Lynx: A scalable solution utilising FPGAs for high performance data processing

Part B: Firmware & Software Development
December 2009

CSIR DPSS Presentation by Richard Focke

Experimental Radar Systems Team

Firmware / Software Developer

Introduction: Goals of my Presentation

- Background on firmware
- Demonstrate our design
- Showcase novel concepts
- Not only applicable to radar
- Possible issues

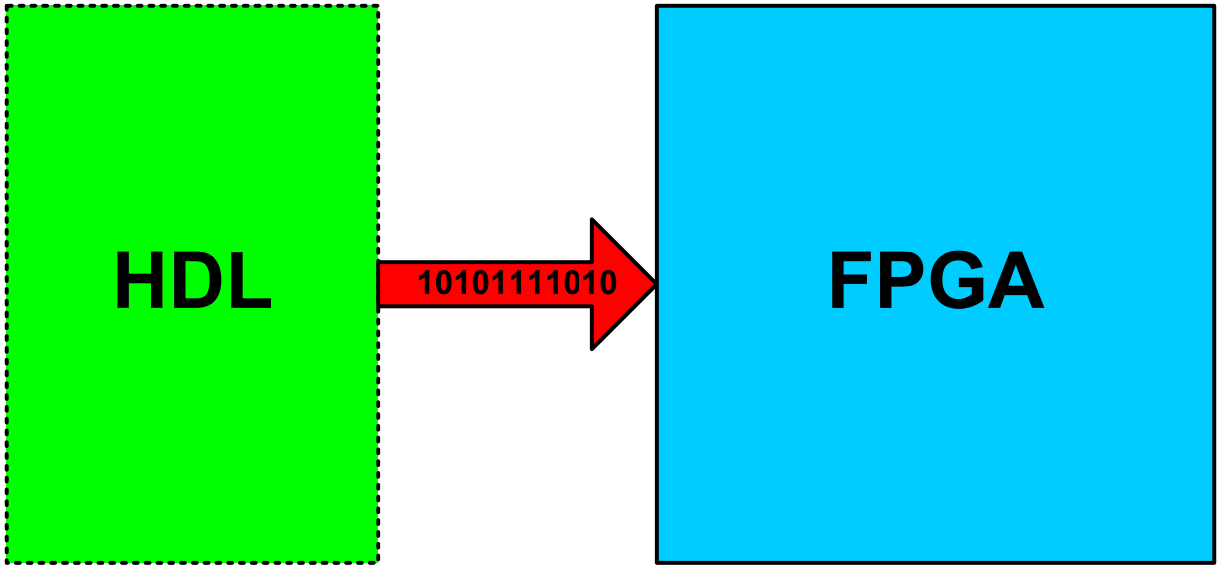
CSIR
our future through science

Part B: Firmware & Software Development Topics

1. **FPGA Firmware Background**
2. System Design Goals
3. Board Support Package
4. BSP Enabling Systems
5. Data Processing Firmware
6. Development Issues

CSIR
our future through science

1. What is FPGA Firmware?



our future through science

1. HDL Uses

- HDL for hardware **simulation** only
  - Model 3rd party hardware
  - Test bench for functional verification
  - Read and write files
  - Dynamic structures
- HDL for FPGA **firmware** (or ASIC)
  - Not all language features
  - Register transfer level (RTL)
  - Model basic FPGA blocks
  - Instantiate advanced FPGA blocks

CSIR
our future through science

Part B: Firmware & Software Development Topics

1. FPGA Firmware Background
2. **System Design Goals**

2. System Design Goals

- Adaptability & Scalability
- All FPGAs one processing space
- Migrate-able functions
- Decoupled hardware interfaces
- Functional independence
- Software: generic adaptable
- Consistent software interface
- Independent of processing flow

