FPGA Scoping and IV&V

Presented by Pradip Maitra and John C. Ryan
Sep 11, 2014

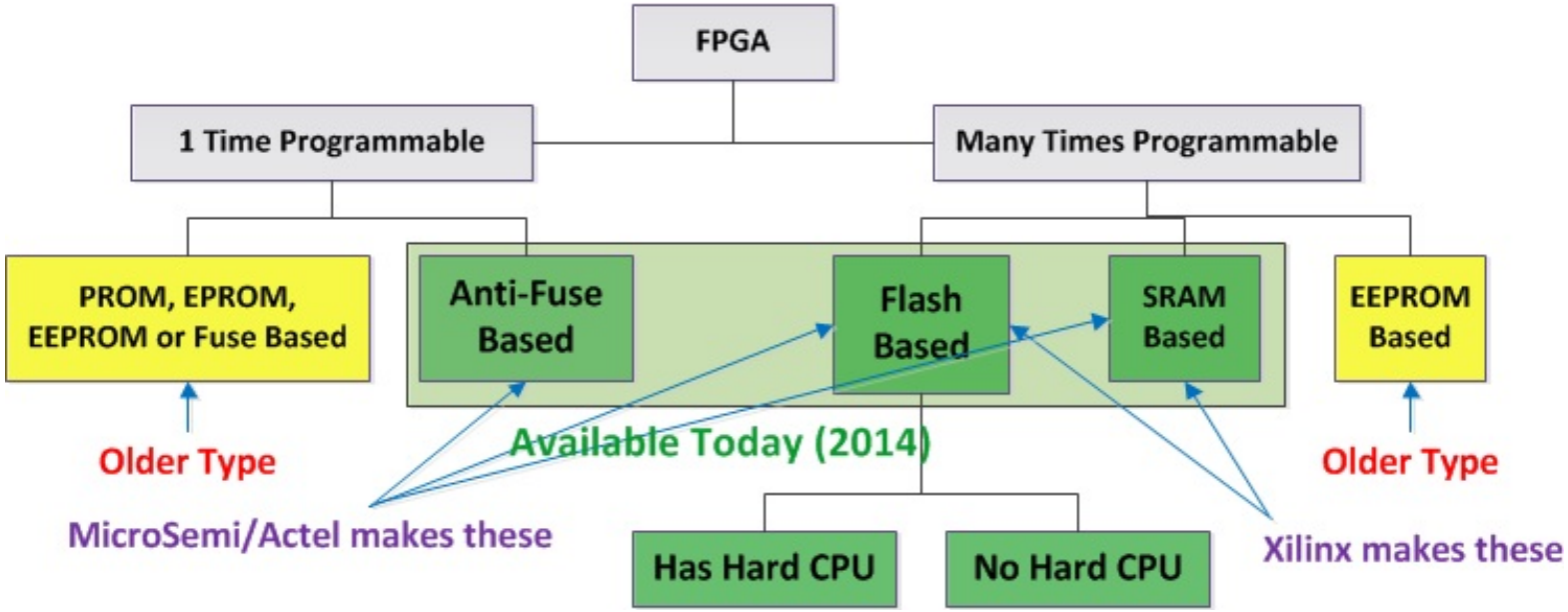
Agenda

FPGA Scoping And IV&V	
FPGA Basics	03
FPGA Hard CPU Example	05
FPGA Without CPU Example	06
Why Bother About FPGA?	07
The Hardware/Software Boundary	09
Scoping FPGA for IV&V work	10
Performing IV&V on FPGAs in Scope	12
Summary	13

FPGA Basics

- FPGA is **F**ield **P**rogrammable **G**ate **A**rray. You might think of it like a box full of basic components (transistors, resistors, etc.) and you have means to interconnect them in any way you want at your own lab.
- Digital circuits are made of transistor **gates**. You connect them in a certain way to get certain functionality. The FPGA has revolutionized the process of logic implementation by allowing you to manipulate the connections between these gates… sometimes **one time only** and sometimes **as many times as you wan**t under the control of an external CPU-memory subsystem.
- It has infiltrated in our lives: portable gadgets, security systems, transportation systems, aviation, space, what not!

