

20 YEARS of

**ALTERA**®

INNOVATION



# **SOPC**

# **WORLD**

## 2003



**SOPC**  
**WORLD**  
2003

# Developing Custom Instructions & Peripherals for Embedded Processing

# Agenda

- Embedded Systems
- MP3 Player Demonstration
- SOPC Builder & Avalon™ Switch Fabric
- Custom Peripherals
- Developing Custom Instructions for the Nios® Embedded Processor
- Applications & Examples

# What Is an Embedded System?

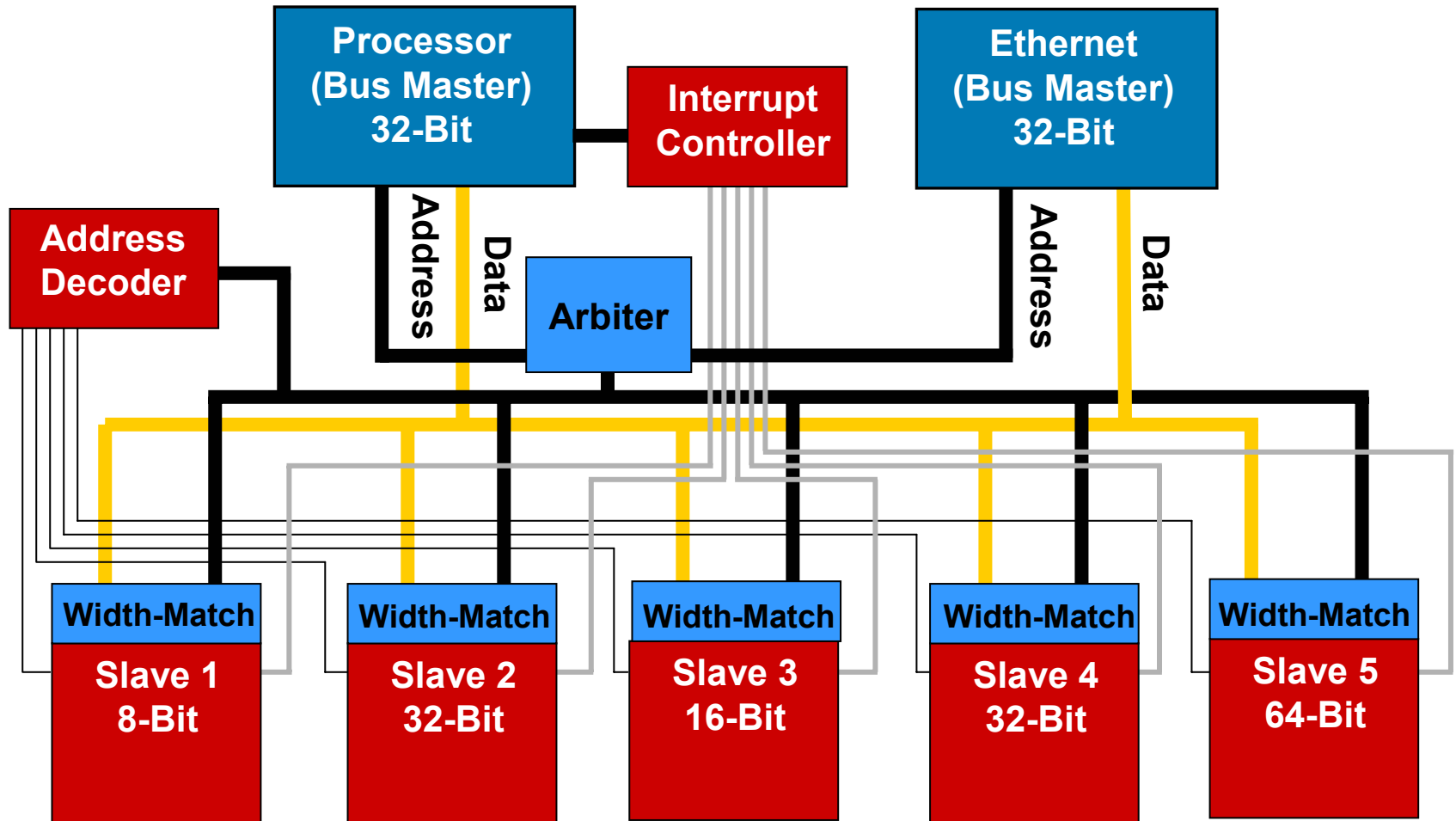
- Special Purpose Computer
- Consists of Both Hardware & Software
  - Usually Contains At Least One Microprocessor
  - May Utilize An Operating System
  - All I/O Is Task-Specific
- Employs Several Function-Specific Blocks
- Used Where Full-Size Computers Are
  - Too Big
  - Too Expensive
  - Too Generic in Purpose

# Traditional Design Method

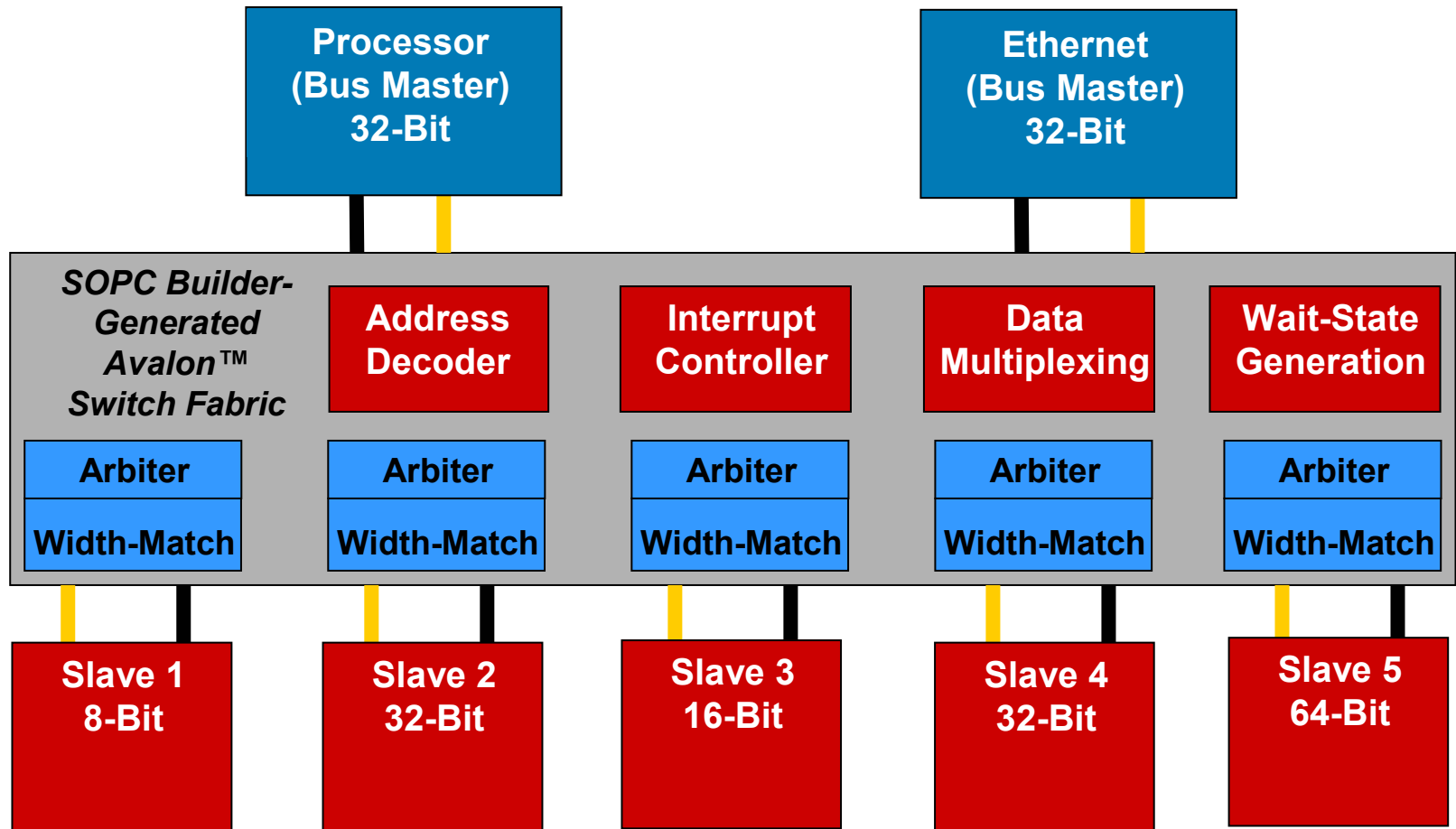
## Typical Order of Steps

1. Select System Controller (Processor)
2. Select Available Peripherals
3. Define Custom Logic
4. Adapt to Bus Standard
5. Develop Decode Logic
6. Multiplex Data Paths
7. Design Arbitration Logic
8. Create Interrupt Scheme
9. Develop Timing Logic

# System Implementation



# What If...



# What If...

## Typical Order of Steps

1. Select System Controller (Processor)
2. Select Available Peripherals
3. Define Custom Logic
4. ~~Adapt to Bus Standard~~
5. ~~Develop Decode Logic~~
6. ~~Multiplex Data Paths~~
7. ~~Design Arbitration Logic~~
8. ~~Create Interrupt Scheme~~
9. ~~Develop Timing Logic~~





**SOPC**  
**WORLD**  
2003

# Build MP3 Player

*Demonstration*



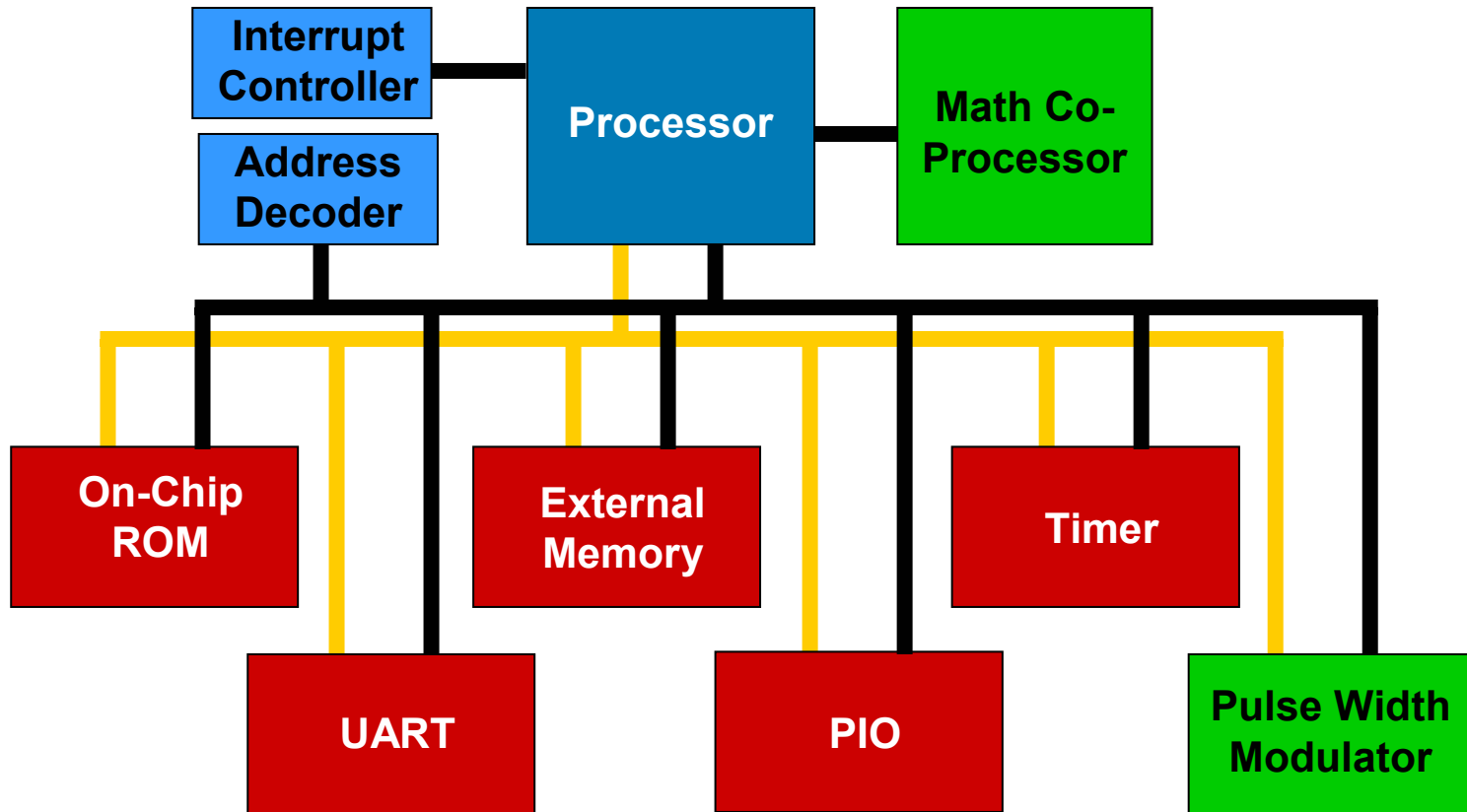
**SOPC**  
**WORLD**  
2003

# What Did We Just Do?

*The Power of SOPC Builder*

# MP3 Components Needed

## *Complete MP3 Player System*



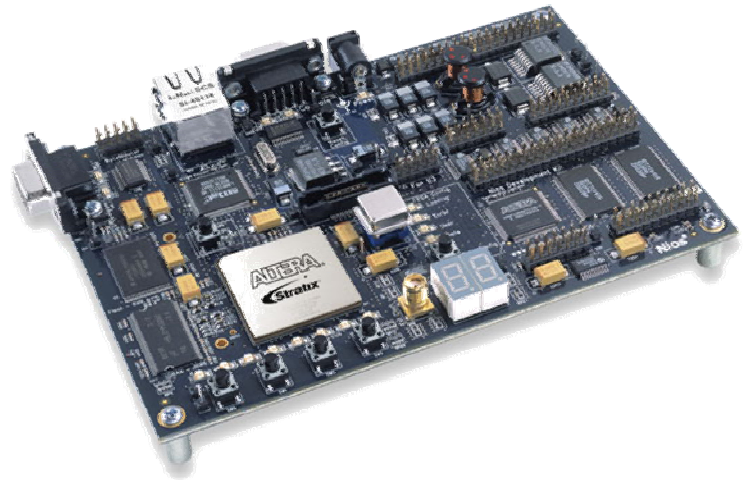
# The Nios Microprocessor

## ■ Soft-Core Microprocessor from Altera®

## ■ Features

- Basic RISC Processor
- Harvard Architecture
- Multi-Stage Pipeline
- 16-or 32-Bit Data Path
- 16-Bit Instruction
- 64 Prioritized Interrupts
- Custom Instructions