CSIR
our future through science

Part B: Firmware & Software Development Topics

1. FPGA Firmware Background
2. System Design Goals
3. **Board Support Package**

our future through science

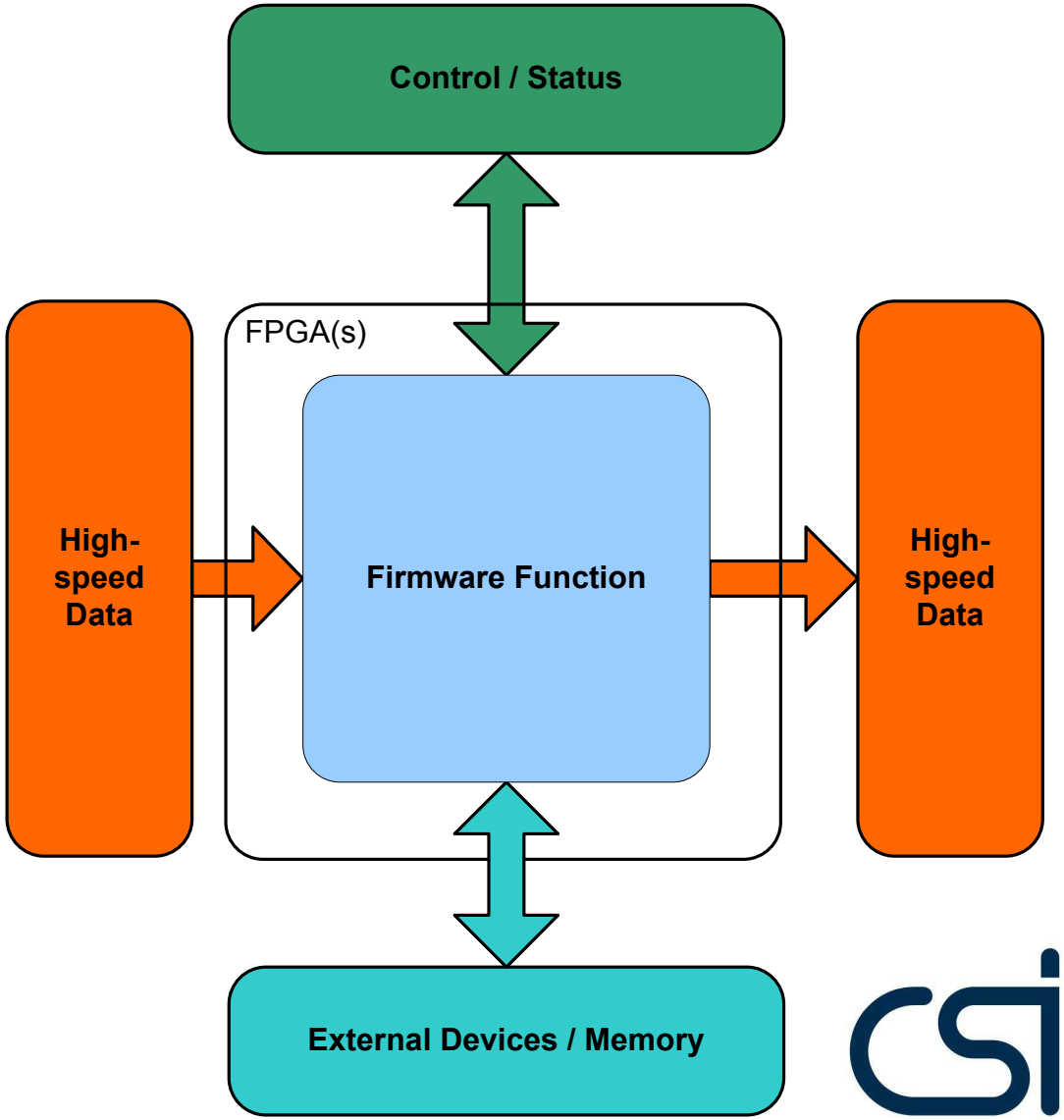
4. Board Support Package (BSP)

- Make in-house boards usable
- Appropriate BSP crucial
  - Memory
  - Communications
  - Control
  - Monitoring
  - Real-time data viewing
  - Real-time data capture
- BSP abstract FPGA
  - Constrains function
  - Not dictate design

3. BSP Components

- Components of the BSP:
  - Firmware wrapper (Processing FPGAs)
  - System controller (SCON)
  - RSP controller (RSPCON)
  - CCM FPGA
- Extra systems:
  - Display processor (DIP)
  - Data Acquisition Processor (DAP)

3. BSP Firmware Wrapper



3. BSP Control Interface

0	FPGA	Size = 8
1	Function A	Size = 4
2	Group M	Size = 3
3	Control / Status Register	
4	Control / Status Register	
5	Function B	Size = 3
6	Group N	Size = 2
7	Control / Status Register	

3. BSP Dynamic Memory Map

- Command Interpretation
  - Address translation on each layer
  - Group size and implementation
- System Usability
  - Able to determine the meaning of each group
  - Can apply a template to define registers
  - Search for and use any register in any group
  - Rescan can automatically pick-up changes

