Dynamically Reconfigurable FPGA

Dr. Shubhajit Roy Chowdhury

Centre for VLSI and Embedded Systems Technology,

IIIT Hyderabad

Selecting a Target FPGA

Factors to be considered:

- Dynamic reconfigurability
- Reconfiguration time
- Partial vs Full reconfiguration
- Granularity

Dynamic reconfigurability

- The ability of a FPGA to modify operation during runtime.
- Correct FPGA selection very important.
- The primary advantages of run-time reconfiguration in devices are reduced power consumption, hardware reuse and flexibility.
- Main problem is the speed of reconfiguration.

Granularity

Two main architectures:

- Course grained: smaller number of larger, more powerful logic blocks

Advantage:

Faster because of easy routing

- Fine grained : consists of a large number of small logic blacks

Advantage:

Good utilization

Easy conversion to ASIC

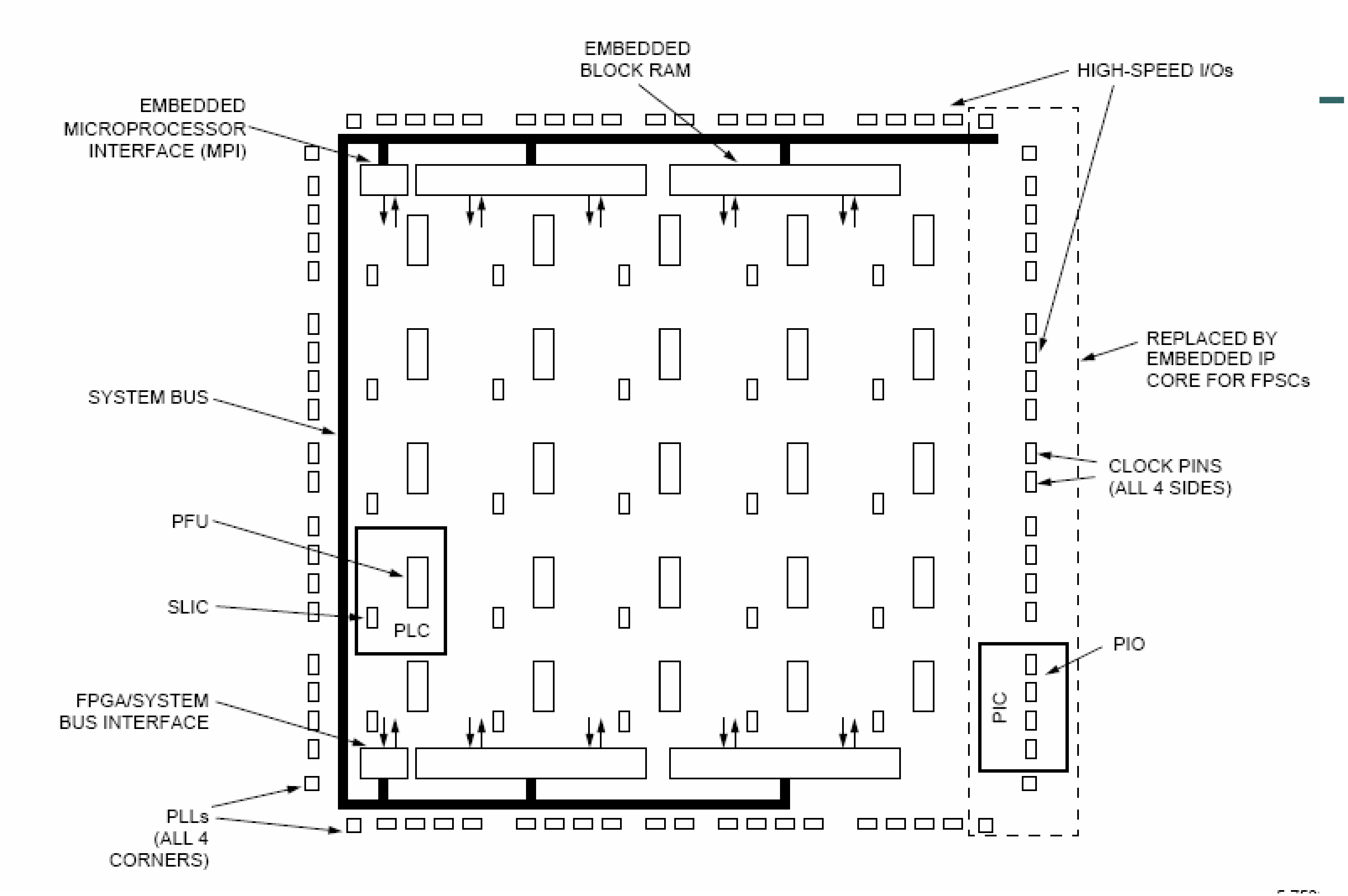
Configuration of Xilinx Virtex

- Fully and Partially reconfigurable
- Module-Based partial reconfiguration
- Distinct portions of an FPGA are referred to as reconfigurable modules.
- Used for independent design applications
- Small-Bit Manipulations
- accomplished by making a small change to the design
- Switching configuration is fast as bitstream difference is smaller than device difference.

Lattice ORCA Architecture

- Coarse grained architecture
- Four basic elements: programmable logic cells (PLCs), programmable input/output cells (PIOs), embedded block RAMs (EBRs), and system-level features.
- Programmable functional unit (PFU) is the basic functional unit containing eight 4-input LUTs, 8 latches/FFs and one additional flip-flop for arithmetic functions