## ■ Optimized for Altera FPGAs



# SOPC Builder-Ready IP

## ■ SOPC Builder-Ready Certification Requirements

- Avalon / AHB Compatible Interface
- OpenCore® Evaluation Support
- Evidence of Functional System Verification
- Successful Generation & Compilation of SOPC Builder System
- Plug-&-Play Compatibility with SOPC Builder



## ■ Examples of What's Available

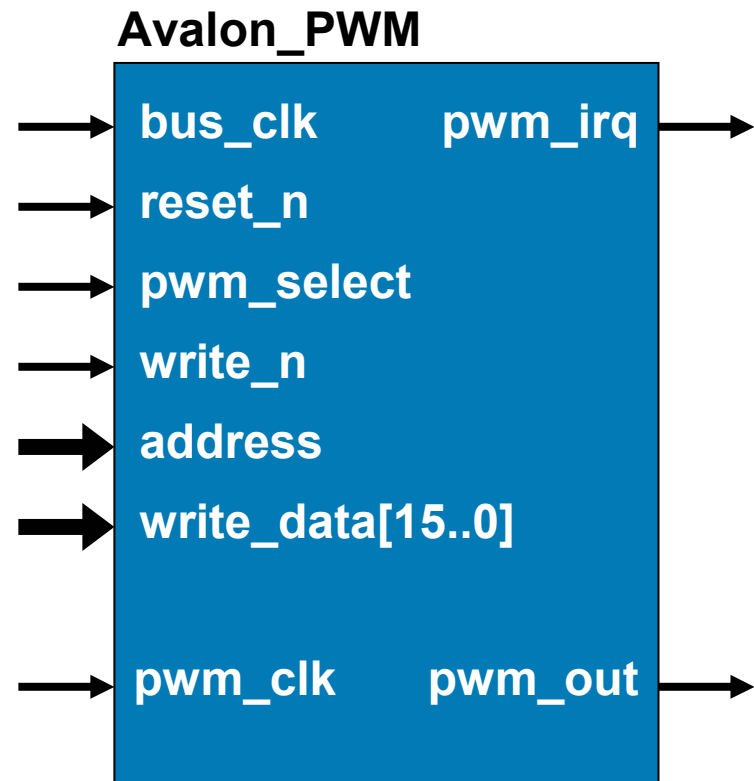
- Processors
- PCI, Ethernet & Communication Cores
- Memory & Memory Controllers
- USB, I2C, SPI

***It's Also Easy to Create Your Own***



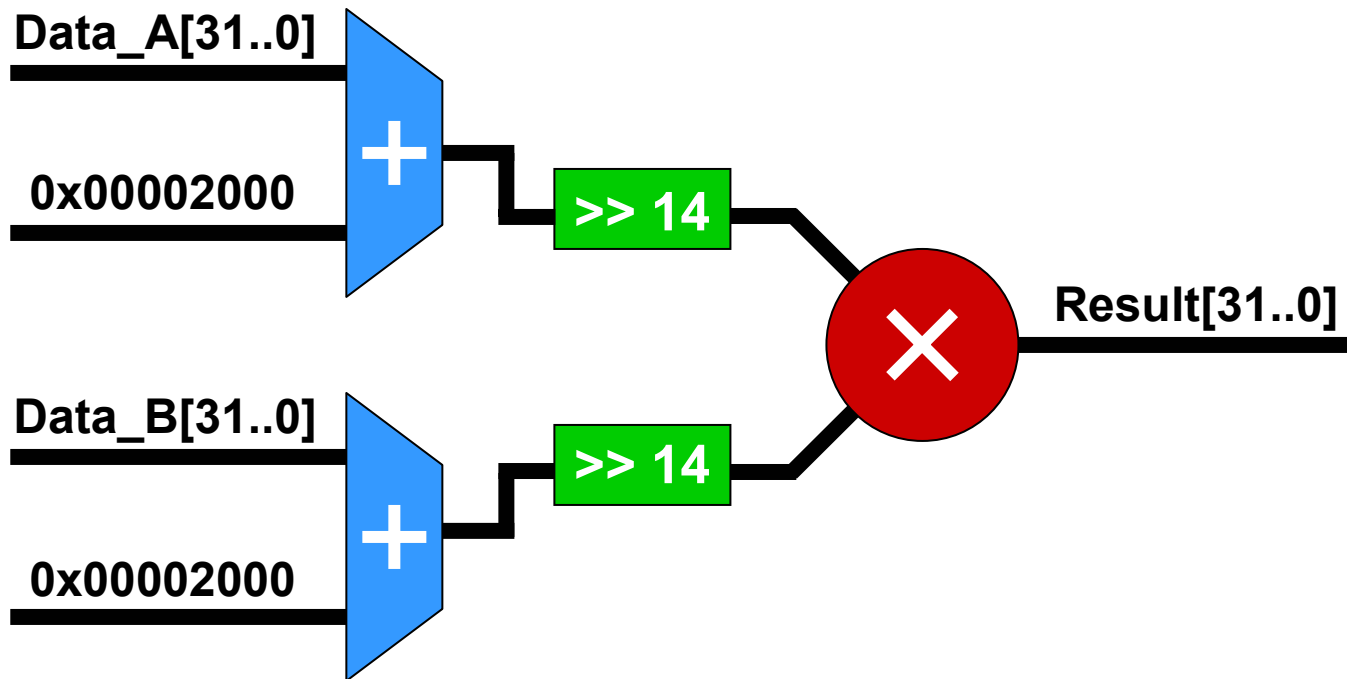
# Custom PWM Peripheral

- Audio PWM
- Verilog HDL
- Avalon Interface
  - Use Only Needed Signals
  - Provides Access to:
    - Period
    - Pulse Width
- External Interface
  - PLL Clock Input
  - PWM Output

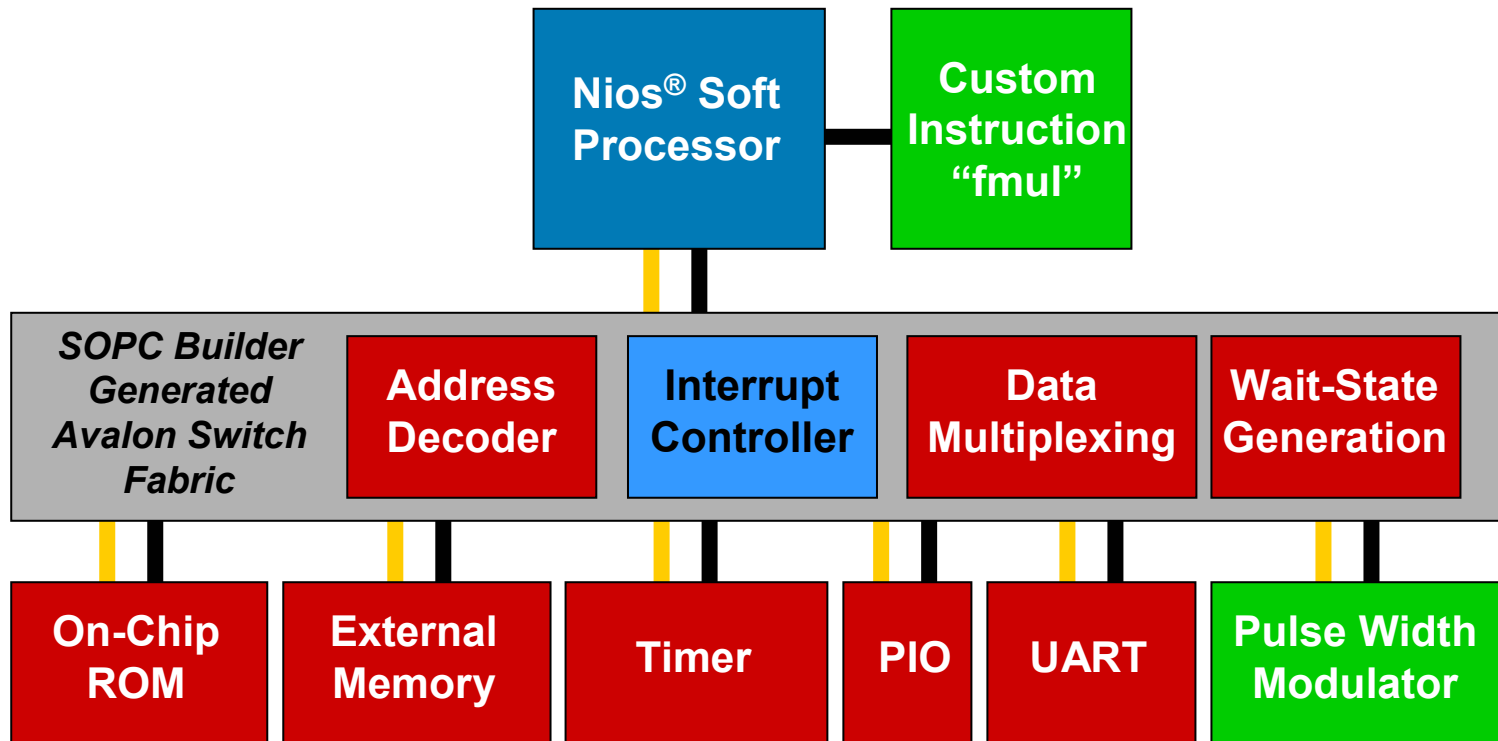


# Custom Instruction: fmul

*Function Replaced with Hardware*



# MP3 Player Result



***SOPC Builder Creates  
All the Interconnect***





**SOPC**  
**WORLD**  
2003

# What Can SOPC Builder Do for My System?

*A Closer Look at the Tool*

# How SOPC Builder Helps

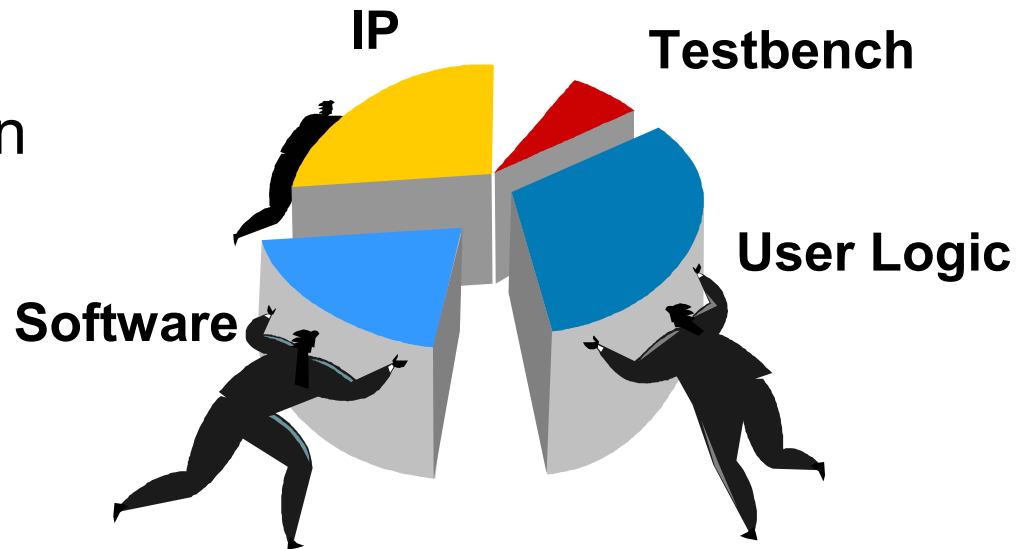
## ■ Automates Block-Based Design

- System Definition
- Component Integration
- System Verification
- Software Generation

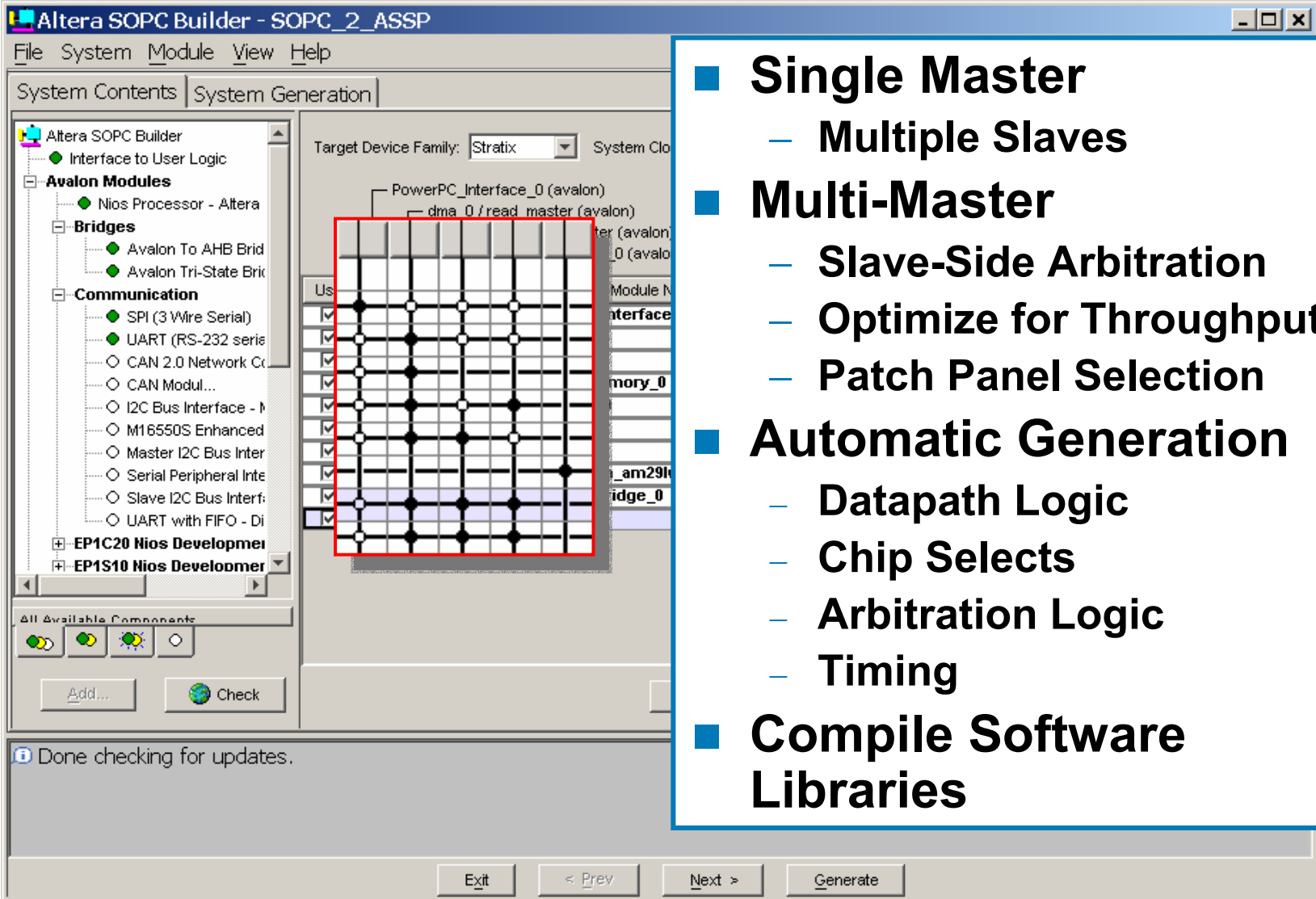
## ■ Fast & Easy

## ■ Supports Design Reuse

- Third-Party Intellectual Property (IP) Cores
- Internally Developed Peripherals