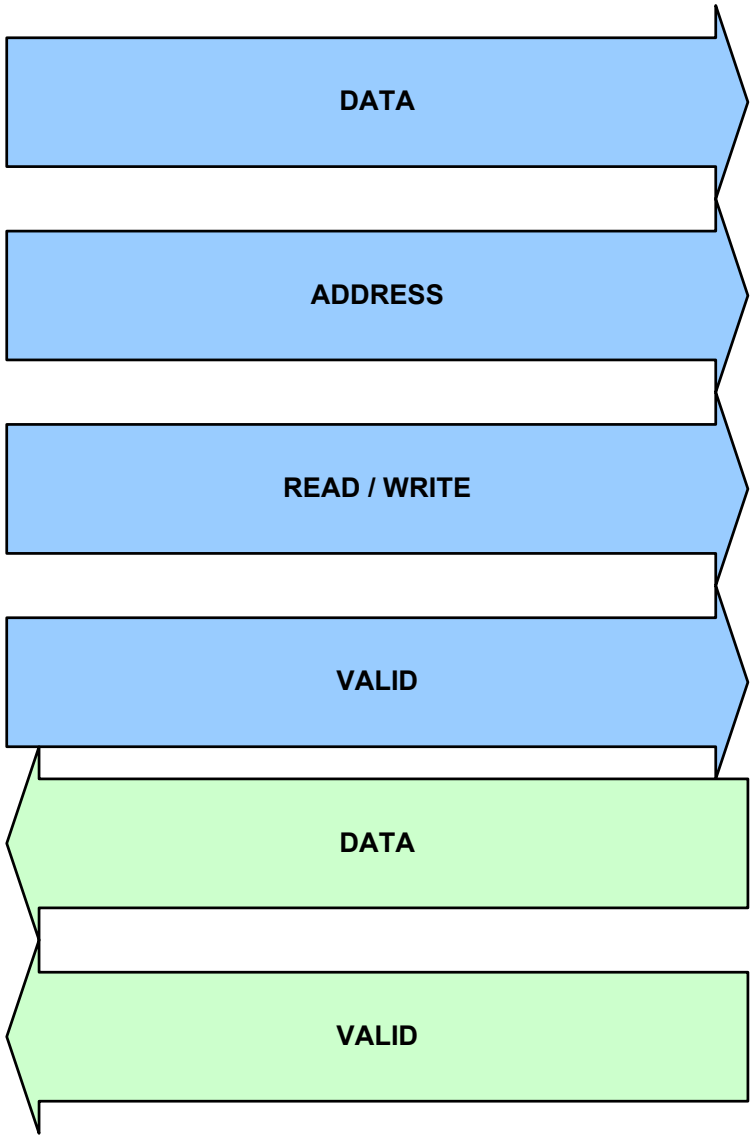
our future through science

3. BSP High-speed data interface

- Based on Xilinx RocketIO
  - Utilising Xilinx Aurora (free)
  - SerialRapidIO, PCIe, 10GBE & FibreChannel
- Combine two RocketIO channels 6.4 Gbps
  - Carry 128-bit data at 40 MHz (640 MBps)
  - Potentially can combine up to 8 links!
  - Data conveyed as bursts with headers.

our future through science

3. BSP External Interfaces



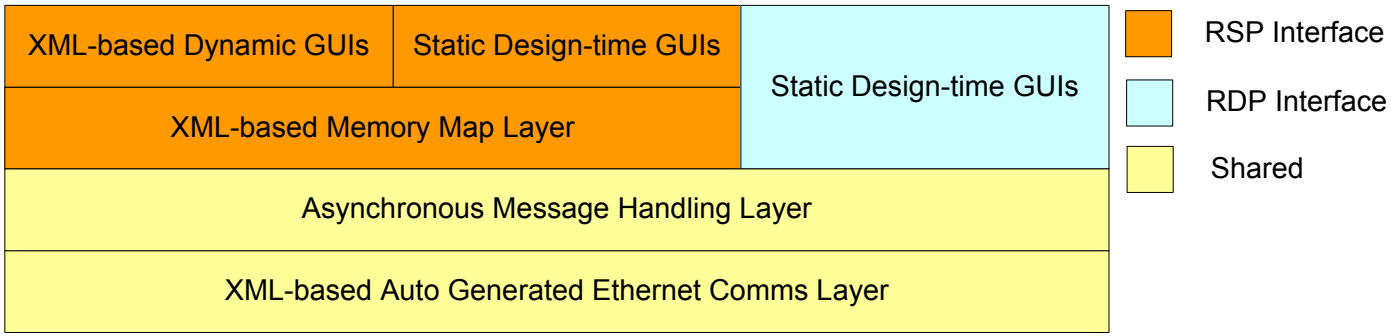
CSIR
our future through science

Part B: Firmware & Software Development Topics

1. FPGA Firmware Background
2. System Design Goals
3. Board Support Package
4. **BSP Enabling Systems**

4. System Controller (SCON) Overview

- Control Software Application for signal processor
- Architecture
  - Layered OO design using C++ Builder (Windows)



- Role
  - User interface for the signal processor
  - RSP Setup, Control, Monitoring and BIT
  - Centralised Facility Mode and Status Indication
  - Developed in-house

CSIR
our future through science

4. SCON Features

- High level of re-configurability using XML
  - Add extra processing functions
  - No recompile necessary
- Extensible event-driven OO framework
  - Add new software functionality
- Highly multi-threaded design allows multiple concurrent background and foreground tasks

our future through science

4. SCON Reconfigure-ability

- VME Boards
- Memory Map in XML format (Gateway to Hardware)

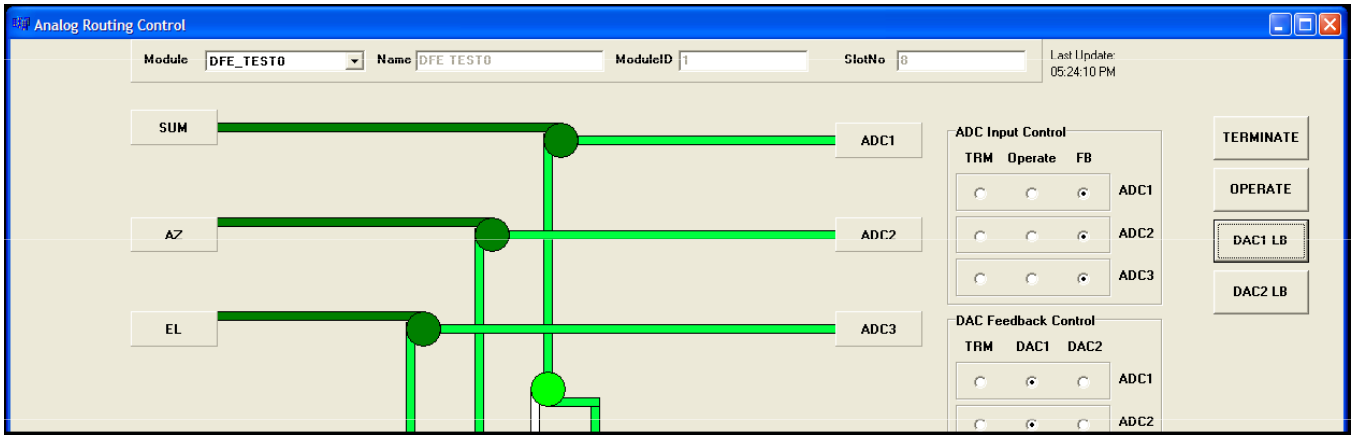
```
<Register Name="STATUS_REGISTER" Address="0x00000000"/>
```

- Static Memory Map
  - Board level registers
  - Parsed after VME discovery process
- Dynamic Memory Map (TLV representation)
  - Discovery process:
    - Read TLV-type registers of function
    - XMLs parsed based on structure and type value.
  - Rediscovery when:
    - New Firmware Component added
    - Firmware Component changes

