Introduction to Zynq
Required Reading

The ZYNQ Book

• Chapter 1: Introduction
• Chapter 2: The Zynq Device (“What is it?”)
• Chapter 5: Applications and Opportunities (“What can I do with it?”)

Xilinx Educational Video

• Why Zynq?
  http://www.xilinx.com/training/zynq/why-zynq.htm
What is Software/Hardware Codesign?

*Integrated design of systems that consist of hardware and software components*

- Analysis of HW/SW boundaries and interfaces
- Evaluation of design alternatives
Embedded Systems vs. General-Purpose Computing

- **Embedded Systems**
  - Few applications that are known at design-time.
  - Not programmable by end user.
  - Fixed run-time requirements (additional computing power not useful).
  - Criteria:
    - cost
    - power consumption
    - predictability
    - meeting time bounds

- **General Purpose Computing**
  - Broad class of applications.
  - Programmable by end user.
  - Faster is better.
  - Criteria:
    - cost
    - average speed

Source: ETHZ, Prof. Lothar Thiele
Idea of an Embedded System

Source: ETHZ, Prof. Lothar Thiele
Software vs. Hardware Trade-offs

- Improve Performance
- Improve Energy Efficiency
- Reduce Power Density

Implement more in Hardware

- Manage Design Complexity
- Reduce Design Cost
- Stick to Design Schedule
- Handle Deep Submicron

Implement more in Software

Source: A Practical Introduction to Hardware/Software Codesign
Energy Efficiency of AES Implementations on Various Platforms

Source: A Practical Introduction to Hardware/Software Codesign
## Distinct Features of Hardware and Software Design

<table>
<thead>
<tr>
<th></th>
<th>Hardware</th>
<th>Software</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design Paradigm</td>
<td>Decomposition in space</td>
<td>Decomposition in time</td>
</tr>
<tr>
<td>Resource</td>
<td>Area (#gates, #Slices)</td>
<td>Time (#Cycles)</td>
</tr>
<tr>
<td>Flexibility</td>
<td>Must be designed in</td>
<td>Implicit</td>
</tr>
<tr>
<td>Parallelism</td>
<td>Implicit</td>
<td>Must be designed in</td>
</tr>
<tr>
<td>Modeling</td>
<td>Model ≠ Implementation</td>
<td>Model ≈ Implementation</td>
</tr>
<tr>
<td>Reuse</td>
<td>Uncommon</td>
<td>Common</td>
</tr>
</tbody>
</table>
Why Codesign?

**classic design**

```
        hw
       /  
      /    
     sw    
```

**co-design**

```
        hw
       /  
      /    
     sw   hw
```

```
        hw
       /  
      /    
     sw   sw
```
System Design Flow

Software

- SW-Compilation
- Instruction Set
- SW-Compilation
- Machine Code

Hardware

- HW-Synthesis
- Instruction Set
- Intellectual Prop. Block
- Net lists

Source: ETHZ, Prof. Lothar Thiele
Implementation Alternatives

- General-purpose processors
- Application-specific instruction set processors (ASIPs)
  - Microcontroller
  - DSPs (digital signal processors)
- Programmable hardware
  - FPGA (field-programmable gate arrays)
- Application-specific integrated circuits (ASICs)

Source: ETHZ, Prof. Lothar Thiele
Traditional Discrete Component Architecture

Source: The Zynq Book
System-on-Chip (SoC)

Source: The Zynq Book
FPGA with Soft Processor Core

Source: The Zynq Book
A Simplified Model of the Zynq Architecture

Source: The Zynq Book
Simplified Hardware Architecture of an Embedded SoC

Source: The Zynq Book
Mapping of an Embedded SoC Hardware Architecture to Zynq

Source: The Zynq Book
Mapping of an Embedded SoC Hardware Architecture to Zynq

Source: Xilinx White Paper: Extensible Processing Platform
## Comparison with Alternative Solutions

<table>
<thead>
<tr>
<th></th>
<th>ASIC</th>
<th>ASSP</th>
<th>2 Chip Solution</th>
<th>Zynq</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Performance</strong></td>
<td>+</td>
<td>+</td>
<td>■</td>
<td>+</td>
</tr>
<tr>
<td><strong>Power</strong></td>
<td>+</td>
<td>+</td>
<td>—</td>
<td>+</td>
</tr>
<tr>
<td><strong>Unit Cost</strong></td>
<td>+</td>
<td>+</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td><strong>Total Cost of Ownership</strong></td>
<td>■</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td><strong>Risk</strong></td>
<td>—</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td><strong>Time to Market</strong></td>
<td>—</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td><strong>Flexibility</strong></td>
<td>—</td>
<td>—</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td><strong>Scalability</strong></td>
<td>—</td>
<td>■</td>
<td>+</td>
<td>+</td>
</tr>
</tbody>
</table>