Lattice OCRA block diagram



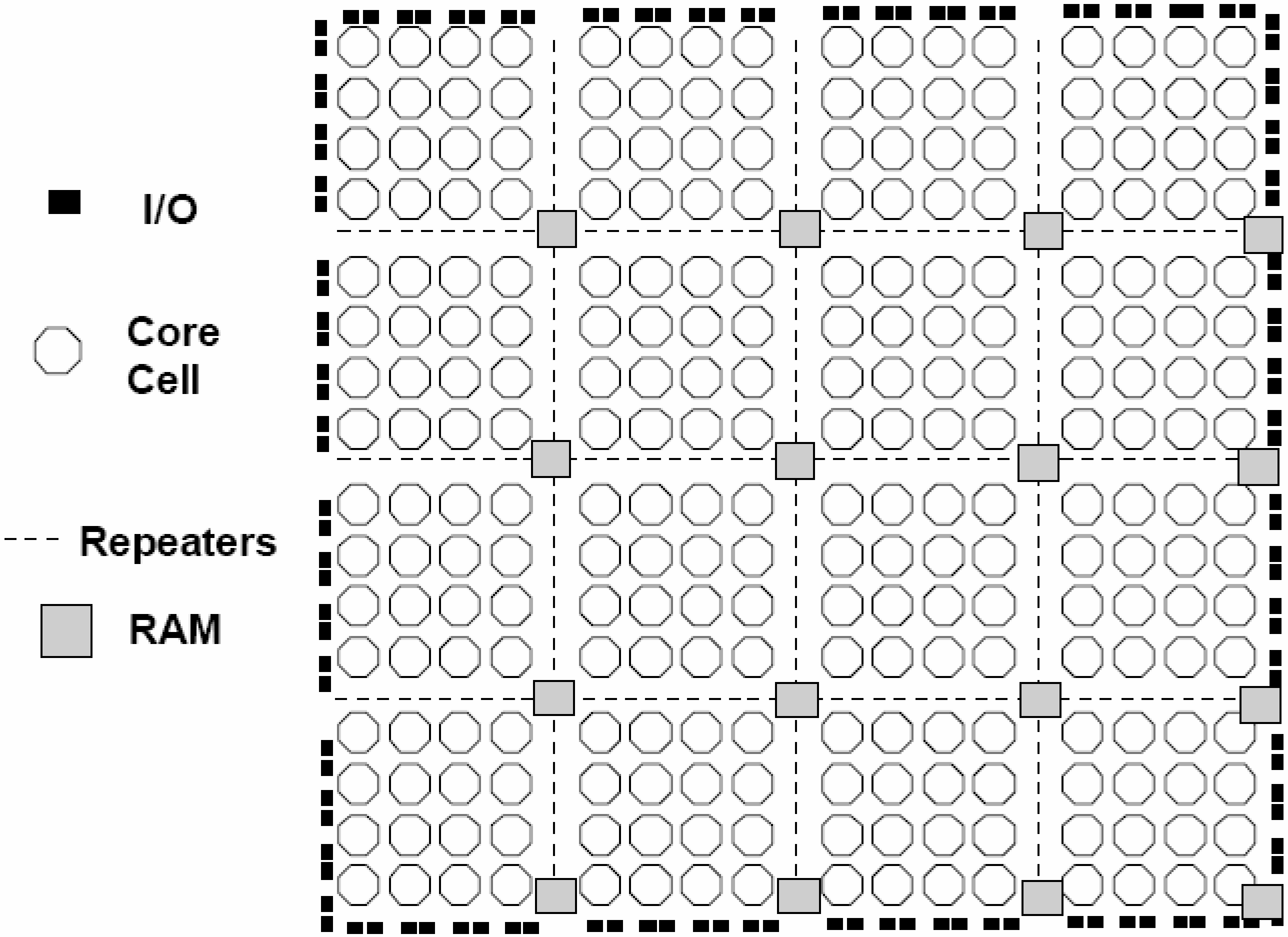
Configuration of Lattice ORCA

- Fully and Partially reconfigurable
- Partial reconfiguration is done by setting a bitstream option that tells the FPGA to not reset the entire configuration
- Options available to allow one portion of FPGA to remain in operation while the other is being reconfigured
- Off chip reconfiguration