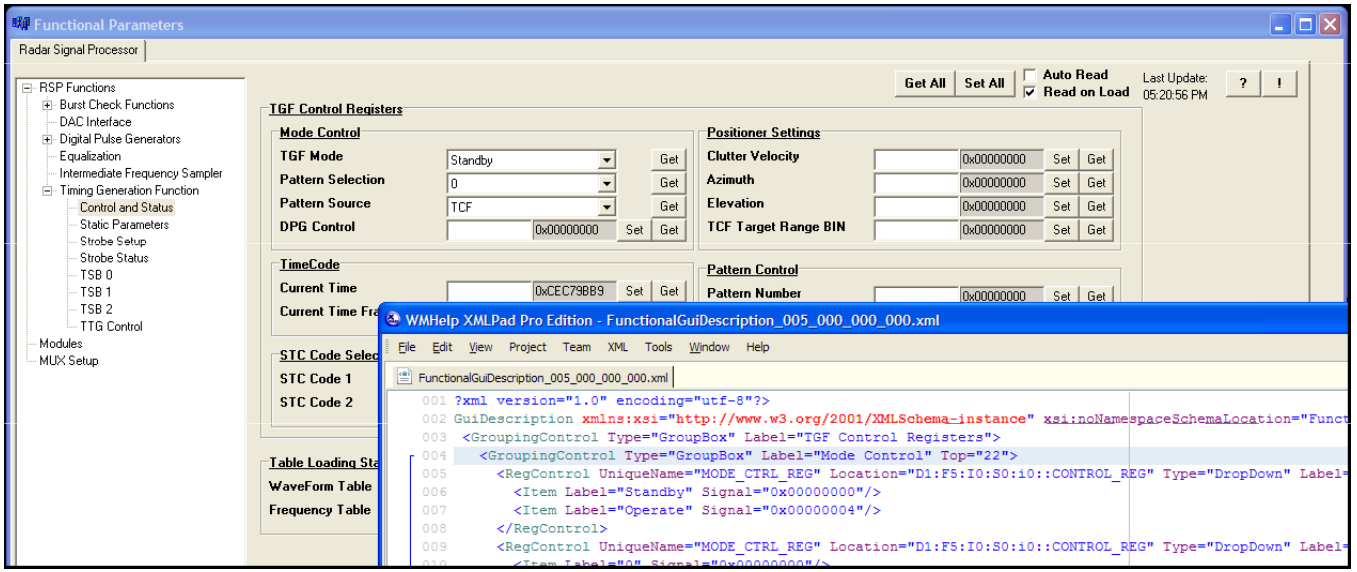
CSIR
our future through science

4. SCON GUI Types

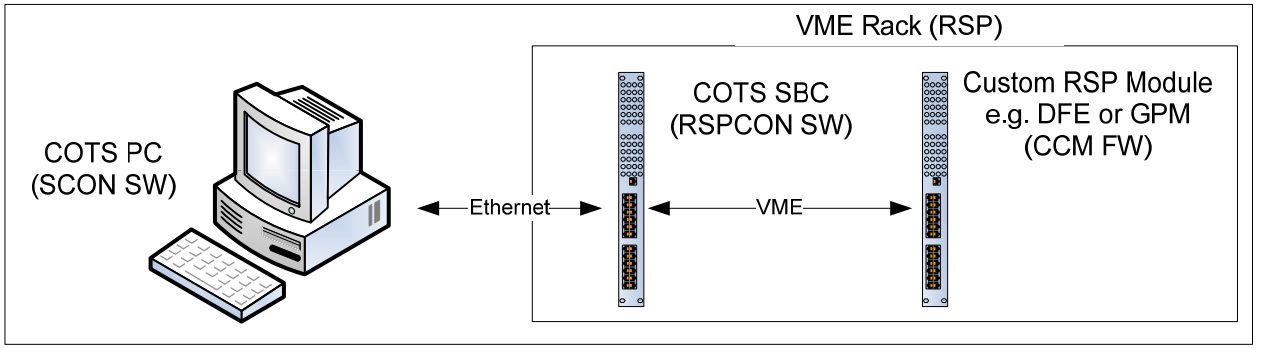
- Static design-time GUI



- XML-Based Dynamic GUI



4. Ethernet-VME Bridge



- RSPCON - COTS Single-board VME PC
  - Ethernet Messages ↔VME Transactions
  - Decouples SCON from VME standard
  - Runs on VxWorks
- CCM – Configuration Control and Monitoring FPGA
  - VME Transactions ↔ LVDS (or GBE ↔ LVDS)
  - Monitors hardware
  - Configures FPGAs
  - Determines placement of functions on FPGAs (Dynamic Memory Map)

our future through science

4. Display Processor

- Components:
  - Embedded PowerPC 440 running VxWorks
  - Display GUI
- Functions performed:
  - Display data from any processing point
  - Intensity image plot, 3D & 2D Waterfall plot A-trace & R-trace
  - Implemented using T-chart
- Enables:
  - Verification of processing functions
  - Provides human comprehensible information

CSIR
our future through science

4. Data Acquisition Processor (DAP)

- Components:
  - DAP Console: GUI for setup and recording
  - Curtiss-Wright Phoenix M6000 VXS Card
  - Fibre-optic Storage Area Network

- High Speed Data Capture
  - From any processing point on any card:
    - data filtering
    - interleaving multiple channels
  - Supports recording data rates up to 640 MB/s (SI Mega)
  - 3.6 TB of storage capacity equates to 1 hour recording of LYNX RSP data at full data rate