# SOPC Builder – System Integration



Altera SOPC Builder - SOPC\_2\_ASSP

File System Module View Help

System Contents System Generation

Altera SOPC Builder

- Interface to User Logic
- Avalon Modules
  - Nios Processor - Altera
  - Bridges
    - Avalon To AHB Bridge
    - Avalon Tri-State Bridge
  - Communication
    - SPI (3 Wire Serial)
    - UART (RS-232 serial)
    - CAN 2.0 Network Controller
    - CAN Module
    - I2C Bus Interface - Master
    - M16550S Enhanced
    - Master I2C Bus Interface
    - Serial Peripheral Interface
    - Slave I2C Bus Interface
    - UART with FIFO - Dual
- EP1C20 Nios Developer Kit
- EP1S10 Nios Developer Kit

Target Device Family: Stratix

PowerPC\_Interface\_0 (avalon)

dma\_0/read\_master (avalon)

Us

Module N

Interface

mory\_0

am29k

idge\_0

All Available Components

Add... Check

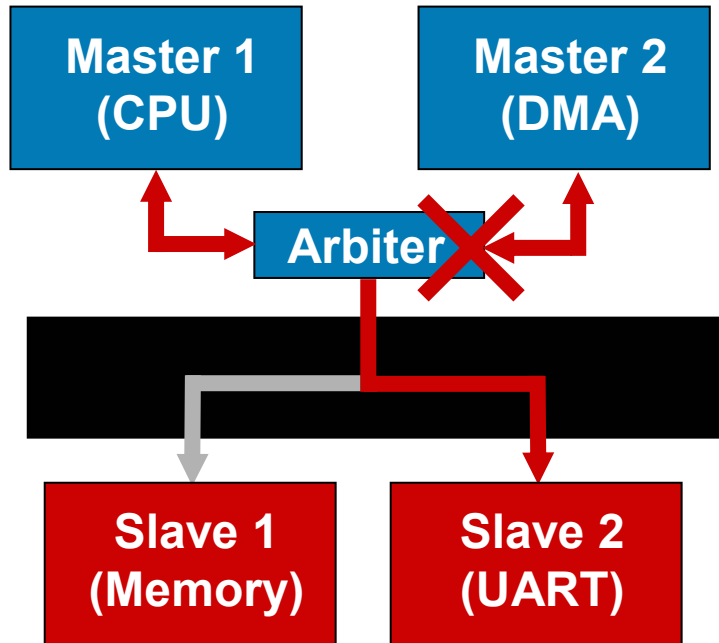
Done checking for updates.

Exit < Prev Next > Generate

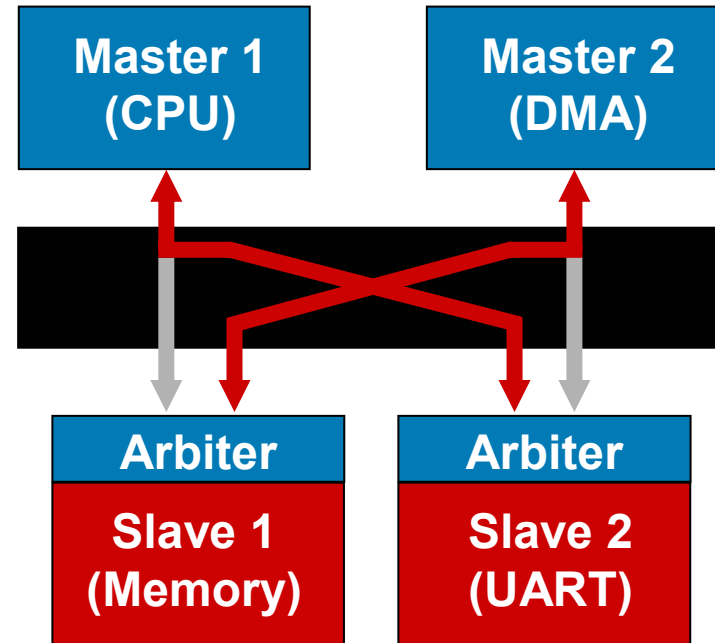
- **Single Master**
  - Multiple Slaves
- **Multi-Master**
  - Slave-Side Arbitration
  - Optimize for Throughput
  - Patch Panel Selection
- **Automatic Generation**
  - Datapath Logic
  - Chip Selects
  - Arbitration Logic
  - Timing
- **Compile Software Libraries**

# Slave Side Arbitration

## *Bus Arbitration*



## *Slave Side Arbitration*

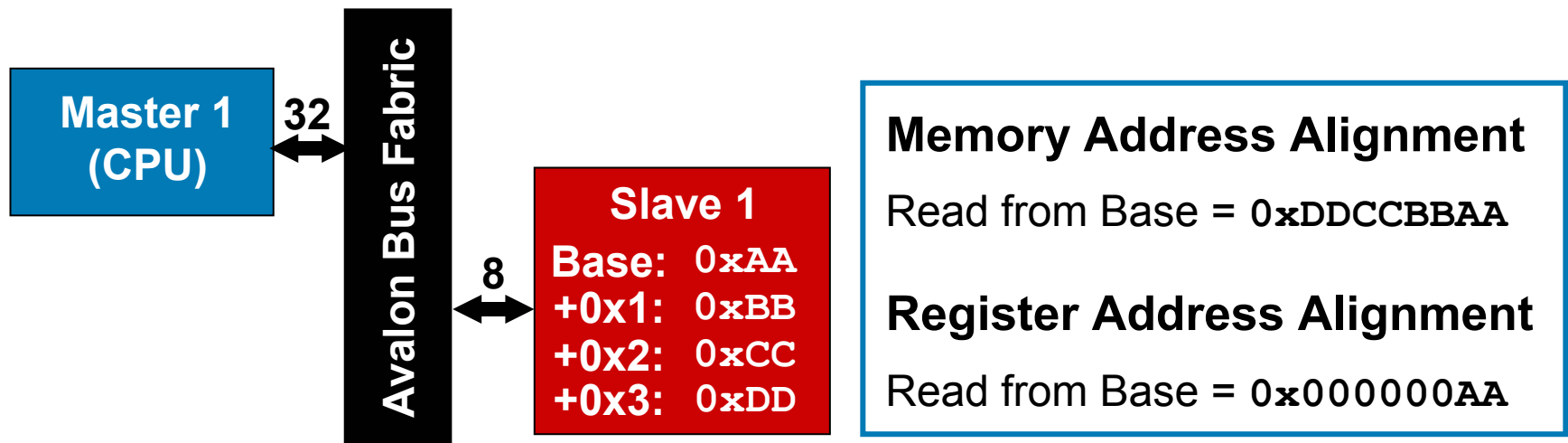


***Higher System Throughput & Efficiency***

# Dynamic Bus Sizing

## ■ Narrow Slave

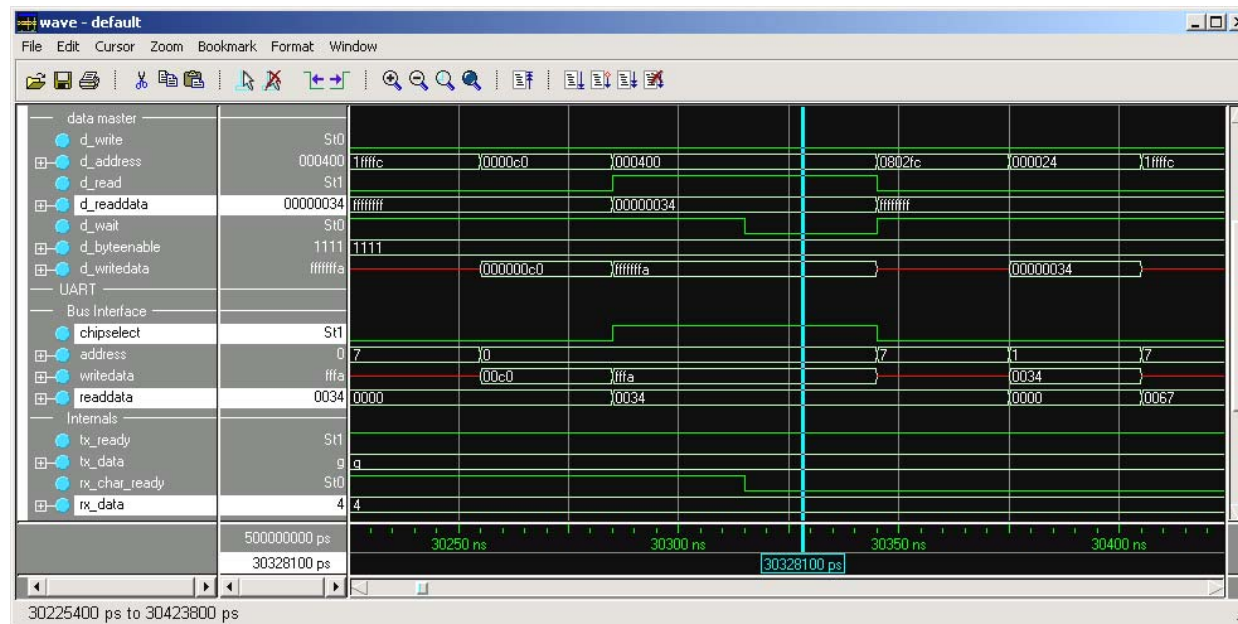
- Can be Translated to Master's Width
- Or Upper Bits Can Be Masked
- Your Choice – Transparent to Master



***Done Automatically  
by SOPC Builder!***

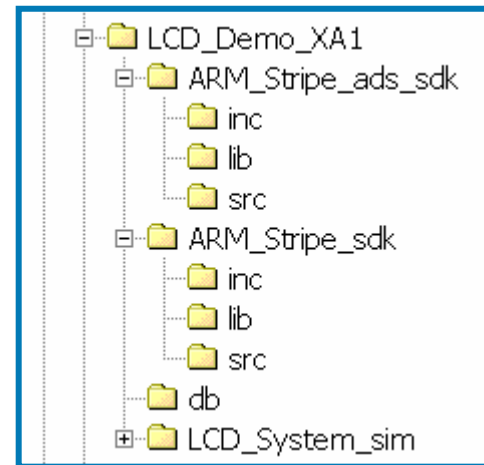
# SOPC Builder - System Verification

- Automated Simulation Generation
  - Generate Complete System Simulation Model
  - Generate Testbenches
  - Setup Project Environment
- Immediate Simulation of Hardware & Software



# SOPC Builder Software Support

- Software Development Kit (SDK) Automatically Generates
  - Headers (INC)
    - Memory Map
    - Register Declarations
  - Libraries (LIB)
    - Runtime
  - Source (SRC)
    - Supplied by Peripherals
    - Examples for Processor
- Uses Software Compilers
  - Compile Runtime Libraries
  - Generate Memory Contents
  - Hardware & Software Simulation
- Advanced Software Components
  - Network Protocol Library
  - RTOS Components



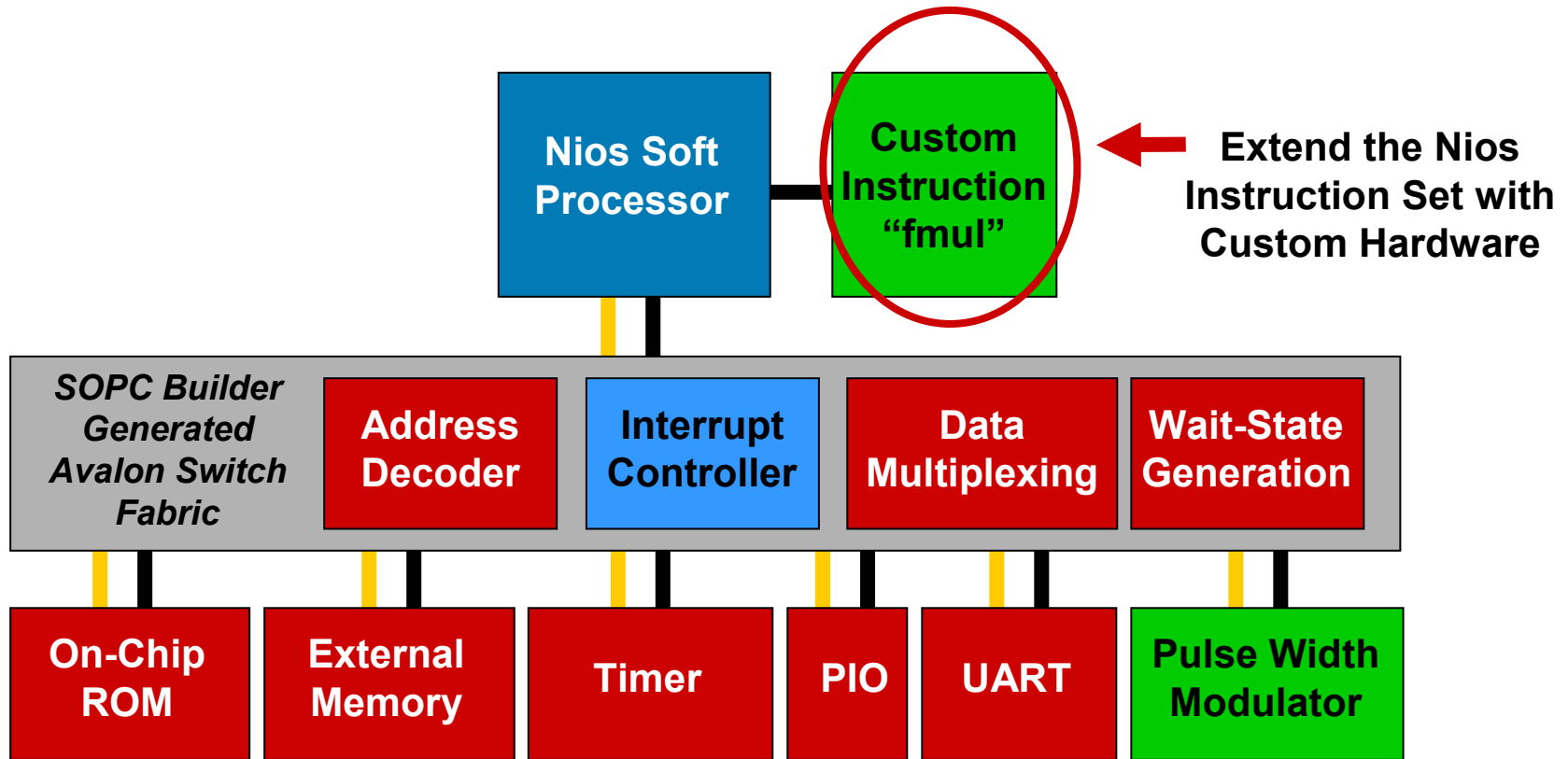
Function	Module	Offset	Address
Reset Location		0x0	
Vector Table (256 bytes)		0x0	
Program Memory			
Data Memory			
Primary Serial Port (printf, GERMS)			
Auxiliary Serial Port			

System Boot ID:  (25 chars max)

Use	Name	Description
<input checked="" type="checkbox"/>	Altera Plugs TCP/IP Networking Library	Lightweight, RTOS-independent network...