Atmel AT40K Architecture

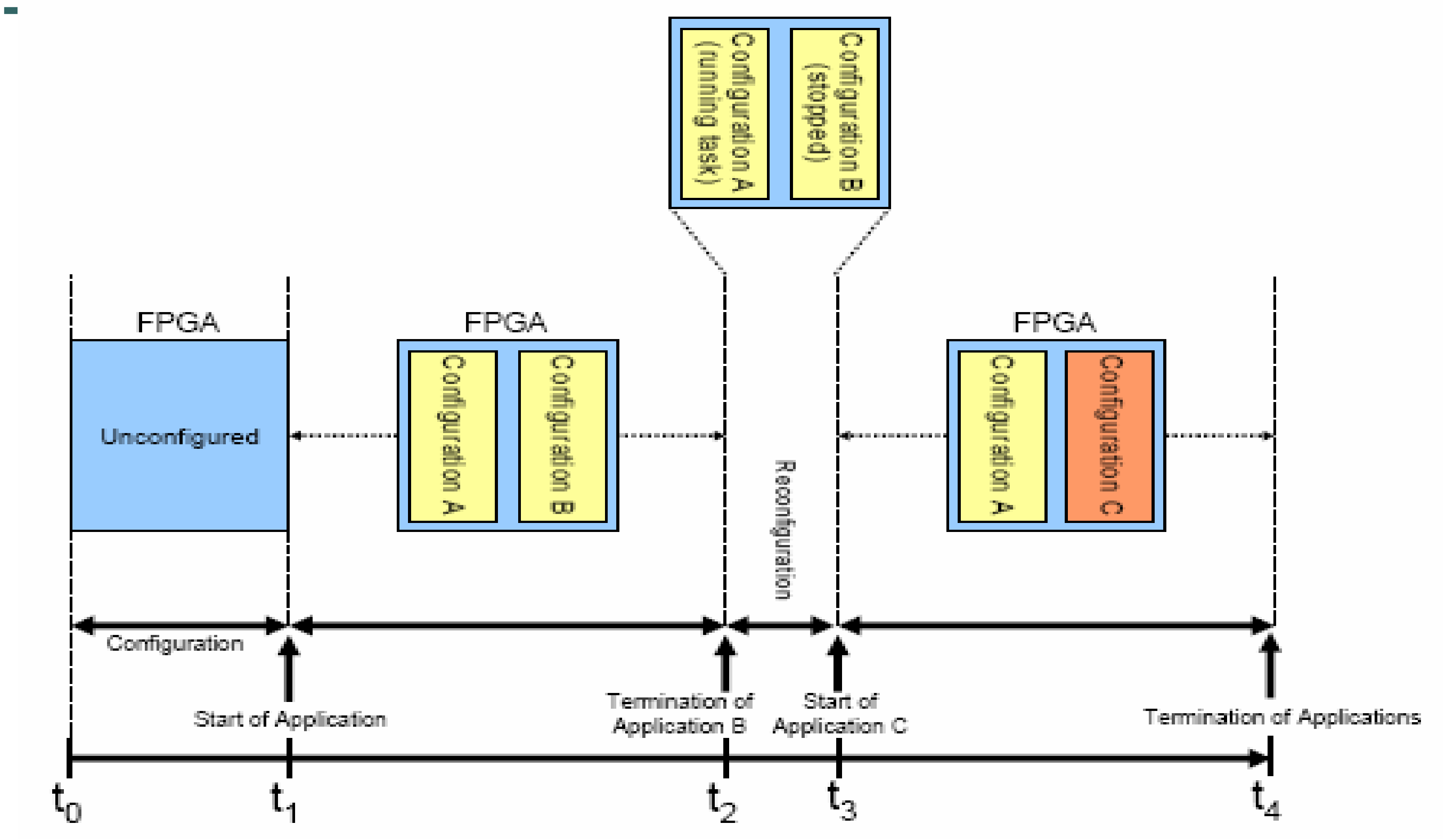
- Fine Grained architecture
- Symmetrical array of identical cells
- 8-sided core cells with direct horizontal, vertical, and diagonal cell-to-cell connection
- Small cells lead to large number of cells which leads to greater functionality
- Each cell can implement 2 Boolean operations of 3 inputs or 1 operation of 4 inputs