CSIR
our future through science

Part B: Firmware & Software Development Topics

1. FPGA Firmware Background
2. System Design Goals
3. Board Support Package
4. BSP Enabling Systems
5. **Data Processing Firmware**

5. Data Processing Firmware

- Types of functions:
  - Generation of timing & signals
  - Data conditioning functions
  - Mathematical processing
- Required elements:
  - Memories
  - Adders, subtractions, multipliers & dividers
  - State-machines
- Design constraints:
  - FPGA configurable logic
  - FPGA DSP building blocks
  - Memory capacities

CSIR
our future through science

5. How to Create a Data Processing Function

- Control / Status Interface:
  - Compile list of configurable parameters
  - Determine the necessity for tables of data
  - Implement the required DMM interface

- Data Processing:
  - Partition available FPGA resources per function
  - Using actual numbers determine which algorithms will fit
  - Make use of over-clocking and SIMD to expand the available resources
  - Implement the chosen algorithm using suitable tools

CSIR
our future through science

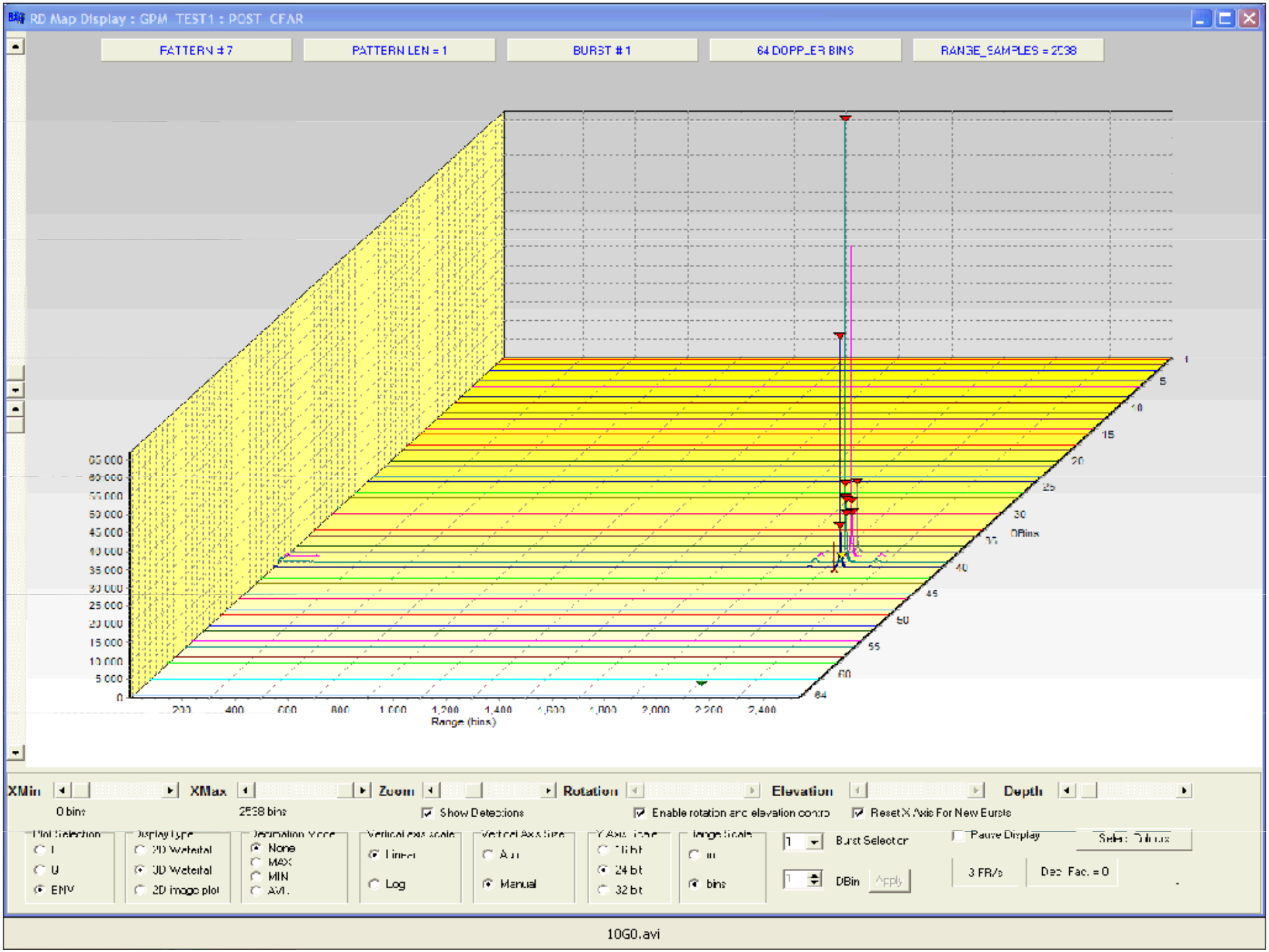
Part B: Firmware & Software Development Topics

1. FPGA Firmware Background
2. System Design Goals
3. Board Support Package
4. BSP Enabling Systems
5. Data Processing Firmware
6. **Development Issues**

CSIR
our future through science

6. Development Issues

- Mapping firmware into clock regions
- Full FPGAs (> 75%) cause routing problems
- High speed designs > 400 MHz placement sensitive
- Automatic BRAM and DSP mapping: poor timing & no control
- IP cores require a lot of FPGA resources: sometimes have bugs
- Tools over-promise & under-deliver