+ positive, — negative, ■ neutral

Source: Xilinx Video Tutorials
Zynq Highlights

➢ Complete ARM®-based Processing System
  – Dual ARM Cortex™-A9 MPCore™, processor centric
  – Integrated memory controllers & peripherals
  – Fully autonomous to the Programmable Logic

➢ Tightly Integrated Programmable Logic
  – Used to extend Processing System
  – High performance ARM AXI interfaces
  – Scalable density and performance

➢ Flexible Array of I/O
  – Wide range of external multi-standard I/O
  – High performance integrated serial transceivers
  – Analog-to-Digital Converter inputs

Source: Xilinx Video Tutorials
ARM Processor Roadmap

Source: Xilinx White Paper: Extensible Processing Platform
Basic Design Flow for Zynq SoC

Source: The Zynq Book
Design Flow for Zynq SoC

Source: Xilinx White Paper: Extensible Processing Platform
Zynq SoC Ecosystem

- Ecosystem
- Silicon Devices
- Peripherals and Accelerators
- Development Platforms
- Operating Systems
- Development Tools
- Zynq 101 (Training)
Zynq SoC Ecosystem

Source: The Zynq Book
Alternative Solutions

**Xilinx Zynq**
Zynq-7000 All Programmable SoCs with Cortex-A9 MPCore

**Altera Arria V & Cyclone V**
Hard processor system (HPS) with Cortex-A9 MPCore

**Microsemi Smartfusion2**
Cortex M3
The Zynq Processing System

Source: The Zynq Book
Simplified Block Diagram of the Application Processing Unit (APU)

- Memory Management Unit
- NEON & floating point extensions
- ARM Cortex-A9 processing core
- Level 1 cache memory (32KB each, data & instructions)
- Shared Level 2 cache memory (512KB, data & instructions)
- On Chip Memory (256KB)

Source: The Zynq Book
SIMD (Single Instruction Multiple Data) Processing in the NEON Media Processing Engine (MPE)

Source: The Zynq Book
Programmable Logic (PL)
CLBs and IOBs

Source: The Zynq Book
Programmable Logic (PL) BRAMs and DSP units

Source: The Zynq Book
AXI Interconnects and Interfaces

Source: The Zynq Book
Using Extended Multiplexed Input/Output (EMIO) to Interface Between PS and PL

Source: The Zynq Book
Automotive Applications

- Functions by embedded processing:
  - ABS: Anti-lock braking systems
  - ESP: Electronic stability control
  - Airbags
  - Efficient automatic gearboxes
  - Theft prevention with smart keys
  - Blind-angle alert systems
  - ... etc ...

- Multiple networks
  - Body, engine, telematics, media, safety

- Multiple processors
  - Up to 100
    - 8-bit – door locks, lights, etc.
    - 16-bit – most functions
    - 32-bit – engine control, airbags
  - Processing where the action is
  - Sensors and actuators distributed all over the vehicle
  - Networked together
Automotive Applications
Lane and Road Sign Recognition

Source: The Zynq Book
Computer Vision
Detection of Cars at a Junction

Source: The Zynq Book
Smart Home

- Intelligent lighting
- Energy efficient heating/cooling in each room
- Temperature & ambient light sensors in each room
- Smart electricity meter
- Security system
- Wireless internet to all rooms
- Networked entertainment
- Smart gas meter
- Soil sensors and smart watering

Source: The Zynq Book
Software Defined Radio (SDR)

- a radio which can be reconfigured while in operation
- all of the physical layer functions are software defined
- used initially in military applications (JTRS 1998), recently entering the commercial arena
- can support multiple radio standards (for cellular networks [2G, 3G, 4G], WiFi, Bluetooth, GPS reception, etc.)
- May be used in smartphones, tablets, e-readers, TVs, cars, transportation, emergency services, etc.)
Software Defined Radio (SDR)

Source: The Zynq Book
Software Defined Radio (SDR)