AT40K Device Overview



- Array sector size is 4 x 4
- Distributed RAM
- Flexible I/O

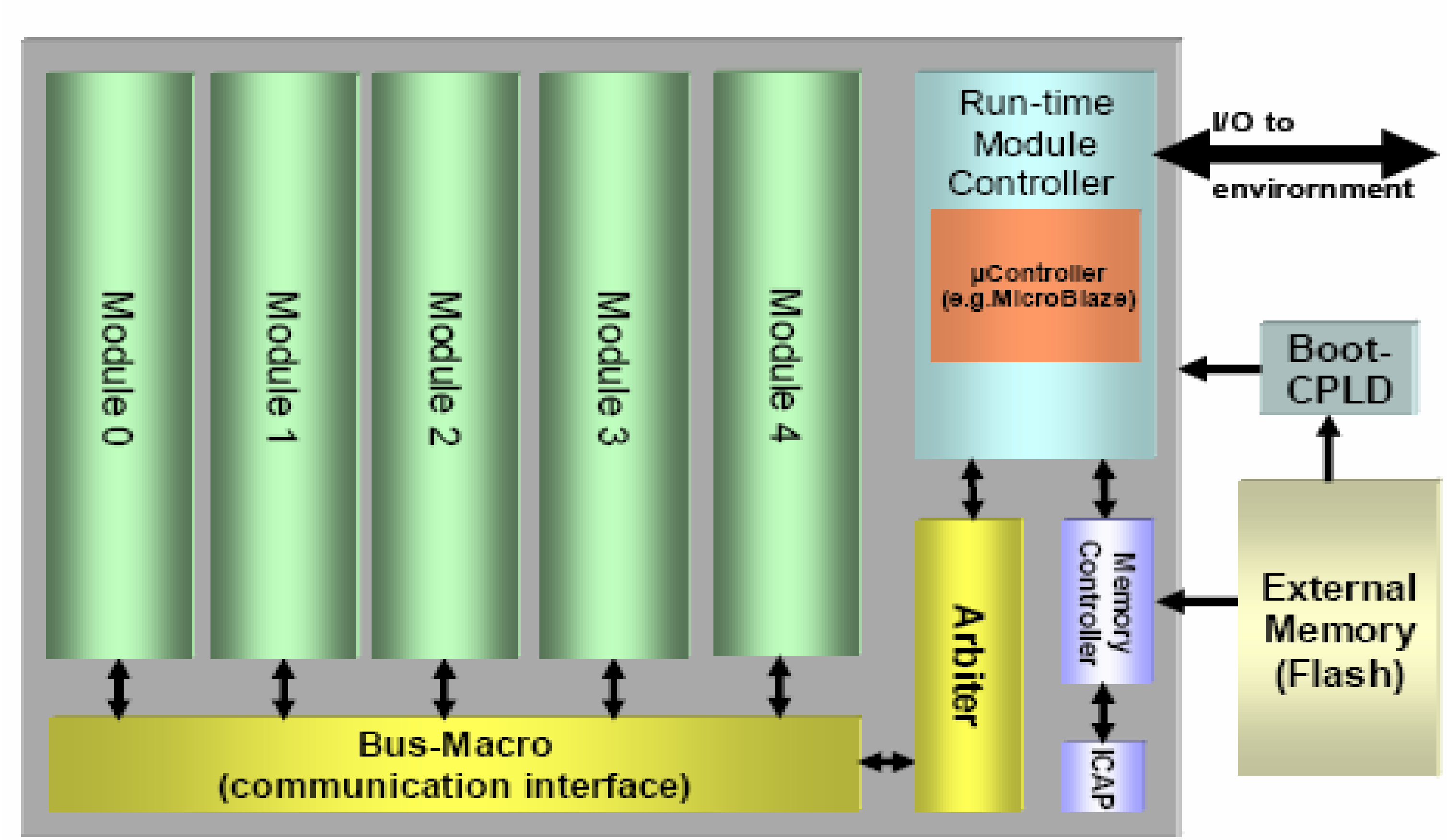
Partial Reconfiguration



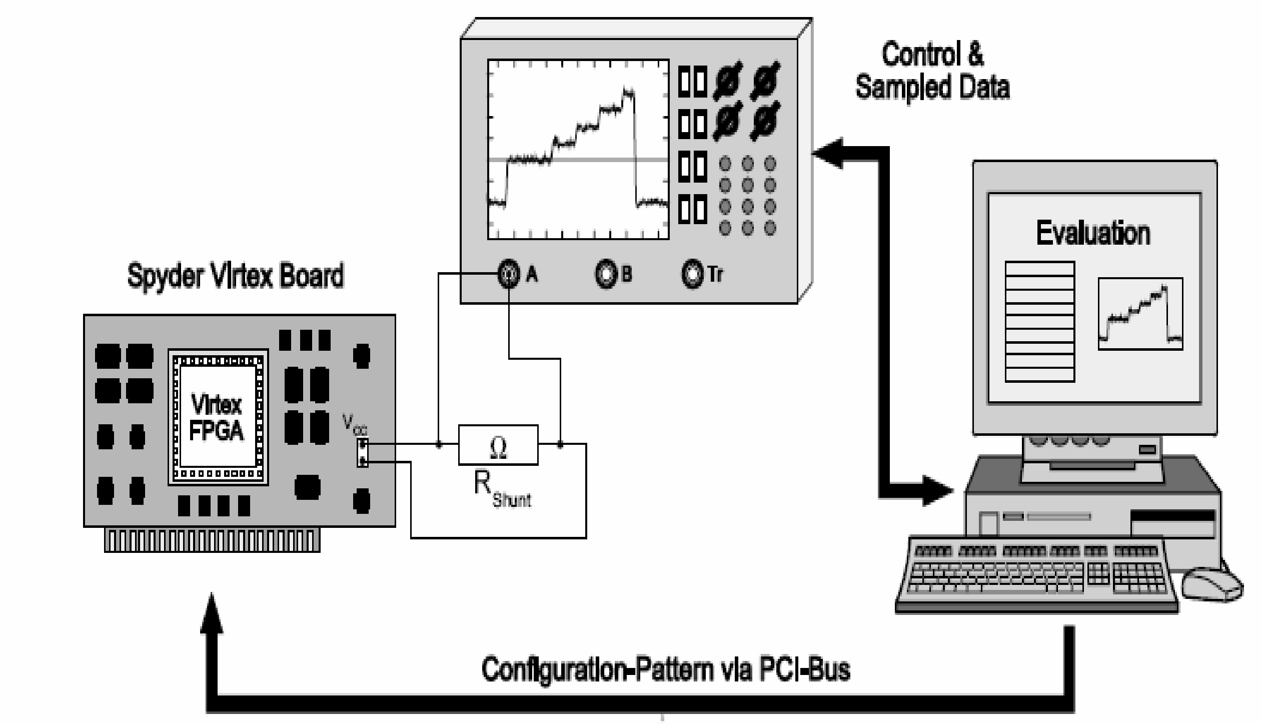