# Nios Processor in SOPC Builder

*Allows You to Create A Custom Instruction*





# Custom Instruction - Performance

- Replace Library Call with Custom Instruction

```
#define mad_f_mul ( x ,y ) nm_fmulo ( x, y )
```

- Dramatically Accelerate Software Algorithms

Category	Number of Cycles to Complete mad_synth_frame()	Number of Logic Elements Used
CPU with Hardware Multiplier	1,279,000	n
CPU with fmul (Remove Hardware Multiplier)	293,000	n + 100

***Entire Function Sees 4x  
Improvement Just from fmul  
Acceleration***



**SOPC**  
**WORLD**  
2003

## Run MP3 Demo

*System Created in Minutes*



**SOPC**  
**WORLD**  
2003

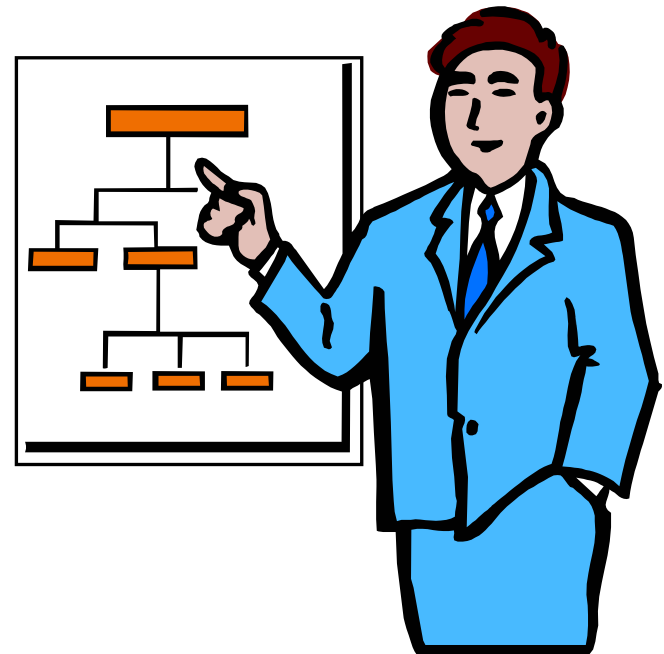
## How Does It Work?

*Looking Under the Hood*

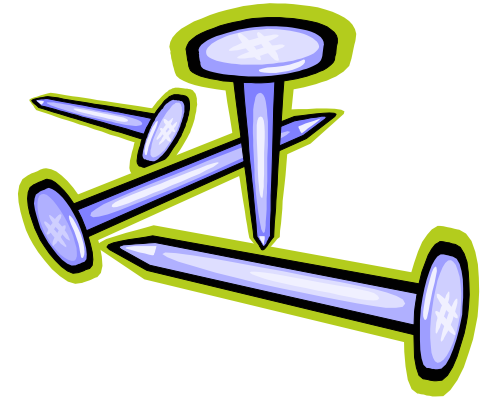
# Avalon Switch Fabric

## ■ Avalon – SOPC Interface Standard

- Backbone of SOPC Builder
- Easy to Use Interface
- Parameterized
- Optimized for Altera FPGAs
- Introduced in Fall 2000
  - Native Bus for Nios Processor
- Has Since Expanded
  - Altera & AMPP<sup>SM</sup> IP Cores
  - Customer-Defined Peripherals
  - 100+ Cores Planned for 2003



# Bus Interface Standards



- Why Bus Standards Are Used
  - Flexibility
    - Provides Wide Range of Capabilities in One Package
    - Guarantees Compatibility
  - Bus Designed to Handle All Contingencies
- Pitfalls of Typical Bus Standards
  - Must Be Complex to Support Everything
  - Even Small Peripherals Must Fully Comply

***Sledgehammer Is Used  
for Every Size Nail***

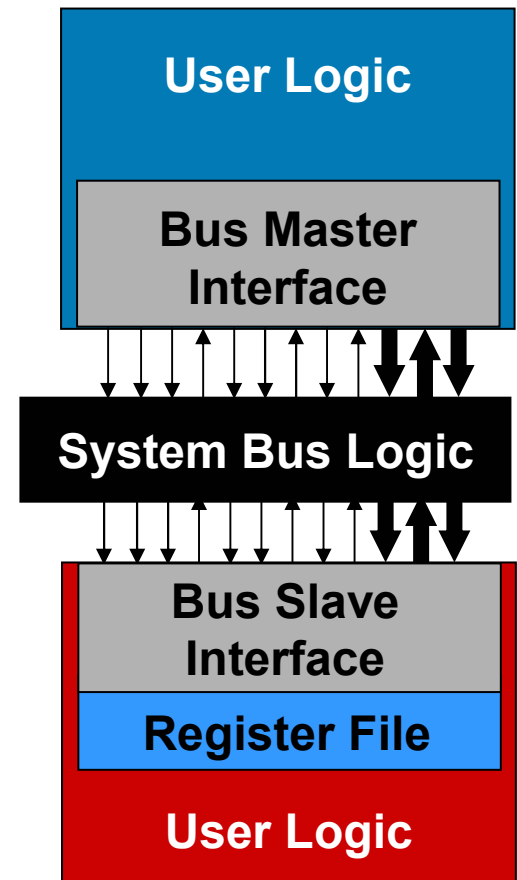
# Avalon Switch Fabric Is Different

- Fabric Custom-Generated for Peripherals
  - Contingencies on per-Peripheral Basis
  - System Is Not Burdened by Bus Complexity
- SOPC Builder Automatically Generates
  - Arbitration
  - Address Decoding
  - Data Path Multiplexing
  - Bus Sizing
  - Wait-State Generation
  - Interrupts

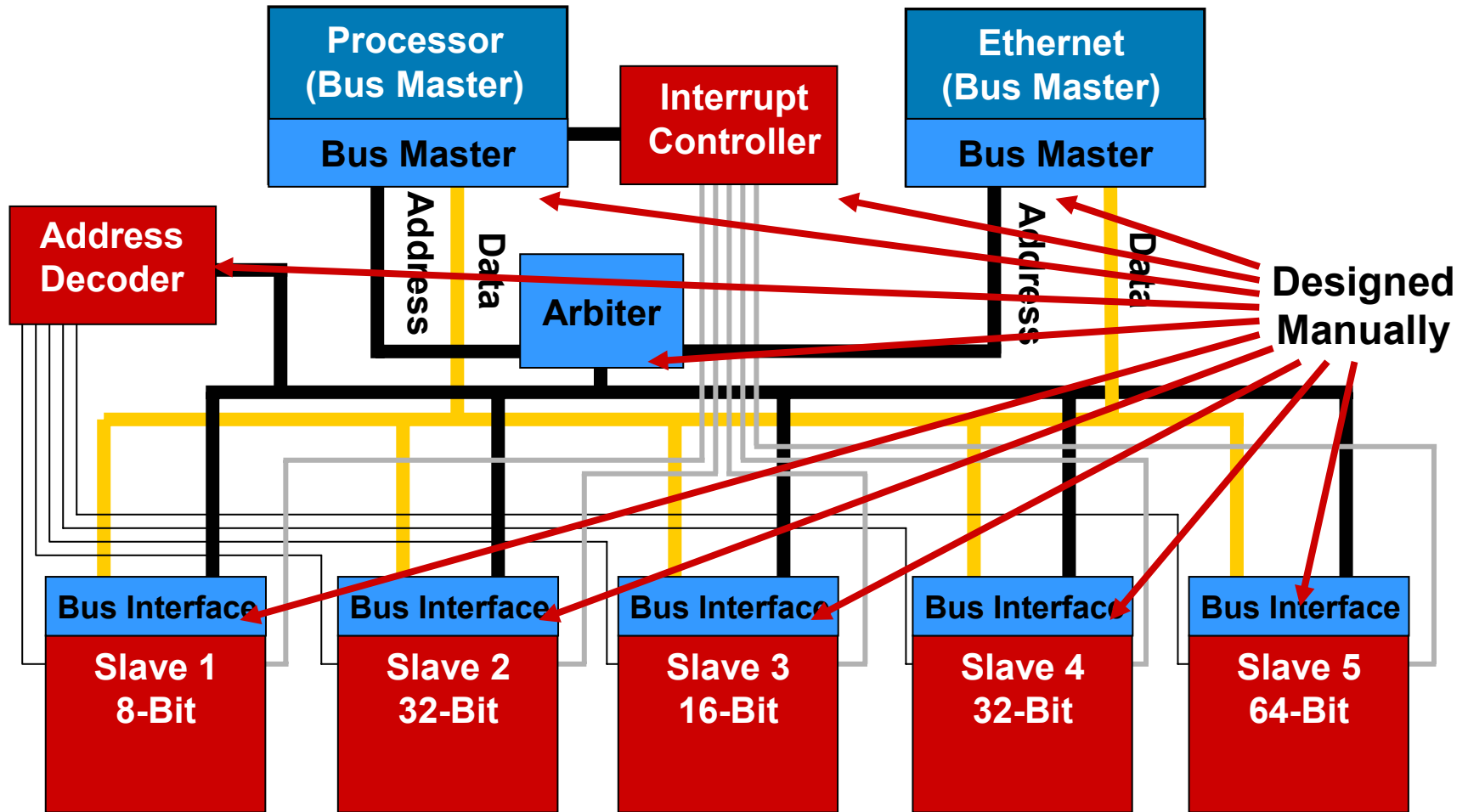
# Traditional Bus Master / Slave

- Must Comply Fully to Chosen Bus Standard
  - Bus Standard Adds Complexity
  - Consumes Resources
  - Designed in Reverse
    - Design Starts at Bus Interface
    - Back-End Adapted to Comply

***Result = Non-Optimal Implementation***



# Traditionally Designed System

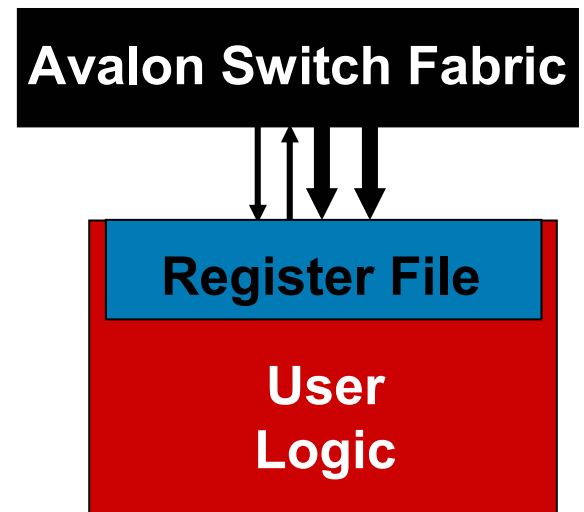




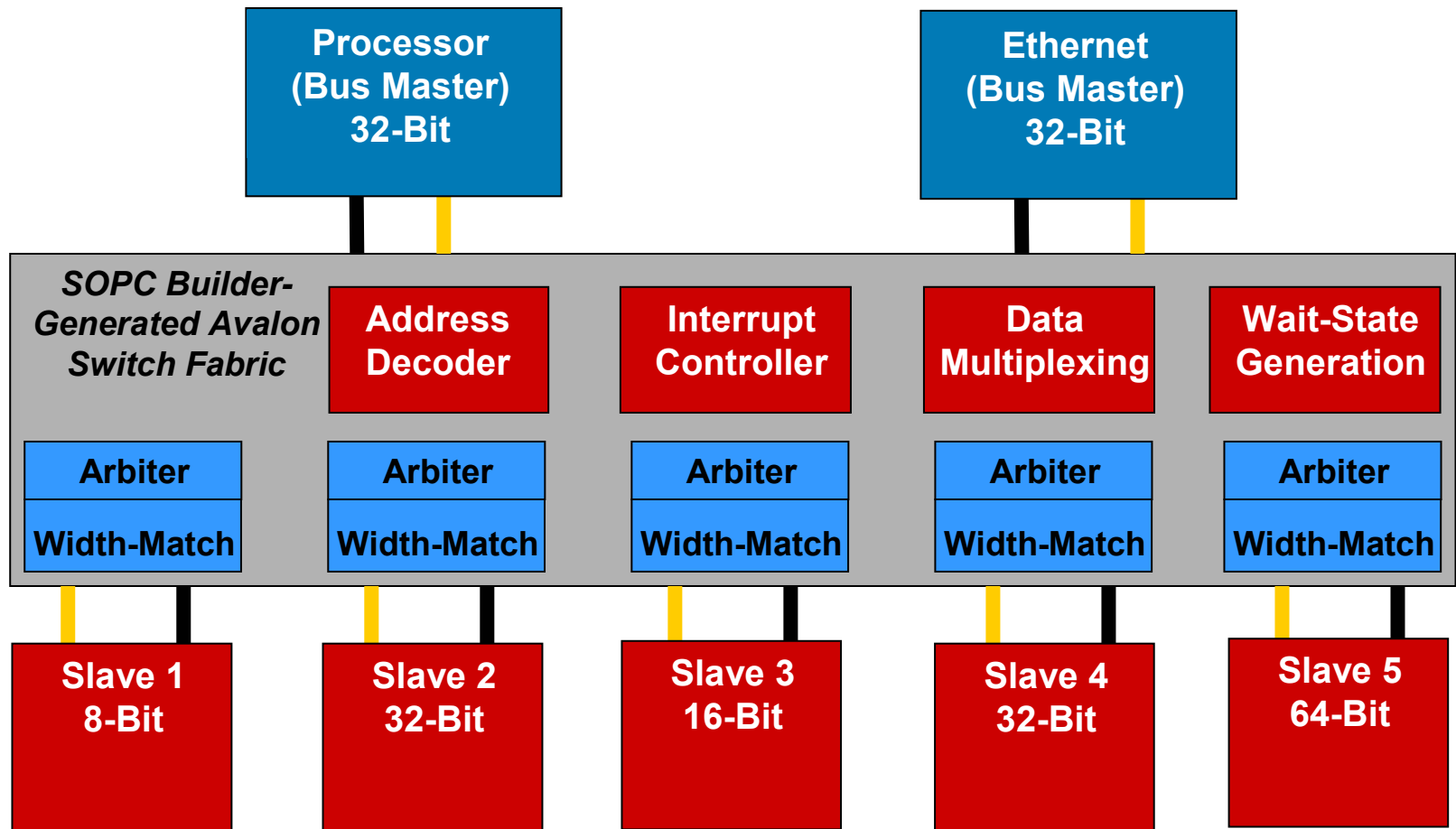
# Avalon Slave

- No Need to Worry about Bus Interface
- Use Interface Optimal for Nature of Peripheral
- Implement Only Signals Needed
- Avalon Switch Fabric Adapts to Peripherals
- Timing Automatically Handled
- Fabric Created for You
- Arbiters Generated for You