- The Physical Layer (PHY) – the part of radio directly adjacent to the Radio Frequency (RF) circuitry and air interface
- Computationally intensive, implementing high-speed filters, modulation, coding, DSP algorithms, support for ADC and DAC
- Most complex computations implemented in hardware (with parameters set from software)
- Less complex computations can be performed in either hardware or software
Cognitive Radio

- an intelligent radio that can be programmed and configured dynamically
- its transceiver is designed to use the best (under-used) wireless channels in its vicinity
- automatically detects available channels in wireless spectrum, and changes its transmission or reception parameters accordingly
- allows more concurrent wireless communications in a given spectrum band at one location
- a form of dynamic spectrum management
Communication Systems

Wireless Basestation

Satellite Groundstation

Wired Network Switches

Source: The Zynq Book
Control and Instrumentation Systems

Industrial Control Room  Wind Turbines  High Energy Physics Experiment

Source: The Zynq Book
Medical Applications

MRI Scanning  Robot Assisted Surgery

Source: The Zynq Book
# Choice Among Various Implementation Platforms

<table>
<thead>
<tr>
<th>Platform</th>
<th>Total System Cost</th>
<th>Flexibility</th>
<th>Differentiation</th>
<th>Time-to-Market</th>
<th>Cost of Derivatives</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zynq SoC</td>
<td>Low + best value</td>
<td>Most flexible: HW and SW programmable + programmable I/O</td>
<td>Highest degree of programmability, HW/SW co-design</td>
<td>Fastest for integrated HW &amp; SW differentiation</td>
<td>Lowest due to HW &amp; SW programmability</td>
<td>Predictably low risk</td>
</tr>
<tr>
<td>ASSP + FPGA</td>
<td>Higher than Zynq SoC (system dependent)</td>
<td>Highly flexible but ASSP I/O limited compared to Zynq SoC</td>
<td>HW and SW programmable, ASSP-dependent</td>
<td>Fastest if ASSP requires HW differentiation</td>
<td>Low to high depending on FPGA vendor</td>
<td>Low to high depending on FPGA vendor</td>
</tr>
<tr>
<td>ASSP</td>
<td>Lowest if SW-only programmability is sufficient</td>
<td>Good but SW-programmable only</td>
<td>Limited to SW programmable only - easy cloning</td>
<td>Fastest if SW-only differentiation required</td>
<td>Lowest if SW-only derivatives needed</td>
<td>Can be Lowest if SW-only programmability is sufficient</td>
</tr>
<tr>
<td>ASIC</td>
<td>High to prohibitive</td>
<td>Once manufactured only limited SW flexibility</td>
<td>Best HW differentiation but limited SW differentiation</td>
<td>Lowest &amp; riskiest</td>
<td>Highest</td>
<td>Terrible (respins)</td>
</tr>
</tbody>
</table>

Source: Xcell Journal, no. 88, Q3 2014
# Advantages of Zynq

<table>
<thead>
<tr>
<th>Lowest NRE, Best Risk Mitigation</th>
<th>Greatest Flexibility &amp; Differentiation</th>
<th>Streamlined Productivity &amp; Fast TTM</th>
<th>Lowest Cost of Derivatives &amp; Highest Profitability</th>
</tr>
</thead>
<tbody>
<tr>
<td>✓ Already manufactured silicon</td>
<td>✓ All Programmable HW, SW &amp; I/O</td>
<td>✓ Instant HW/SW co-development</td>
<td>✓ IP standardized on ARM AMBA AXI4</td>
</tr>
<tr>
<td>✓ Negligible development &amp; design tool costs</td>
<td>✓ Anytime field programmable</td>
<td>✓ All Programmable Abstractions (C, C++, OpenCV, OpenCL, HDL, model-based entry)</td>
<td>✓ Reuse precertified code (ISO, FCC, etc.)</td>
</tr>
<tr>
<td>✓ Extensive development boards</td>
<td>✓ System Secure (encryption)</td>
<td>✓ Broad and open OS &amp; IDE support (Open-source Linux &amp; Android, FreeRTOS, Windows Embedded, Wind River, Green Hills, &amp; many others)</td>
<td>✓ Volume silicon, power circuitry, PCBs &amp; IP licensing</td>
</tr>
</tbody>
</table>

Source: Xcell Journal, no. 88, Q3 2014
# Comparison of the Development Time & Cost

