   - Download to target FPGA
4. **Software Project**
   - Off-the-shelf drivers
   - Nios II Eclipse IDE
   - Hardware information file (.sopcinfo)
   - .elf file
   - HAL
   - PE IP drivers

**PE system design using Real time/ off-line simulation**

**QUARTUS II**

- **POWER ELECTRONICS LIBRARY (PE Specific IPs)**
- **Block Diagram/HDL File**
- **PI**
- **PWM**
- **JTAG**
- **SYSID**
- **NIOS II CPU**
- **RAM**
SOPC Hardware Project

- Drag and drop approach

ALTERA SOPC Builder / Qsys

Processor (Block view)

For Xilinx - Vivado
Application Software Design

- NIOS II Eclipse IDE – NIOS II EDS
- Supports Assembly, C/C++
  - HAL/MicroC/OS II
    - HAL
      - Confirming POSIX Standards
      - Vendor - Altera
- BSP is generated from ‘.sopcinfo’ file and loads the required drivers from library
Control blocks are accessed through custom instructions/custom functions

- **Fixed Instruction**
- **Simple and easily understandable by the user**

For the PI Controller

- initialize PI module with Kp and Ki Registers
  
  \[
  \text{hw\_init\_pi(PI\_PEG\_V1\_0\_BASE, Kp\_val, Ki\_val)};
  \]

- enable/disable PI Module
  
  \[
  \text{hw\_enable\_pi(PI\_PEG\_V1\_0\_BASE,PI\_Cntrl\_Reg)};
  \]

- input to the PI module
  
  \[
  \text{hw\_pi\_in(PI\_PEG\_V1\_0\_BASE,error)};
  \]

- output from the PI module
  
  \[
  \text{pwm\_input= hw\_pi\_read\_output(PI\_PEG\_V1\_0\_BASE)};
  \]
Custom instructions/Custom functions

- Good Readability
- Number of lines are less
- Faster execution
Using ModelSim

- Can verify SOPC Hardware and application software together - *Hardware Software – Co verification*

**Hardware Signal Flow**

**JTAG terminal**
Technology Deliverables

- Soft processor Integrated **controller design procedure** and user manual

- **HDL control library elements** for hardware accelerators - (PI controller, Filters, PLL, PWM, Phase Transformations etc)

  ➢ **19 PE Specific IPs**

- **Application evaluation report using** SOPC standard (v1.0) controller
  - Voltage Mode control of DC-DC Boost converter
  - AC drive – VVVF AC Induction Motor – Single core and Multi core
  - FOC of Induction Motor (As HiL in FSS miniature)

www.nampet.in
Development Activities
DC-DC converter control using SOPC controller

[Diagram showing the control system with components like JTAG Debug module, Nios II Processor core, System ID, On-chip RAM, Serial Flash Interface, Flash memory, PWM, PI, Serial ADC Controller, Serial DAC Controller, SCLK, CS, TXD, RXD, and output voltage.]
- PI controller is a Hardware module to the CPU
- Accessed by custom functions
  
  \[
  \text{eg: } - \text{ pi\_in(PI\_PEG\_V1\_0\_BASE, error);} \\
  \text{pwm\_input = pi\_read\_output(PI\_PEG\_V1\_0\_BASE);} \\
  \]
DC-DC Converter control

Test setup of DC-DC Converter

Bench Marking

<table>
<thead>
<tr>
<th>Application</th>
<th>TMS320F2812</th>
<th>SOPC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boost Converter</td>
<td>6 µs(CPU clock-150 MHz)</td>
<td>800 ns(CPU clock-100 MHz)</td>
</tr>
</tbody>
</table>

Output voltage(pink) and load current(green) during softstart

Inductor current (green) and PWM Pulse (blue) in steady state
AC Drive control

Diagram showing the AC Drive control system with the following components:
- Dc-Dc Boost
- VSI
- IM
- Vdc
- Va, Vb, Vc
- Vphα, Vphβ
- Sine - Cos
- Look up table
- θ
- Wset
- Vph
- Vp

Diagram also includes a graph showing the PWMs top switches of the 3phase inverter with Vdc, Vph, and Vphβ.

AC Drive assembled in CDAC
AC Drive control

Plot of Va, Vb and Vc values from NiosII through alpha_beta_abc IP

\( \alpha-\beta \) to abc, Integrator are in Hardware

The C code implementation of \( \alpha-\beta \) to abc, took - 600ns

alpha_beta_to_abc IP took only 300ns to perform the conversion

Main loop execution time = 2 us
AC Drive application in Multi-core

Processor 1

Dc-Dc Boost

Control

Processor 2

VSI

IM

Boost converter control in one CPU Core and Inverter control in other core

Vdc

Vbatt

Va

Vb

Vc

Vphα

Vph β

Vph

θ

∫

Vp

fr

f

Wset
AC Drive application in Multi-core

Processor configuration in Multi-core

Processor I
- Boost converter control
- OnChip Memory
- Mutex
- sysid
- JTAG_UART

Processor II
- Inverter control
- PWM for inverter
- JTAG_UART

FPGA

Main loop execution time is 1.5µs

SOPC View in Multi core

PWM for Boost converter from Processor 1

PWM for Inverter from Processor 2 (a phase and b phase - Top switches)

Processor configuration in Multi-core SOPC View in Multi-core
FOC on SOPC

Red blocks – are hardware IPs

Main loop execution = 10 us in 100MHz
For TMS 320F2814 = 22 us in 150MHz
Design flow

Off the Shelf CPU Cores - NIOS II, Mico Blaze, Leon3 etc.
Micro processor peripherals :- JTAG, Hardware Multiplier, memory interfaces, sys_id etc.

Communication Interfaces :- CAN, USB etc.
Design flow

PE IPs: PWM, PI, abc_to_dq, dq_abc, ADC/DAC controllers, Integrators etc.
For Multiplication - MUL A,B  Similarly For PI Controller IP
hw_pi_in(PI_ID,error);
out = hw_pi_read(PI_ID);

Number of lines in the application code is less
Soft IP core: Design Advantages

- Higher level of design reuse
- Re-configurability
- Reduced obsolescence risk
- Simplified design update or change
- Lesser overhead in control software design
- Parallel Processing
- Improved Time to market
- Single to multi-core flexibility
THANK YOU

ajeesh@cdac.in