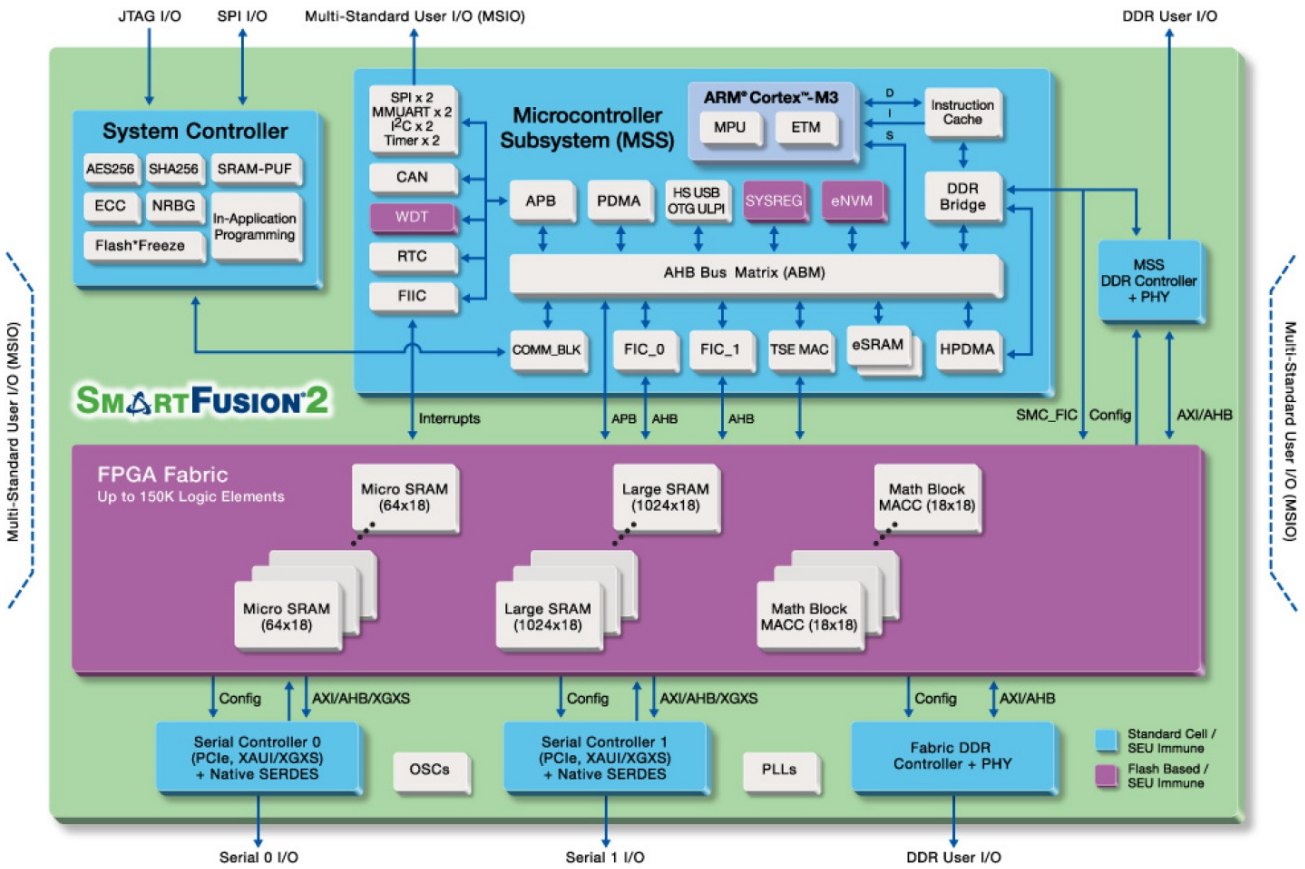
FPGA Basics



- Presence of CPU => **Stored Program Execution; Possible capability to change the stored program on the fly and restart**.
- "Hard" CPU: Has built-in standard CPU core (E.g., ARM Cortex). A System On Chip (SOC) effectively.
- "Soft" CPU: You build the CPU using available gates of the FPGA.