***Concentrate Effort on  
Peripheral Functionality!***



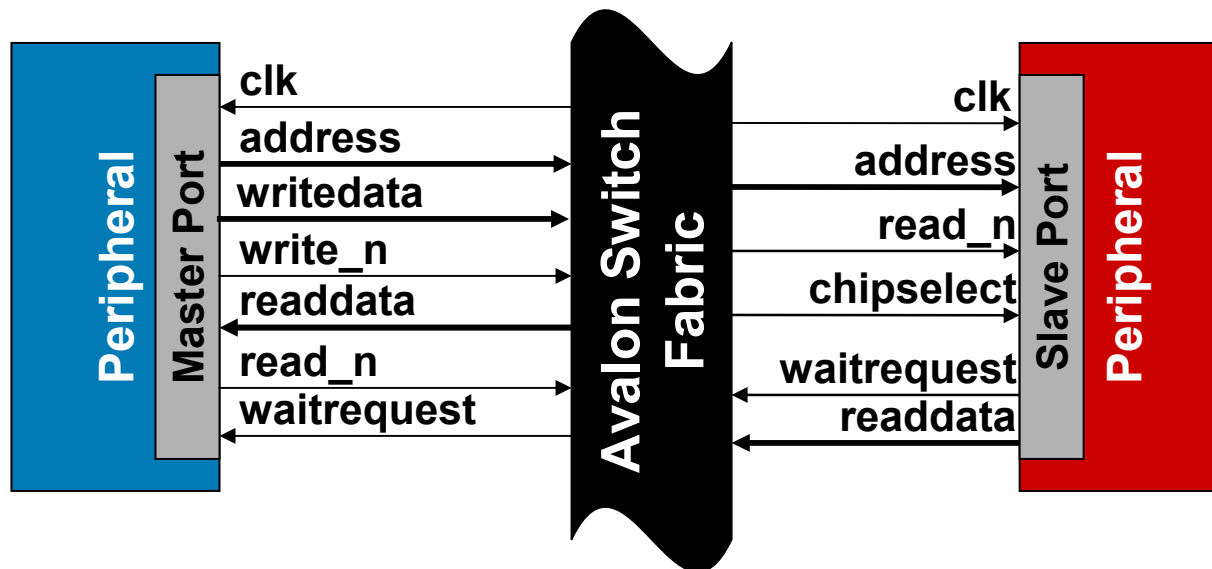
# Avalon System



***Designer Only Needs to  
Worry About Peripherals***

# Example Avalon Peripherals

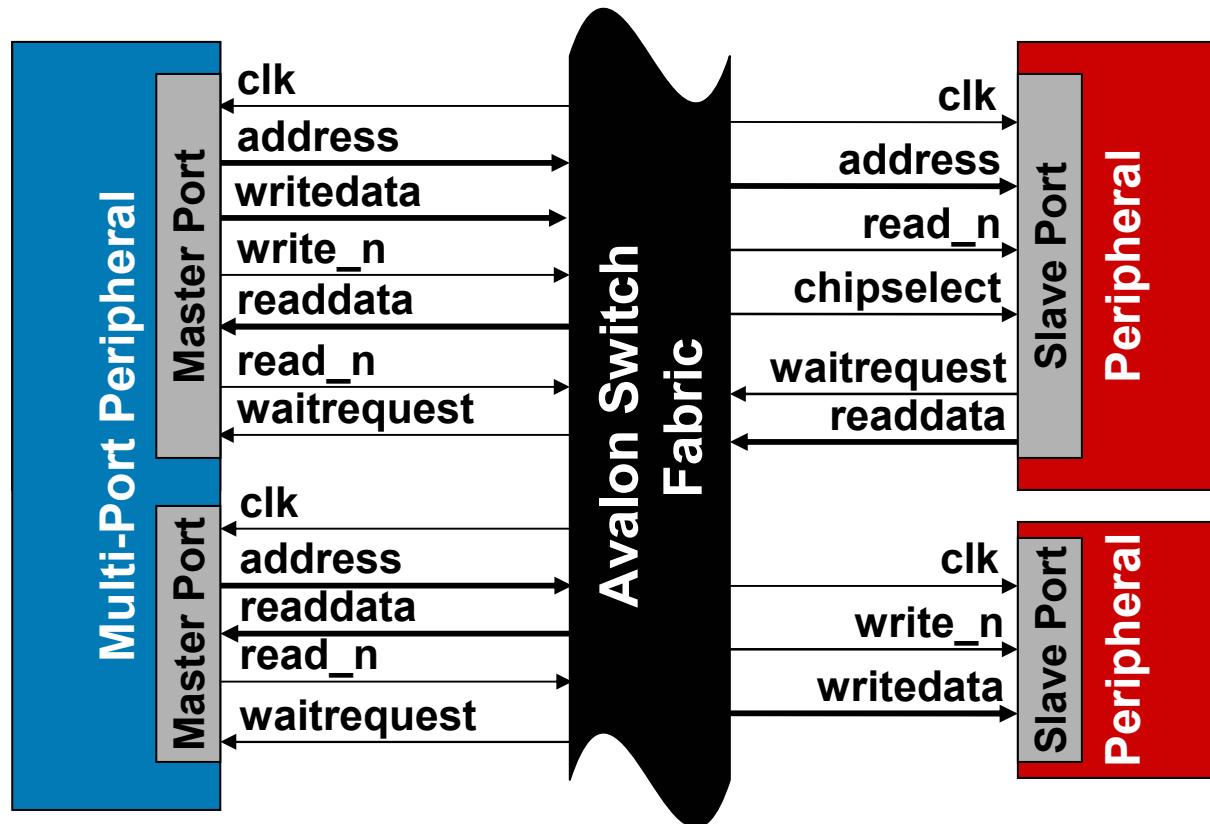
- Master Peripheral that Can Write & Read
- Read-Only Slave Peripheral with waitrequest



# Example Avalon Peripherals

- Master Peripheral that Can Write & Read

- Read-Only Slave Peripheral with waitrequest



- Read-Only Master Port (e.g., - Status Check Port)

- Write-Only Slave Peripheral (e.g., FIFO Write Port)



**SOPC**  
**WORLD**  
2003

# Creating Custom Peripherals

*How Do I Develop My Own  
Hardware for Use in SOPC  
Builder?*

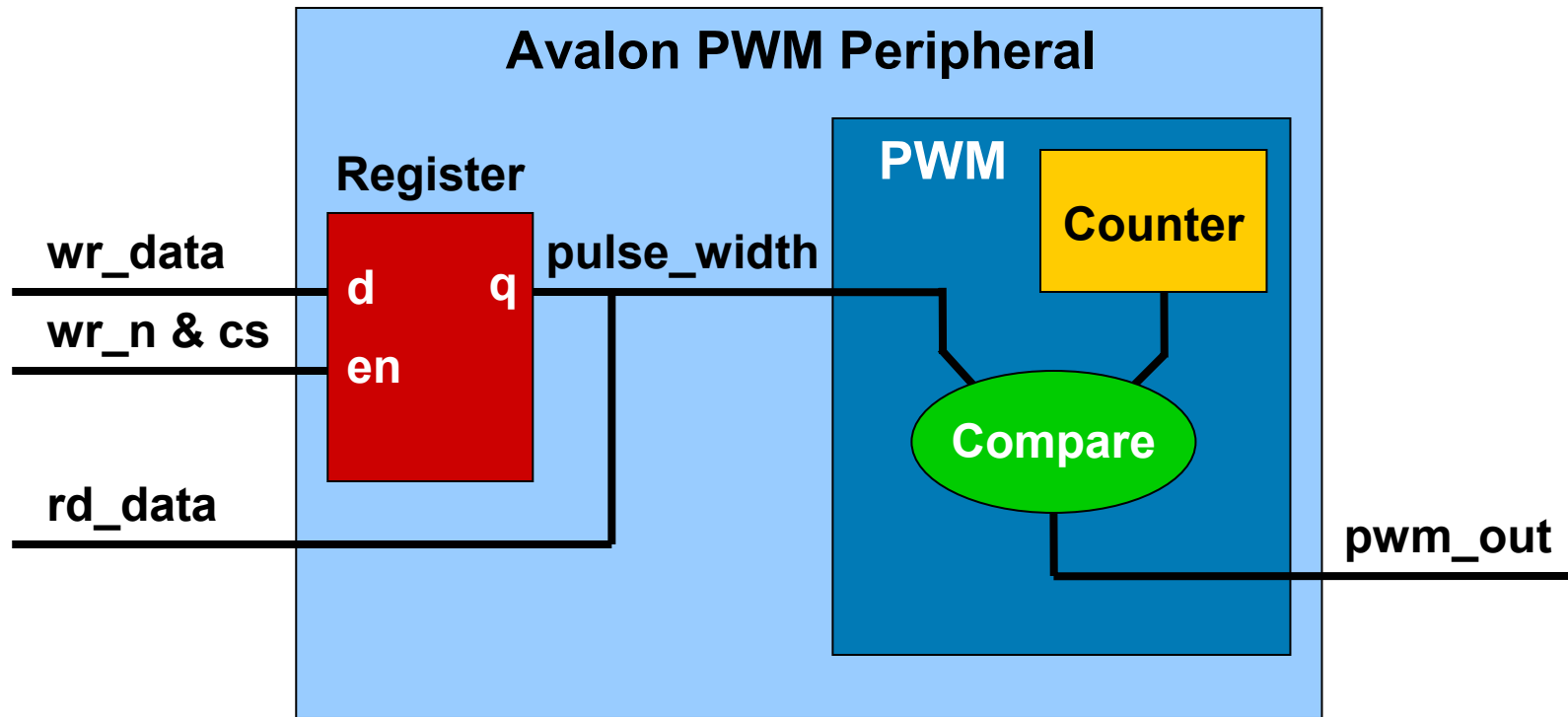
# Reasons for Custom Hardware

- Acceleration
  - Replace Software with Hardware
- Proprietary Functions
  - Algorithms
  - Product Differentiation
  - Design Reuse
- Availability
  - No Such Ready-Made IP



# Creating an Avalon Slave

## Pulse Width Modulator



***Adding Avalon Interface  
Consists of Only Adding a  
Register***

# Creating an Avalon Slave

## ■ PWM Peripheral

- Verilog HDL
- Only 9 Ports

## ■ Dynamic Bus Sizing

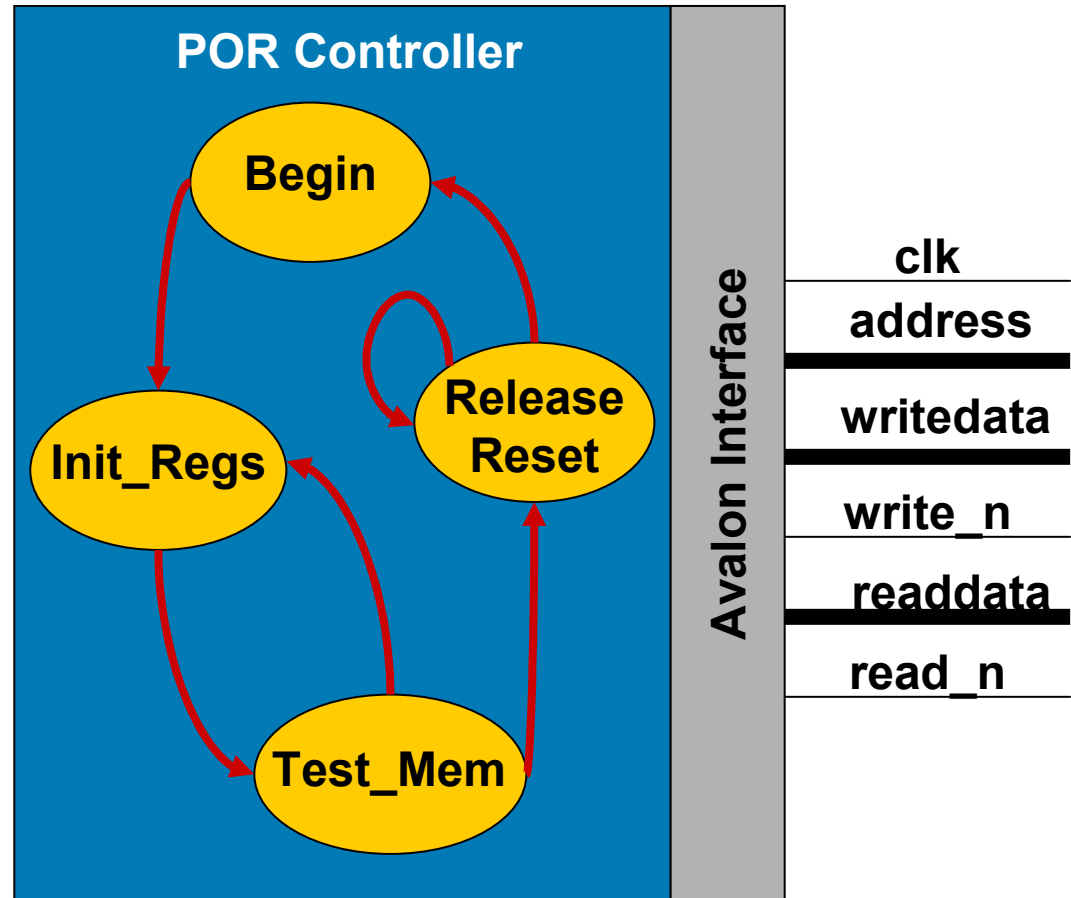
- You Pick Data Width
- Avalon Switch Fabric Adapts
- Register vs. Memory

```
module avalon_pwm (  
    clk,  
    wr_data,  
    byte_n,  
    cs,  
    wr_n,  
    addr,  
    clr_n,  
    rd_data,  
    pwm_out  
);  
  
input  clk;  
input  [31:0] wr_data;  
input  [3:0] byte_n;  
input  cs;  
input  wr_n;  
input  addr;  
input  clr_n;  
output rd_data;  
output pwm_out;
```