RD Map Display : GPM TEST1 : POST CFAR
PATTERN # 7
PATTERN LEN = 1
BURST # 1
64 DOPPLER BINS
RANGE_SAMPLES = 2638
Range (bins)
DBins
XMin
XMax
Zoom
Rotation
Elevation
Depth
0 bins
2638 bins
Show Detections
Enable rotation and elevation control
Reset X Axis For New Bursts
Plot Selection
I
Q
ENV
Display Type
2D Waterfall
3D Waterfall
2D Image plot
Decimation Type
None
MAX
MIN
AVE
Vertical Axis Scale
Linear
Log
Vertical Axis Size
Auto
Manual
Y Axis Type
16 bit
24 bit
32 bit
Range Scale
m
bins
Burst Selector
DBin
Apply
Pause Display
Save Options
3 FR/s
Dec Fac. = 0
10G0.avi

Conclusion

- Adaptable & scalable system
- Easy use through BSP
- Generic data processing
- Covered development issues

our future through science

Thank you

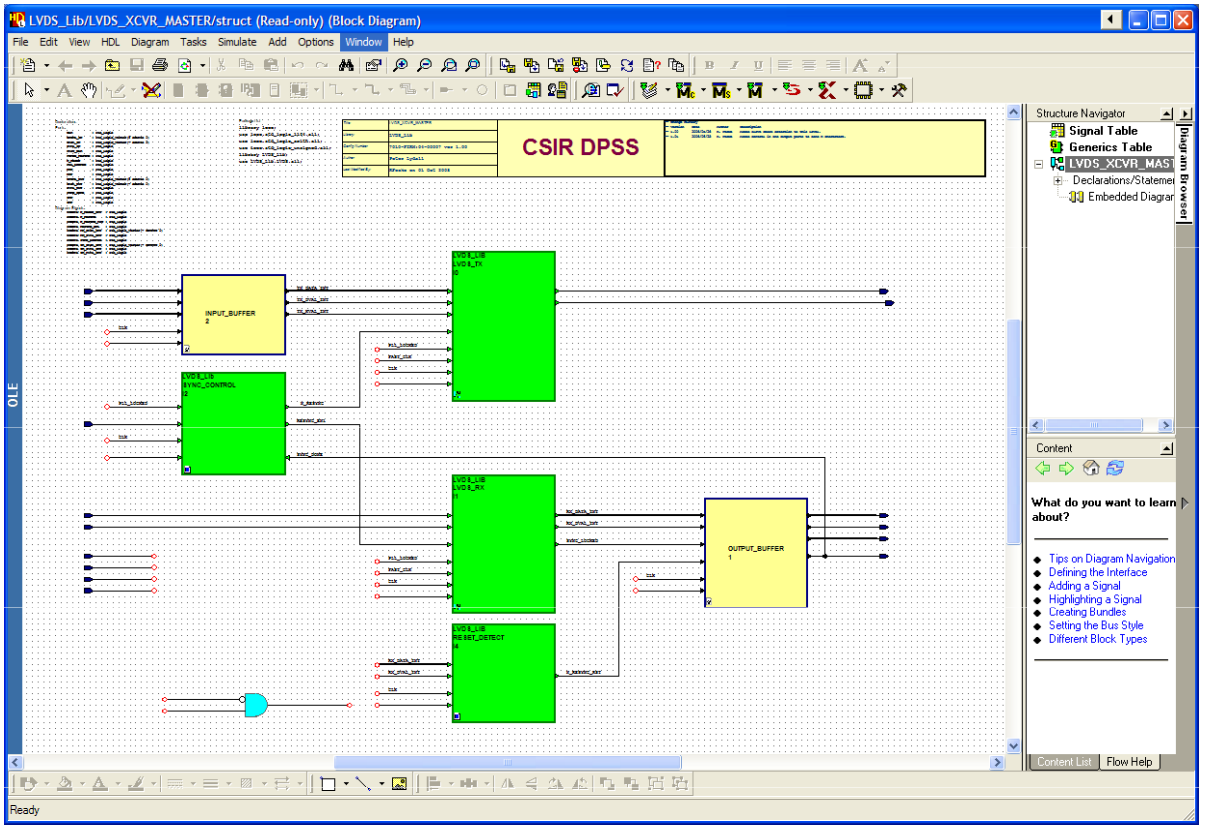
Questions?