Advantages of Dynamic Reconfiguration

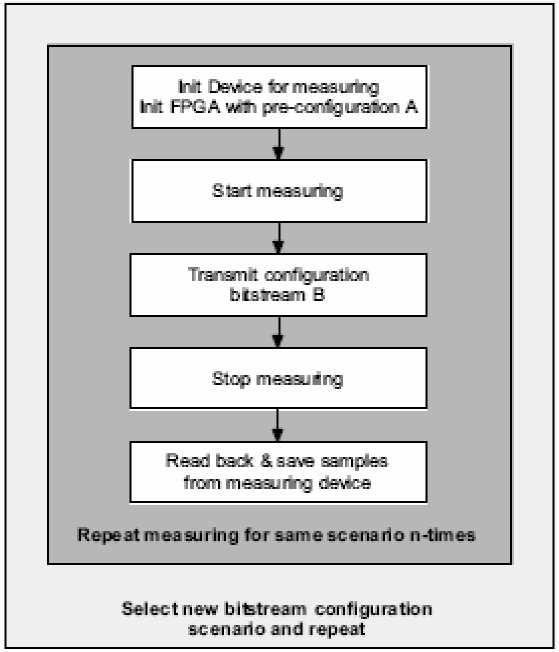
- Power/Size/Cost Reduction
- Hardware Reuse
- Obsolescence Avoidance
- Application Portability