# Creating an Avalon Master

- Example
  - POR Controller
    - State Machine
    - Avalon Interface
  - Bring Up System
- Simple & Easy
  - Avalon Master
  - 4-State FSM



# Interface to User Logic

Interface to User Logic - audio\_pwm

Ports | Instantiation | Timing | Publish

Bus Interface Type: Avalon Register Slave

Design Files

☒ Import Verilog, VHDL, EDIF, or Quartus Schematic File

Add... Delete

Top module: pwm.v

Port Information

Port Name	Width	Direction	Shared	Type
wave_clk	1	input		export
reset_n	1	input		reset_n
pwm_select	1	input		chipselect
reg_address	1	input		address
write_n	1	input		write_n
data_from_cpu	16	input		writedata
periodic_irq	1	output		irq
wave_out	1	output		export

Read port-list from files Add Port Delete Port

☒ Hide Advanced Signal Types

AHB Slave's Addressable Space

Address span: 0x100000000 Bits: 32

Cancel < Prev Next > Finish Editing Add to Library

- Publish Custom Hardware As SOPC Builder Component
- Choose Interface Type:
  - Register Slave
  - Memory Slave
  - Avalon Master
- Add Design Files that Describe User Logic
- Automatically Define Port Table from Design Files
- Make Port Changes or Enter Ports Manually

# Interface to User Logic

## ■ Specify Timing Requirements

- Setup
- Hold
- Wait States

## ■ Units

- Time
- Clock Cycles

Interface to User Logic - audio\_pwm

Ports | Instantiation | Timing | Publish

Setup: 0 Wait: 0 Hold: 0 Units: ns

System Clock 66.66 MHz Timing granularity is System Clock cycle

Read Waveforms

data  
addr  
select  
readn 15ns

Write Waveforms

data  
addr  
select  
writen 15ns

Cancel < Prev Next > Finish Editing Add to Library



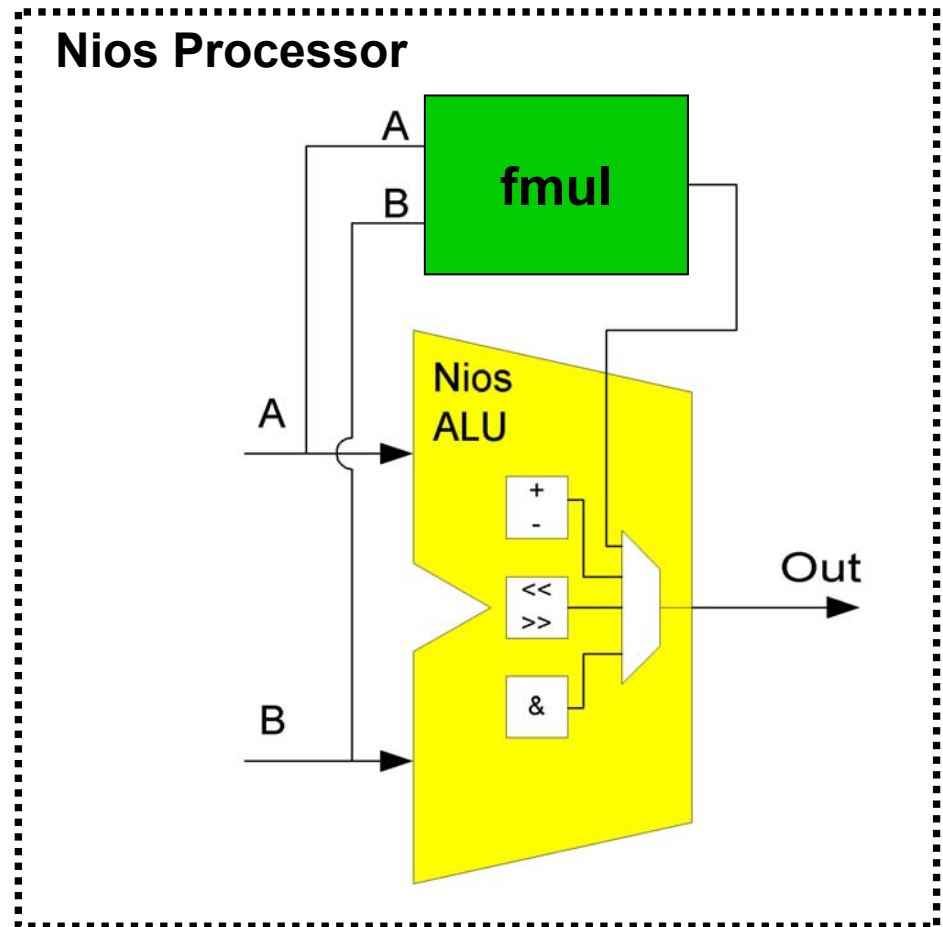
**SOPC**  
**WORLD**  
2003

# Creating Custom Instructions for Nios

*Augmenting Your Embedded  
Processor's Instruction Set*

# Custom Instruction - Definition

- Dramatically Accelerates Software Algorithms Using Hardware
- Extends Nios Instruction Set
  - Up to Five Instructions
- SOPC Builder Development Tool
  - Automatically Adds User Logic to Nios ALU
  - Assigns Op-Code
  - Generates C- & Assembly- Macros



# Custom Instruction - Software

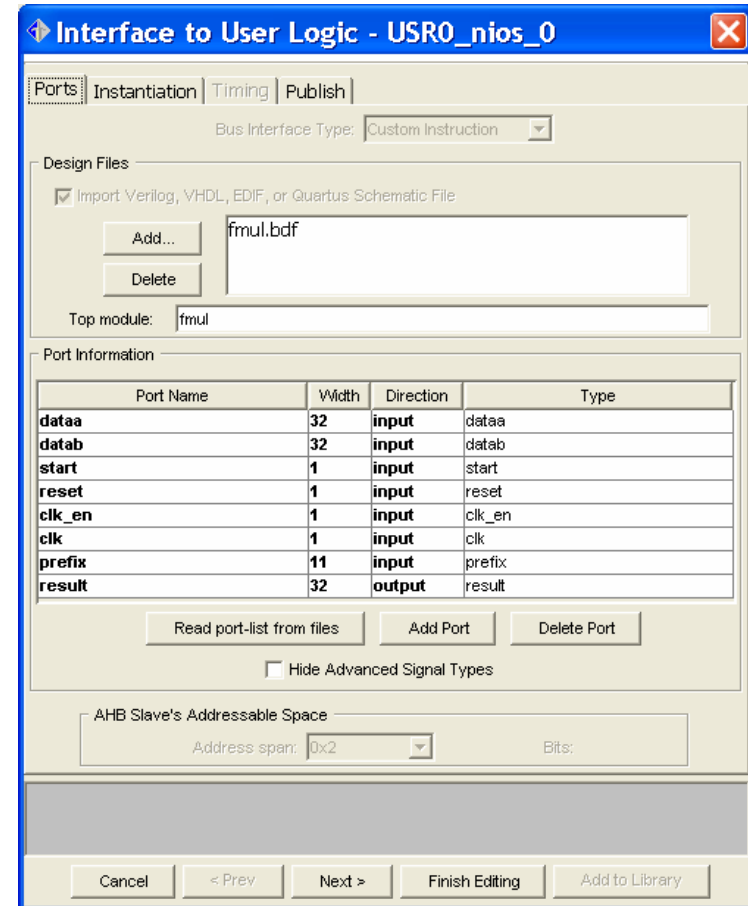
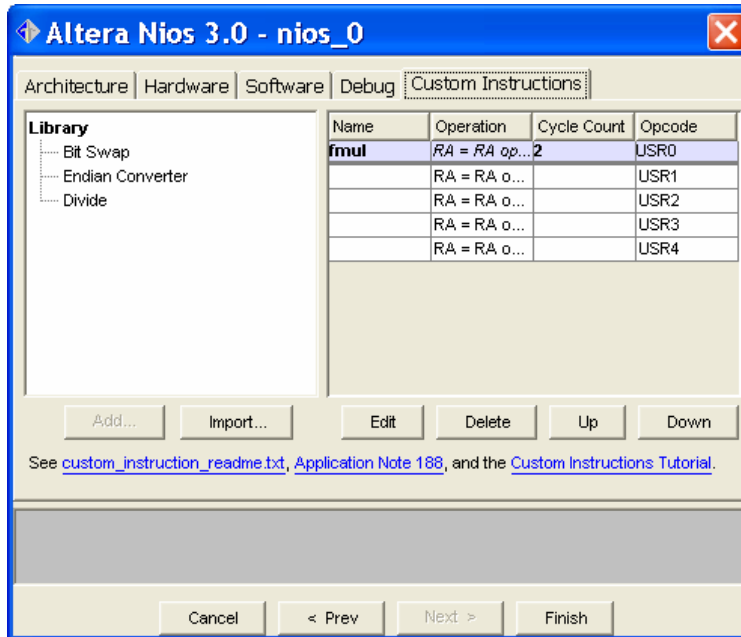
- C Code Macros (include excalibur.h)
  - nm\_<macro\_name> (dataa, datab)
  - nm\_<macro\_name>\_pfx (prefix, dataa, datab)
- Assembly Code
  - Use Opcodes or Assembly Macro

```
LD    %r1, [%L6]           ; Load word at [%L6] into %r1
LD    %r0, [%L2]           ; Load word at [%L2] into %r0
PFX 1                      ; Only needed if using prefix
nm_my_cust_inst %r1        ; Macro calling a Rw opcode, r1 <= r1 "OP" r0
ST    [%L4], %r1           ; %L4 is the pointer, %r1 is stored
```

***Makes Your Custom Instruction Look  
Like a Normal C Function Call***

# Custom Instruction - Integration

## Import “fmul” into Nios CPU



# Which One Do I Use?

## ■ Custom Instruction

- Used for Low-Clock Cycle Calculations
- Provides Quick Access to Inputs/Output
- Accessed Only by CPU
- Stalls CPU

## ■ Custom Peripheral

- Used for Labor-Intensive Operations
- Accessible through the Avalon Bus
- Accessible by Other Masters (i.e., DMA)
- CPU Independent