Part B: Firmware & Software Development Topics

A. Development Tools

our future through science

A. Mentor Graphics HDL Author & Modelsim



- Lynx LVDS Design

our future through science

A. Mentor Graphics HDL Author & Modelsim

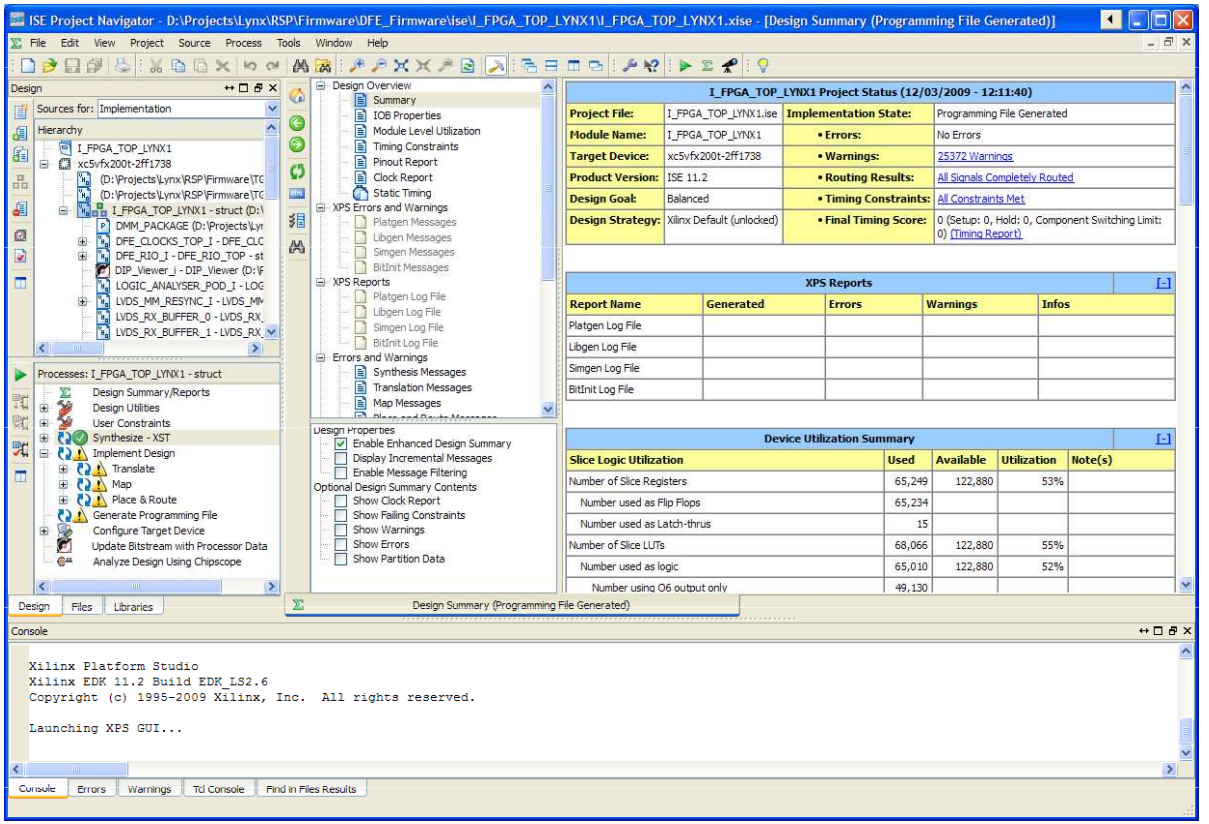
- Hierarchical Graphical Design for HDL

- Pros:
  - FPGA Vendor Independent
  - Graphical view intuitive & easy to manipulate
  - Hierarchical design: maintainable & reusable
  - Only leaf nodes of hierarchy coded in HDL

- Cons:
  - Not an integrated development environment
  - Expensive license
  - Poor integration with some design tools

CSIR
our future through science

A. Xilinx Integrated Synthesis Environment (ISE)



- Lynx TGF ISE Project

our future through science

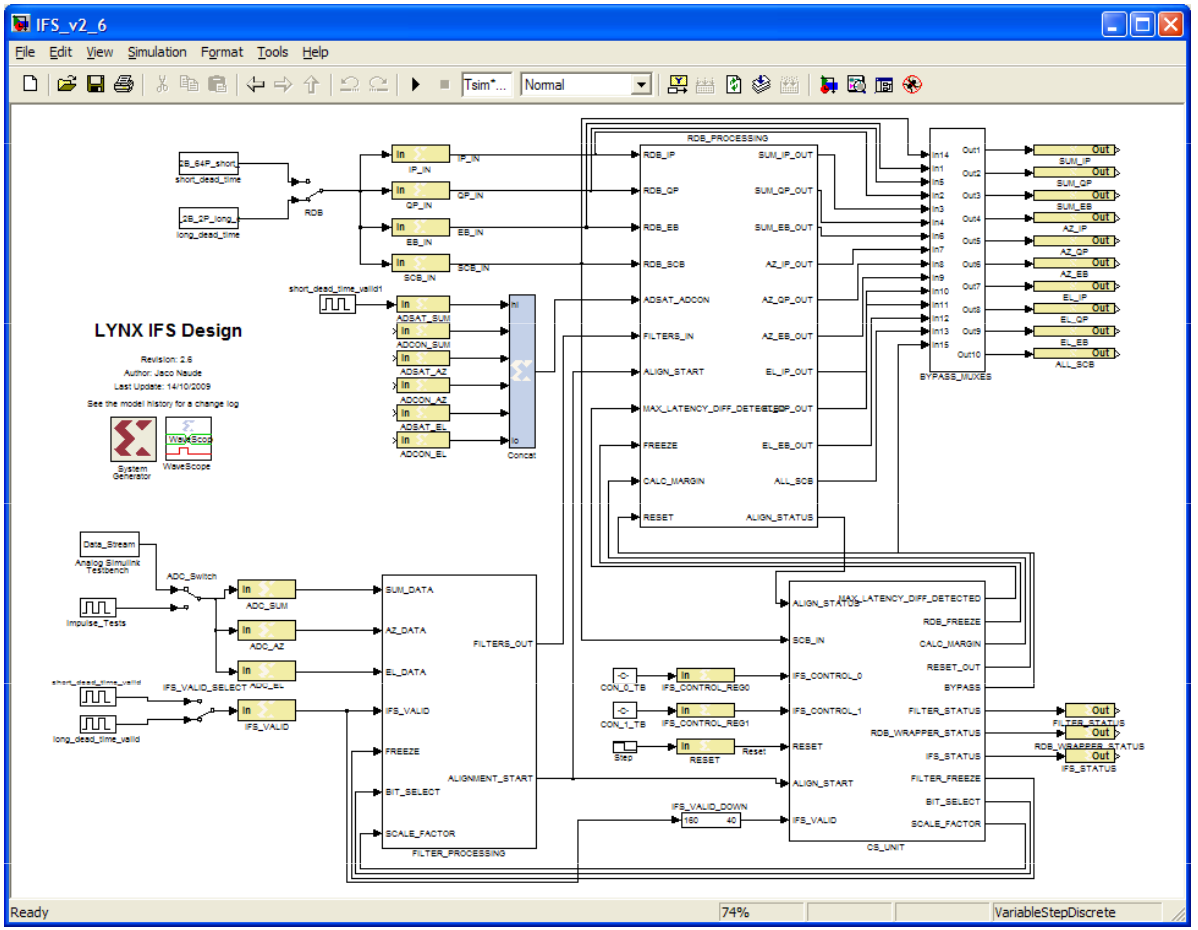
A. Xilinx Integrated Synthesis Environment (ISE)