FPGA Hard CPU Example

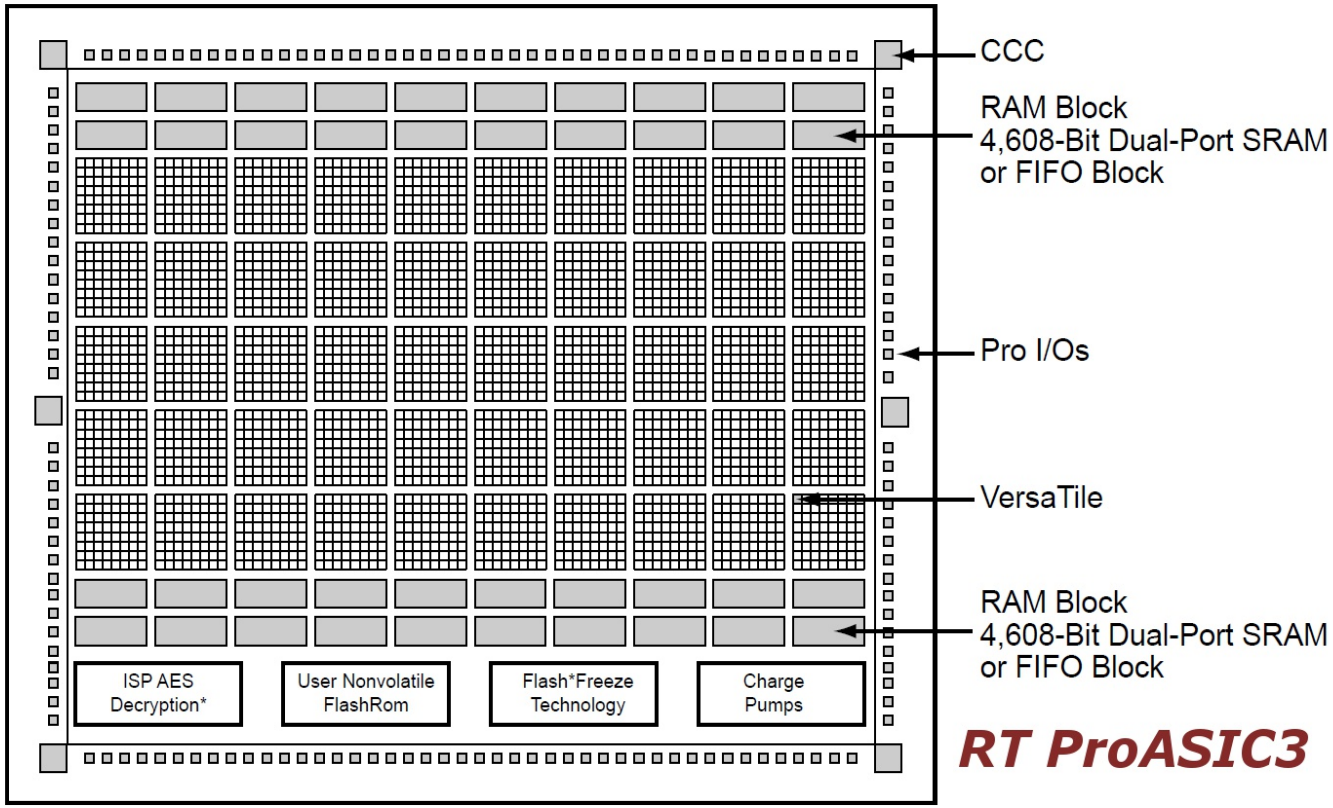
- Hard CPU based FPGAs: SmartFusion, Zynq



Ref Source: http://www.microsemi.com/products/fpga-soc/soc-fpga/smartfusion2

FPGA Without CPU Example

- No CPU, Flash based FPGAs: ProASIC3, Igloo



Ref Source: http://www.microsemi.com/document-portal/doc_download/130714-radiation-tolerant-proasic3-low-power-space-flight-fpgas-datasheet

Why Bother About FPGA?

- FPGA Advantages
  - Typically faster than a CPU in signal processing: Can execute massively parallel logic using hardware gates; few transformations between input and output. A CPU on the other hand needs instructions, fetching them, executing them, storing the result; parallelism comes mostly from using multiple cores (8 max for now).
  - Offloads routine work from CPU: E.g., Network stack, Protocol Conversion in Gigabit network; standard protocol implementation like Ethernet, 1553, SpaceWire.
  - Consumes less power than equivalent CPU based implementation.
  - Can attain very high speed (missile applications; DSP based snooping).
- These make FPGA very attractive in spacecraft control.
- IV&V is seeing an increasing number of FPGA based systems. The famous Dream Mode FPGA of MSL is a glaring example. **Source: http://www.eetimes.com/document.asp?doc_id=1262350**

Why Bother About FPGA?