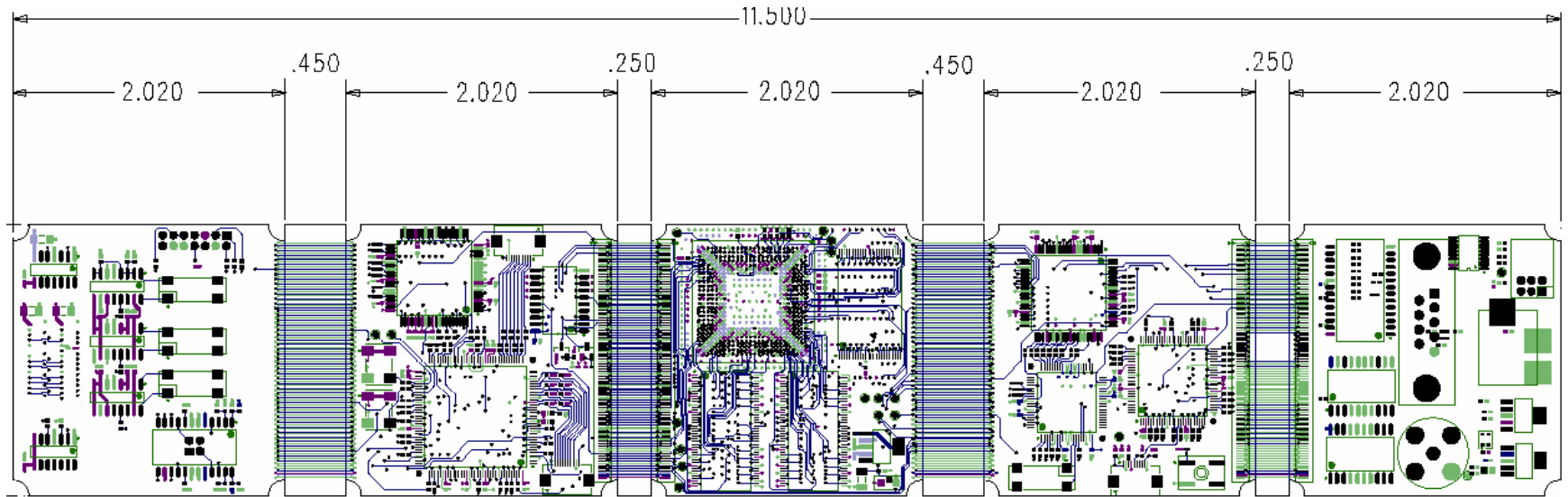
**SOPC**  
**WORLD**  
2003

# Applications & Examples

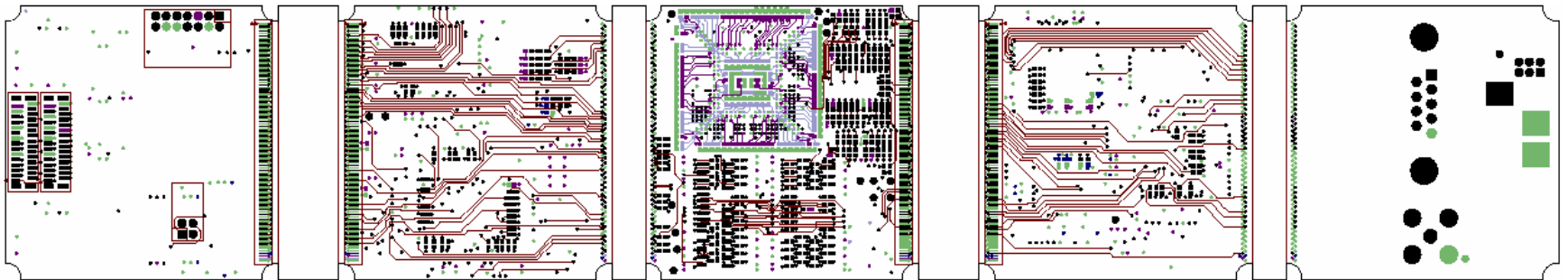
*Real Customer Designs*

# Example 1: SOPC Reality

Top

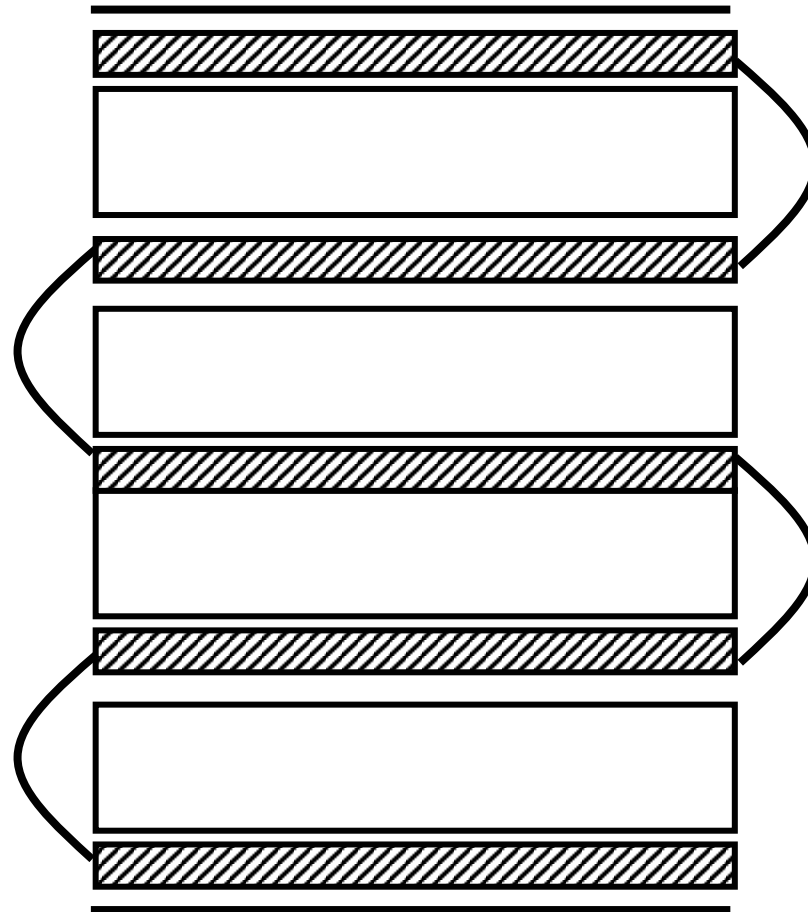


Bottom

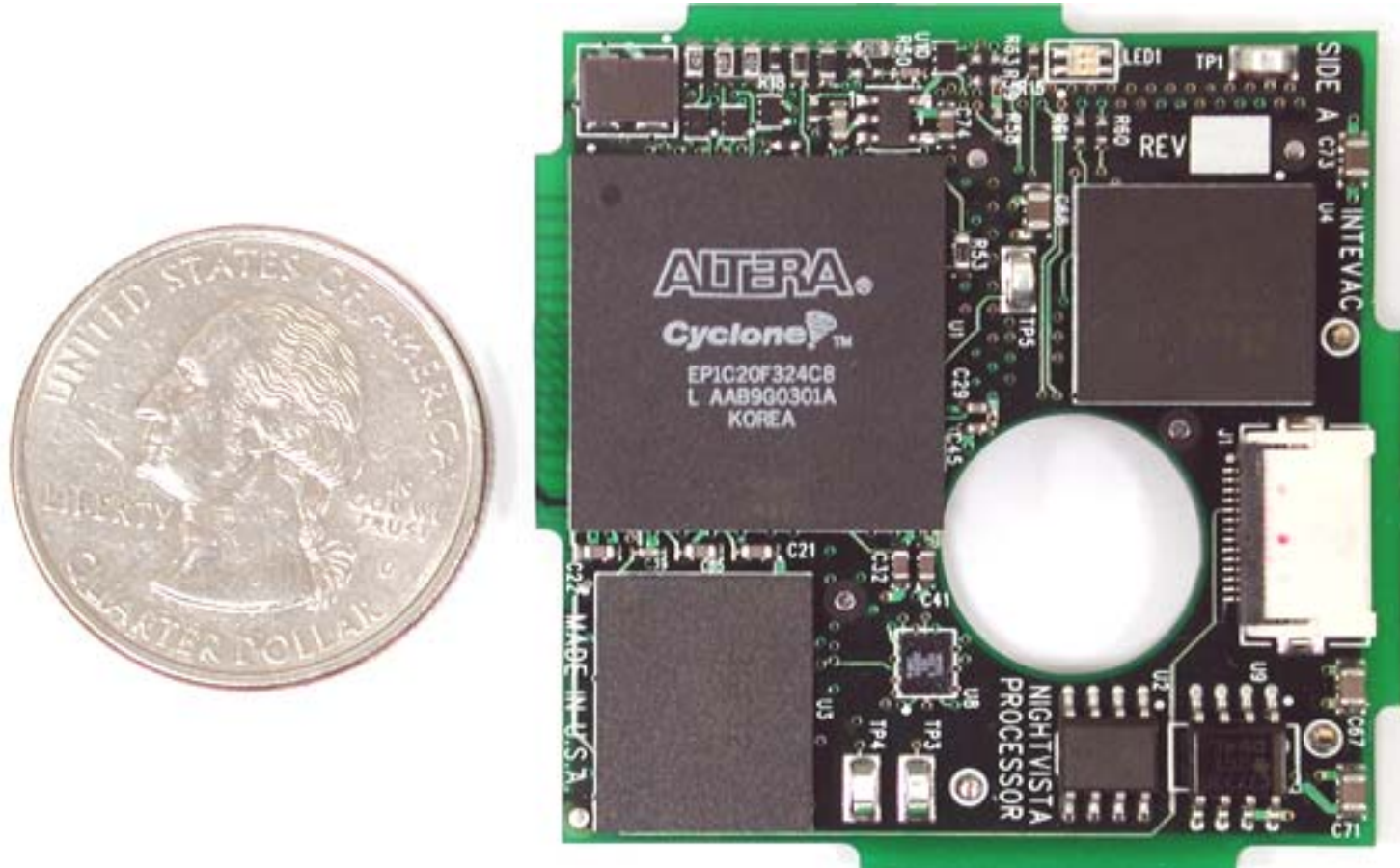


# Example 1: SOPC Reality

## *The Accordion Stackup*



# Example 1: SOPC Reality

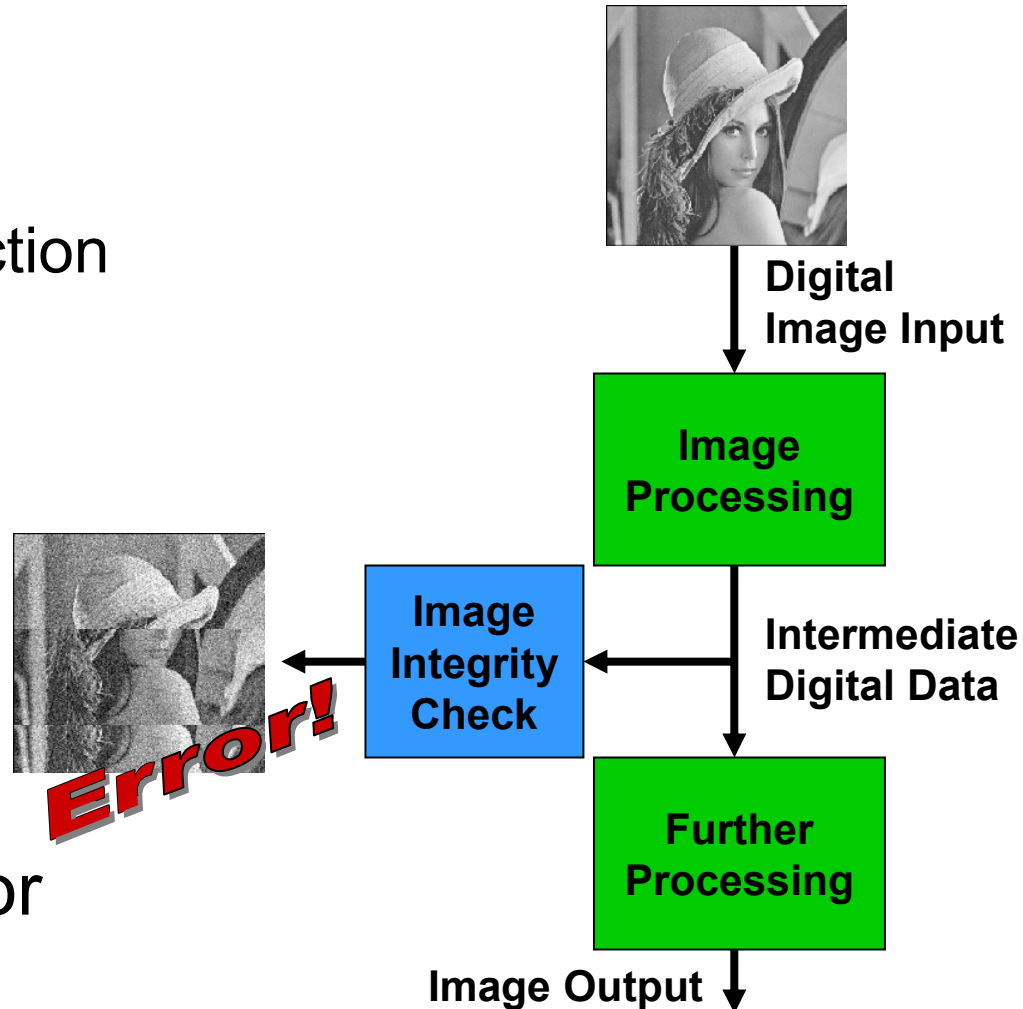


# Example 1: SOPC Reality

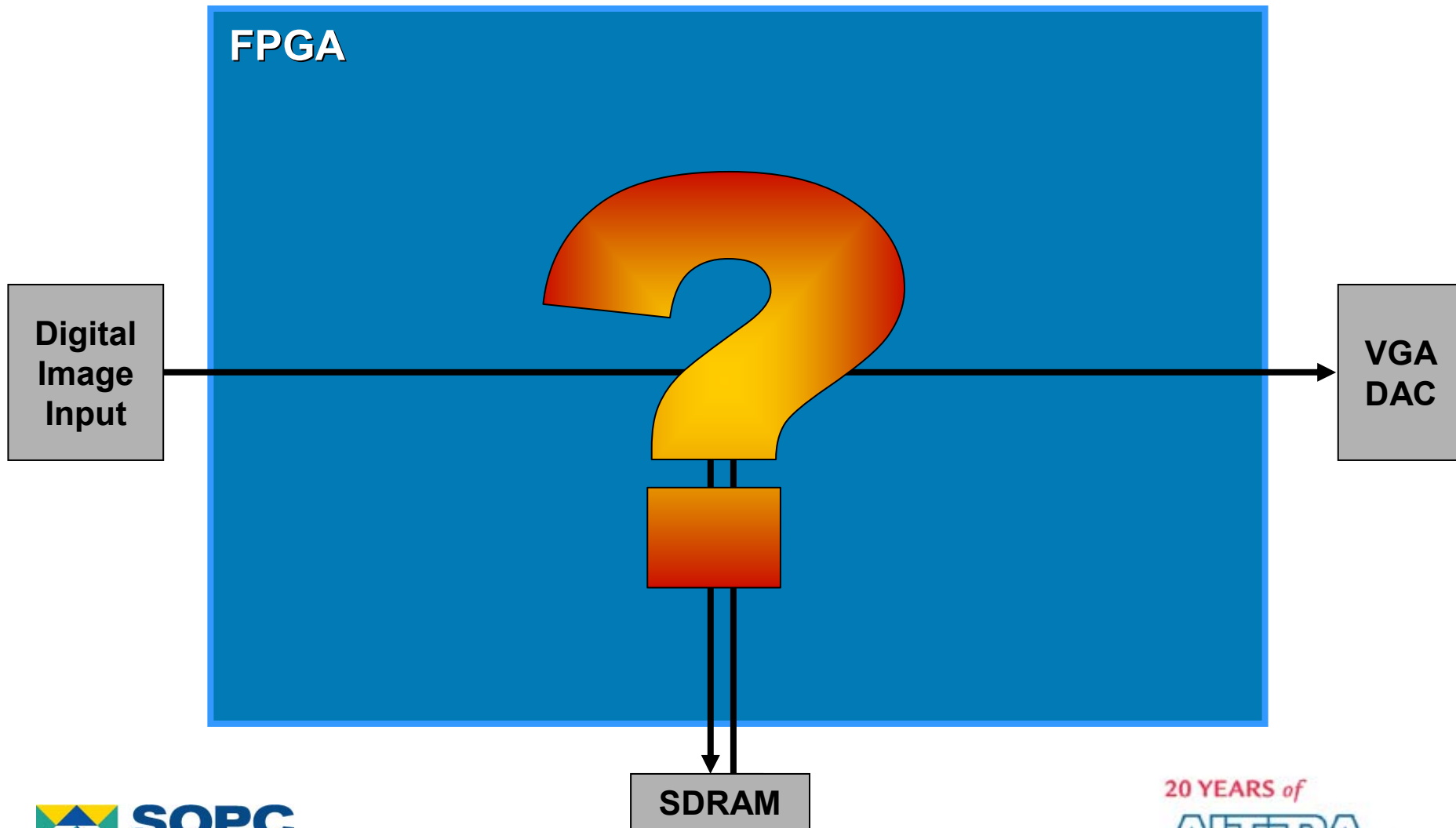


# Example 2: Digital Imaging Equipment

- Application
  - Quality & Assurance During System Production
- Function
  - Intercept Digital-Image Data Stream
  - Display Image on VGA Monitor
  - Verify Image Integrity
- No Need for Processor