<table>
<thead>
<tr>
<th></th>
<th>28nm ASIC (IBS Data)</th>
<th></th>
<th>Zynq SoC (Xilinx Estimates)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>%</td>
<td>Approximate Engineering Months</td>
<td>Total Cost ($M)</td>
<td>%</td>
</tr>
<tr>
<td><strong>Hardware</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IP qualification</td>
<td>26</td>
<td>704</td>
<td>11.8</td>
<td>20</td>
</tr>
<tr>
<td>Architecture</td>
<td>8</td>
<td>209</td>
<td>4.2</td>
<td>45</td>
</tr>
<tr>
<td>Verification</td>
<td>53</td>
<td>1431</td>
<td>28.9</td>
<td>35</td>
</tr>
<tr>
<td>Physical design</td>
<td>13</td>
<td>350</td>
<td>6.9</td>
<td>0</td>
</tr>
<tr>
<td>Subtotal hardware</td>
<td>100</td>
<td>2694</td>
<td>51.8</td>
<td>100</td>
</tr>
<tr>
<td>(design engineering</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>resources)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Software</strong></td>
<td></td>
<td>4296</td>
<td>59.8</td>
<td>720</td>
</tr>
<tr>
<td>Prototype cost ($M)</td>
<td>2.1</td>
<td></td>
<td></td>
<td>1.0</td>
</tr>
<tr>
<td>Validation of prototypes</td>
<td></td>
<td>815</td>
<td>16.6</td>
<td>140</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
<td>7805</td>
<td>130.3</td>
<td>1360</td>
</tr>
</tbody>
</table>

Source: Xcell Journal, no. 88, Q3 2014
Academic Subjects to which Zynq is Relevant

Source: The Zynq Book
The ZYBO Development Board

Source: The Zynq Book
## ZYBO Board Components

<table>
<thead>
<tr>
<th>Callout</th>
<th>Component Description</th>
<th>Callout</th>
<th>Component Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Power Switch</td>
<td>15</td>
<td>Processor Reset Pushbutton</td>
</tr>
<tr>
<td>2</td>
<td>Power Select Jumper and battery header</td>
<td>16</td>
<td>Logic configuration reset Pushbutton</td>
</tr>
<tr>
<td>3</td>
<td>Shared UART/JTAG USB port</td>
<td>17</td>
<td>Audio Codec Connectors</td>
</tr>
<tr>
<td>4</td>
<td>MIO LED</td>
<td>18</td>
<td>Logic Configuration Done LED</td>
</tr>
<tr>
<td>5</td>
<td>MIO Pushbuttons (2)</td>
<td>19</td>
<td>Board Power Good LED</td>
</tr>
<tr>
<td>6</td>
<td>MIO Pmod</td>
<td>20</td>
<td>JTAG Port for optional external cable</td>
</tr>
<tr>
<td>7</td>
<td>USB OTG Connectors</td>
<td>21</td>
<td>Programming Mode Jumper</td>
</tr>
<tr>
<td>8</td>
<td>Logic LEDs (4)</td>
<td>22</td>
<td>Independent JTAG Mode Enable Jumper</td>
</tr>
<tr>
<td>9</td>
<td>Logic Slide switches (4)</td>
<td>23</td>
<td>PLL Bypass Jumper</td>
</tr>
<tr>
<td>10</td>
<td>USB OTG Host/Device Select Jumpers</td>
<td>24</td>
<td>VGA connector</td>
</tr>
<tr>
<td>11</td>
<td>Standard Pmod</td>
<td>25</td>
<td>microSD connector (Reverse side)</td>
</tr>
<tr>
<td>12</td>
<td>High-speed Pmods (3)</td>
<td>26</td>
<td>HDMI Sink/Source Connector</td>
</tr>
<tr>
<td>13</td>
<td>Logic Pushbuttons (4)</td>
<td>27</td>
<td>Ethernet RJ45 Connector</td>
</tr>
<tr>
<td>14</td>
<td>XADC Pmod</td>
<td>28</td>
<td>Power Jack</td>
</tr>
</tbody>
</table>

Source: ZYBO Reference Manual
ZYBO General Purpose Input Output (GPIO)

Source: ZYBO Reference Manual
VGA Circuit

Source: ZYBO Reference Manual
VGA Connector

Pin 1: Red  Pin 5: GND
Pin 2: Grn  Pin 6: Red GND
Pin 3: Blue  Pin 7: Grn GND
Pin 13: HS  Pin 8: Blu GND
Pin 14: VS  Pin 10: Sync GND

Source: ZYBO Reference Manual
USB-UART Bridge

Source: ZYBO Reference Manual
Pmod Connector

Source: ZYBO Reference Manual