- Integrated Design Environment for Xilinx

- Pros:
  - Quick and easy access to all design flows
  - Integrates well with all Xilinx applications
  - Easy to use and configure

- Cons:
  - Specific to Xilinx
  - Poor file management
  - Expensive license

A. System Generator for DSP



- Lynx IF Sampler Example

our future through science

A. System Generator for DSP

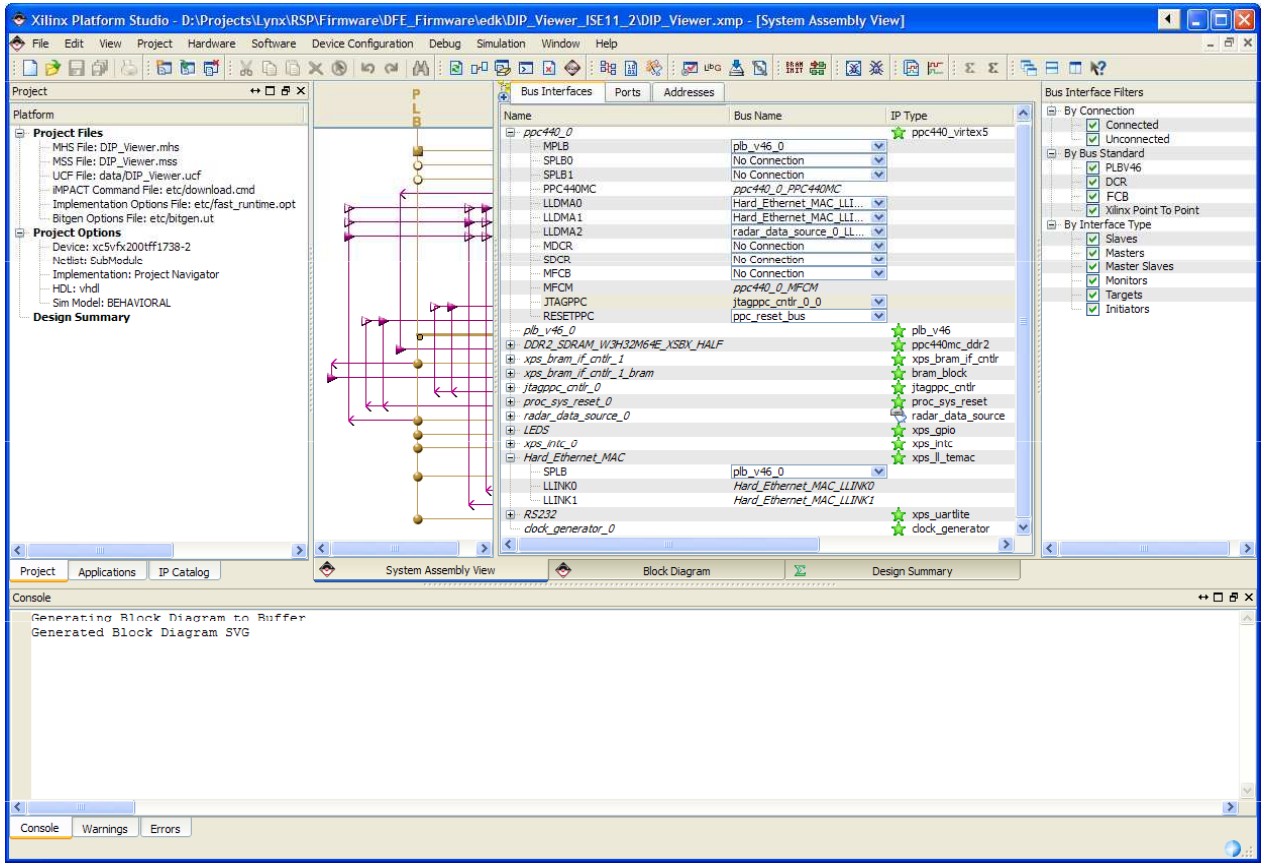
- High Level DSP Design Environment

- Pros:
  - Easy implementation of DSP algorithms
  - Visualisation tools for multi-rate systems
  - Big ready-to-use block set
  - Hardware in the loop simulation
  - Simulate using existing Matlab Simulink blocks

- Cons:
  - High level: difficult to trace problems
  - Poor state-machine design control
  - Matlab based, expensive

CSIR
our future through science

A. Xilinx Embedded Development Kit (EDK)



- Lynx DIP Embedded Processor

our future through science

A. Xilinx Embedded Development Kit (EDK)

- Embedded Processor Implementation

- Pros:
  - XPS: build processor subsystem quickly
  - SDK: eclipse IDE with GNU debugger
  - IP block set for processor type functions
  - Support for Linux, VxWorks and Xilkernel

- Cons:
  - Integrating own IP not well managed
  - Black-box approach to IP
  - Specific to Xilinx

CSIR
our future through science