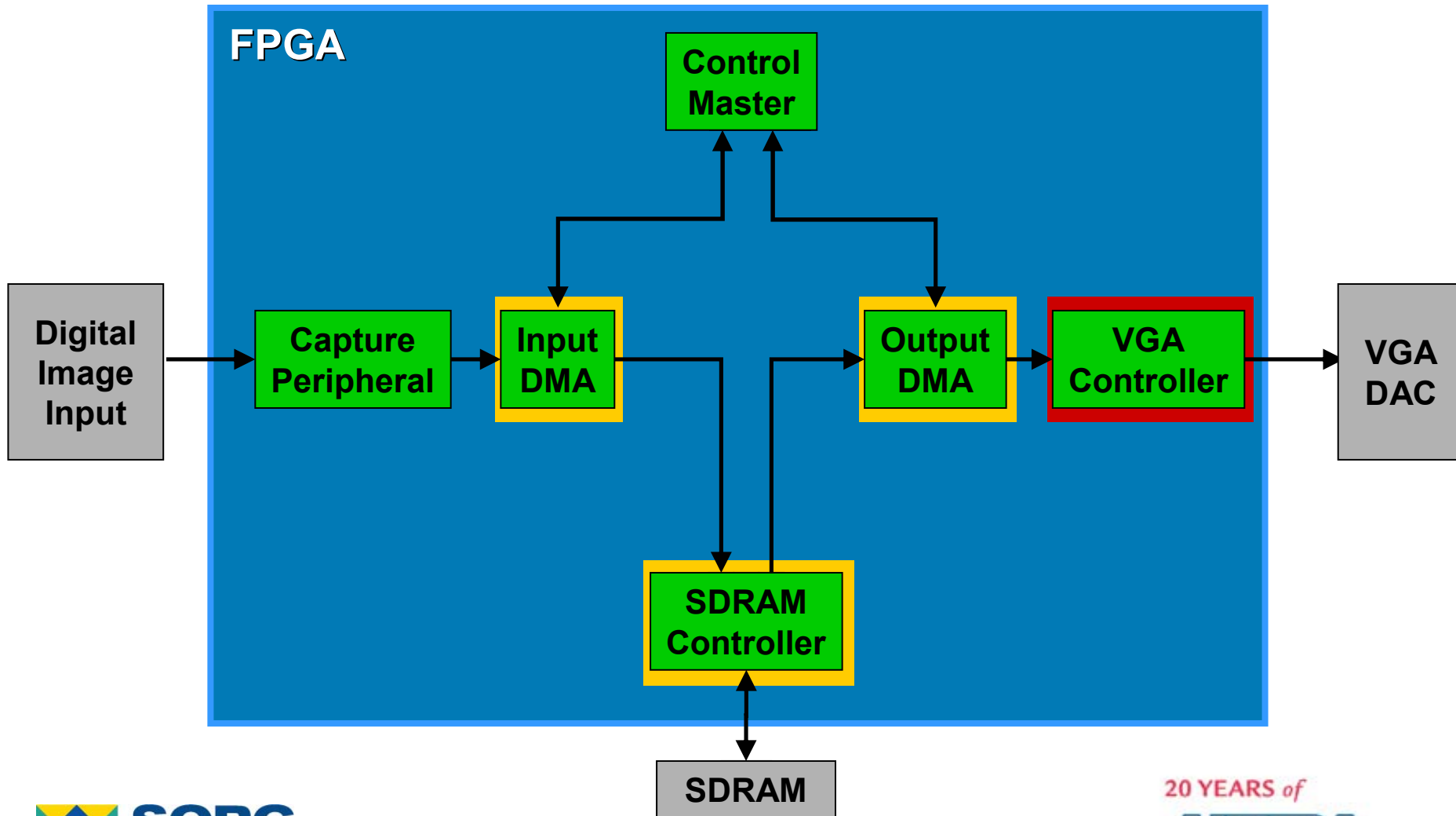


# Example 2: Digital Imaging Equipment



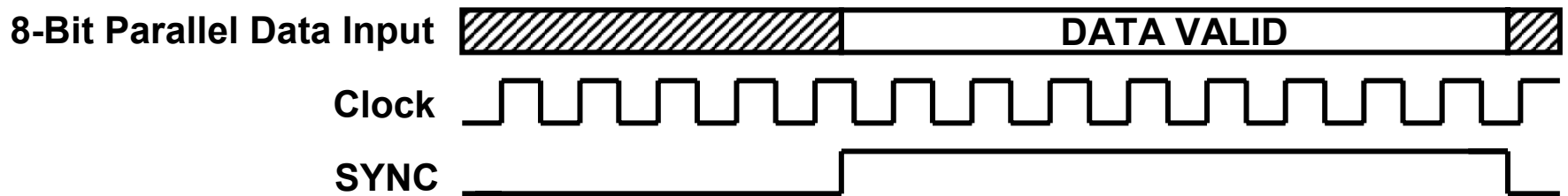
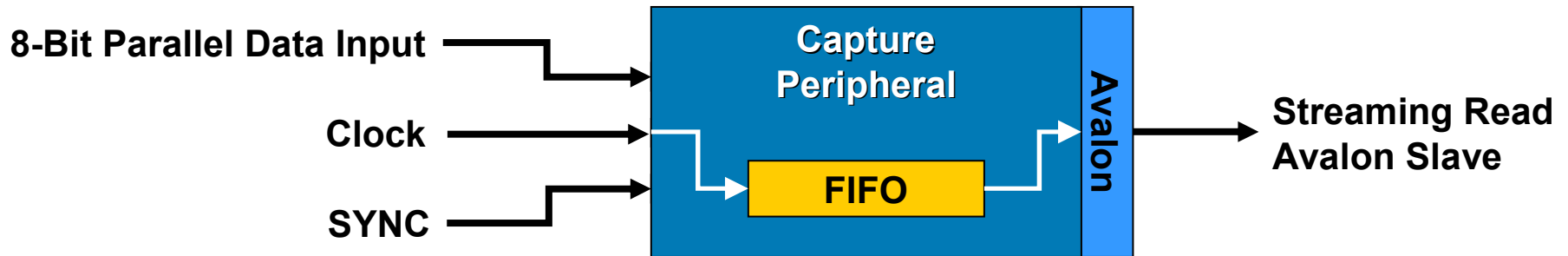


# Example 2: Digital Imaging Equipment

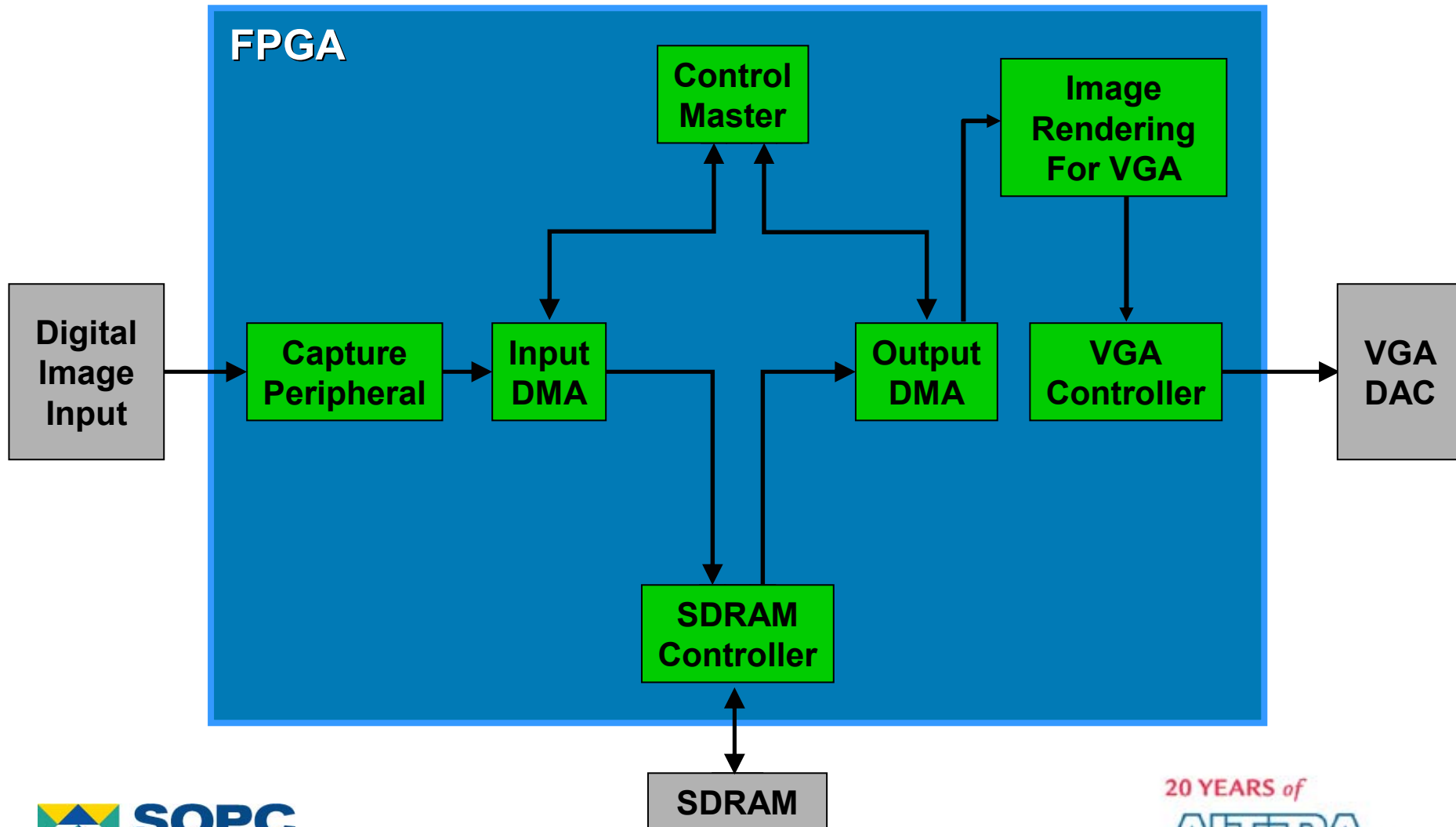




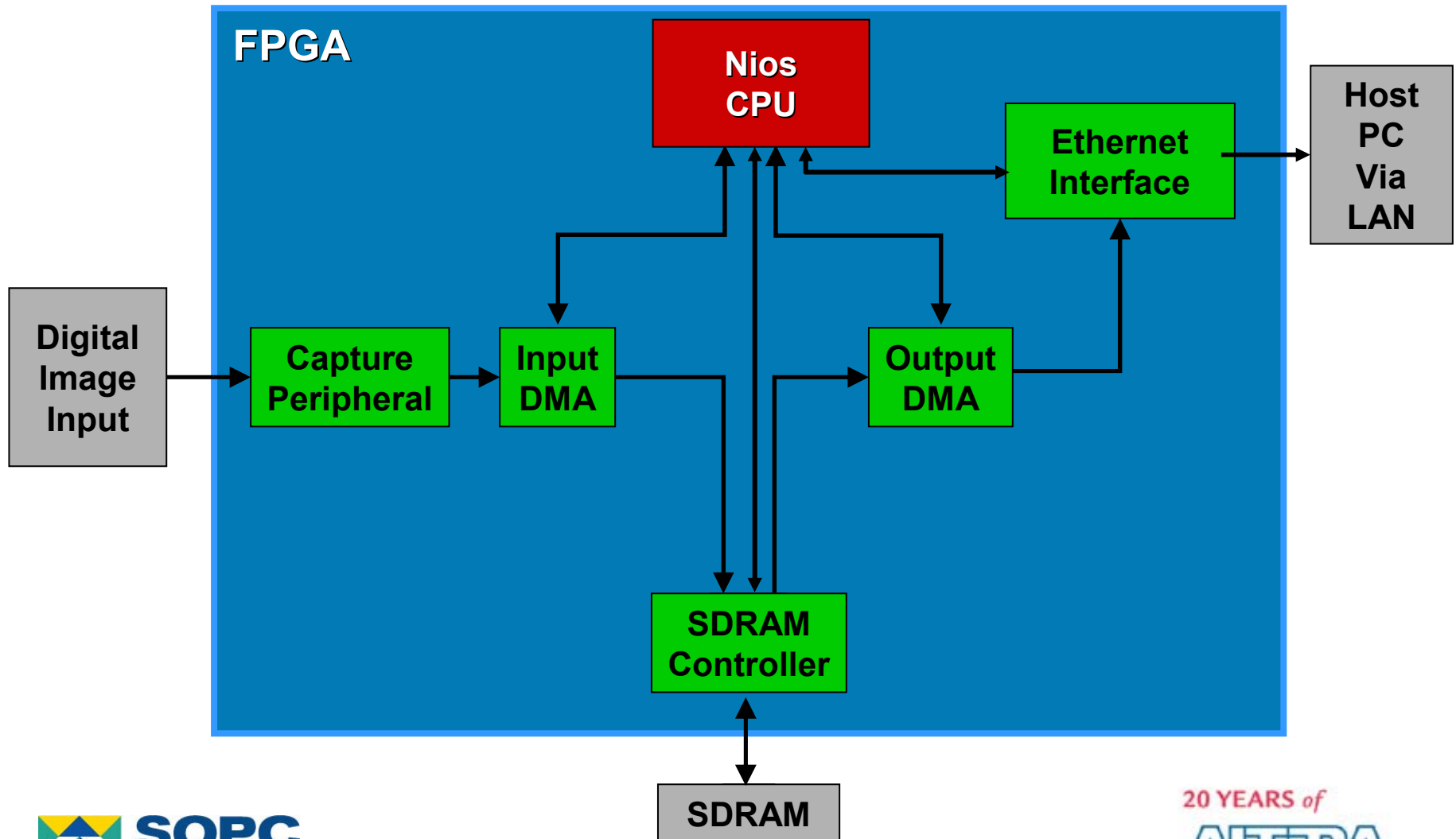
# Example 2: Digital Imaging Equipment



# Example 2: Digital Imaging Equipment



# Example 2: Digital Imaging Equipment



# Example 3: Mandelbrot Algorithm

```
int float_mandelbrot(float cr, float ci, int max_iter)
{
    float xsqr=0.0, ysqr=0.0, x=0.0, y=0.0;
    int iter=0;

    while( ((xsqr + ysqr) < 4.0) && (iter < max_iter) )
    {
        xsqr = x * x;
        ysqr = y * y;
        y = (2 * x * y) + ci;
        x = xsqr - ysqr + cr;
        iter++;
    }

    return(iter);
}
```

# Example 3: Optimizations

- Floating-Point Software in FPU Co-Processor
- Floating-Point Software in Integer Software
- Integer Software Done in Hardware
- Add DMA Transfer to Hardware Acceleration
- Parallelize Subsections of Display
- Simplify Control Master



**SOPC**  
**WORLD**  
2003

## Example 3: Mandelbrot

*Demonstration*

# Conclusion

- Altera Delivers System-Level Integration Solutions
  - SOPC Builder
  - Avalon
  - Nios
- SOPC Builder Accelerates Embedded System Design
  - Design Customization & IP Re-Use
  - Hardware Acceleration
  - Rapid Software Development



20 YEARS of

**ALTERA**®

INNOVATION



# **SOPC**

# **WORLD**

## 2003