- NASA Software Engineering Requirements NPR 7150.2A from **http://nodis3.gsfc.nasa.gov/** effectively mandates definition of **software** in Page 3, Section P.1:
  - Software includes programs and data. This definition includes commercial-off-the-shelf (COTS) software, government-off-the-shelf (GOTS) software, modified-off-the-shelf (MOTS) software, reused software, auto generated code, embedded software, firmware, and open source software components.
- While it does not define what **firmware** is, the mere word "firmware" grabs the attention of those who think software as designing, coding, testing using IDE – all the while being blissfully unaware of the underlying system.
- Spacecraft Control executes in heavily embedded environment – it will not run as is on any Windows PC.

The Hardware/Software Boundary

Software	
- BIOS/bootstrap - Operating system - Applications	Programmed Easily changed Can “do anything” Cannot be 100%, exhaustively tested

Firmware
Software residing in non-volatile storage Programmable Logic Devices (e.g. FPGA) – Compiled/Programmed – May be Reprogrammable on the fly – Cannot be 100% exhaustively tested

Electronic Hardware	
ICs Microprocessors A/D, D/A converters Sensors	Off-the-shelf components Exhaustively Tested by Vendor

Scoping FPGA for IV&V Work

- Begin scoping if IV&V has received documentation that indicates FPGAs planned for development.
  - Usually by Mission PDR
  - May indicate in draft ICDs, Subsystem PDRs
  - Per Breadboard Designs, Engineering Models, FPGA manufacturer
  - Documentation indicating purpose of included firmware and interaction with components
- Identify FPGA and references (e.g. database, spreadsheet)
- Correlate FPGA to Capabilities (PBRA)
- Determine FPGA heritage and usage
- Perform RBA on potential FPGAs

Scoping FPGA for IV&V Work

- Scoping Guidance to distinguish between various FPGAs:
  - Capability: Does it control mission critical software components?

- If Yes, Let us try to distinguish between various FPGA types and their usage:
  - Heritage: Doing standard and well-tested IP like 1553, Ethernet?

    vs
  - One Time Only: First time implementation and also the last?

  - IP Changeable: Is it possible to change the FPGA IP on the fly?
    - Logic (hardware) design and implementation with no capability to modify FPGA usage
    - Command data/parameter updates to modify FPGA usage for logic design

    vs
    - Command implementation patches (reprogrammable) to modify FPGA usage represented as an application or algorithm (if software is used to make the decisions)
  - Update on the fly: The mission plans on changing FPGA IP on the fly?

Performing IV&V on FPGAs in Scope

IV&V Analysis Phase	IV&V Analysis/Limitations	Concerns
Requirements	Requirement Traceability/Validation - Limit to firmware requirements that can support reprogrammable design - Need to build up an FPGA competency and culture across IV&V	1. May not receive Artifacts or receive too late. 2. May not require Traceability 3. Analysts may lack expertise
Design/Design Interface	Validate design is consistent with requirements and available references (e.g. ICD, related PDR/CDR artifacts) - Limit to design (e.g. state machines) that represents the firmware and/or interfaces - Consider MATLAB/Simulink - Potentially treat FPGA as black box and validate in context of inputs/outputs - Need to build up an FPGA competency and culture across IV&V	1. May not receive Artifacts or receive too late, or lack details. 2. Analysts may lack expertise a. FPGA Knowledge b. MATLAB/Simulink
Implementation	Verify Firmware implements requirements and design - Consider spending time to learn VHDL language	1. May not receive Artifacts. 2. Analysts may lack expertise a. VHDL Language
Test	Ensure test cases encompass the proper scenarios and conditions (including modifiable firmware, parameters/environment data on the FPGA) and produce outputs that we can examine are being implemented correctly.	1. May not receive Artifacts. 2. Need to build up an FPGA competency and culture across IV&V

Summary

- FPGAs are becoming increasingly prominent in high level logic implementation.
- There have been instances in the space program where FPGA was used to implement a soft CPU and execute code.
- We presented a way to scope FPGA work.
- We showed a way to perform IV&V on FPGA.

Questions, Comments ?

TASC

INSIGHT APPLIED™