Slot Based Dynamically Reconfigurable System



Power Consumed During Reconfiguration





Init Device for measuring
Init FPGA with pre-configuration A

Start measuring

Transmit configuration bitstream B

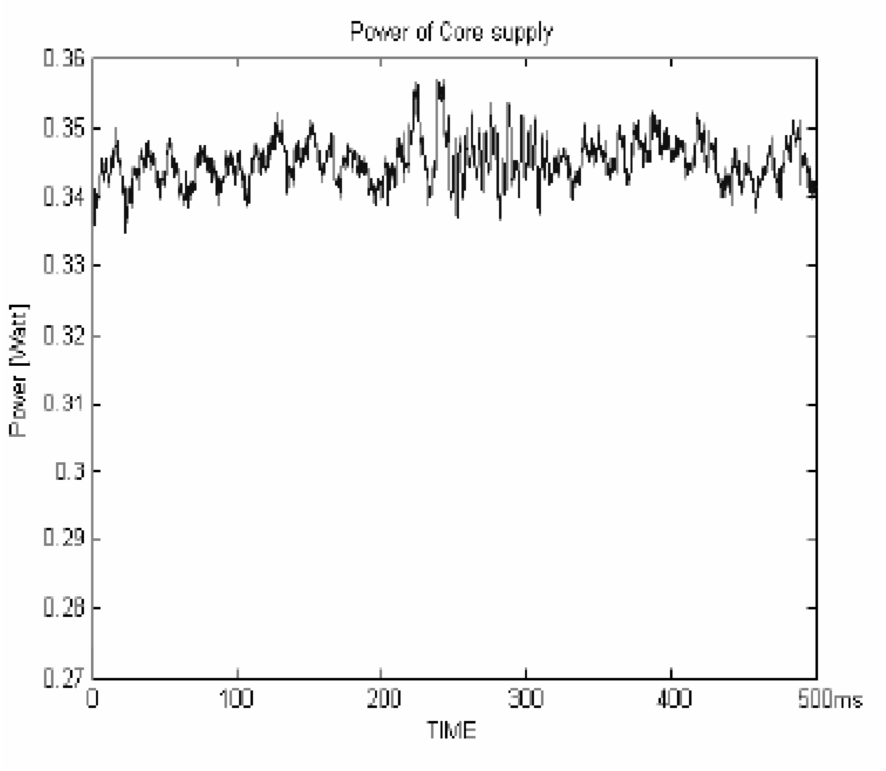
Stop measuring

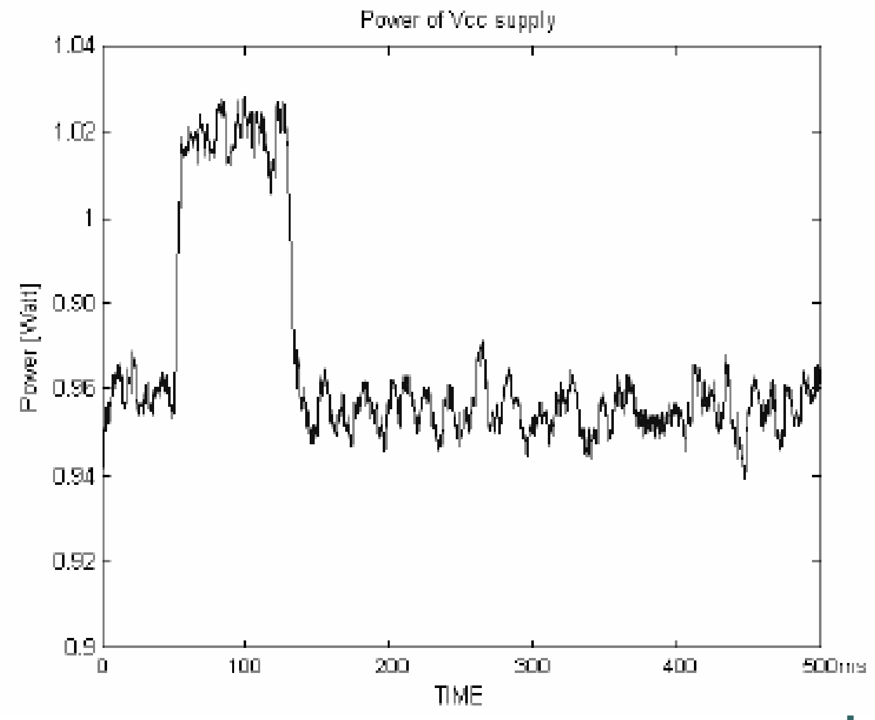
Read back & save samples from measuring device

Repeat measuring for same scenario n-times

Select new bitstream configuration scenario and repeat

Power Consumed During Reconfiguration





Virtex-4 LX15 FPGA Layout

- Overall Structure
  - CLBs – Configurable logic blocks
  - IOBs – Input-output buffers
  - DSP48s – Xilinx's digital signal processing units
  - BRAMs – Block Random Access Memories
  - FIFOs – First-in First-out buffers
  - DCMs – Digital Clock Managers

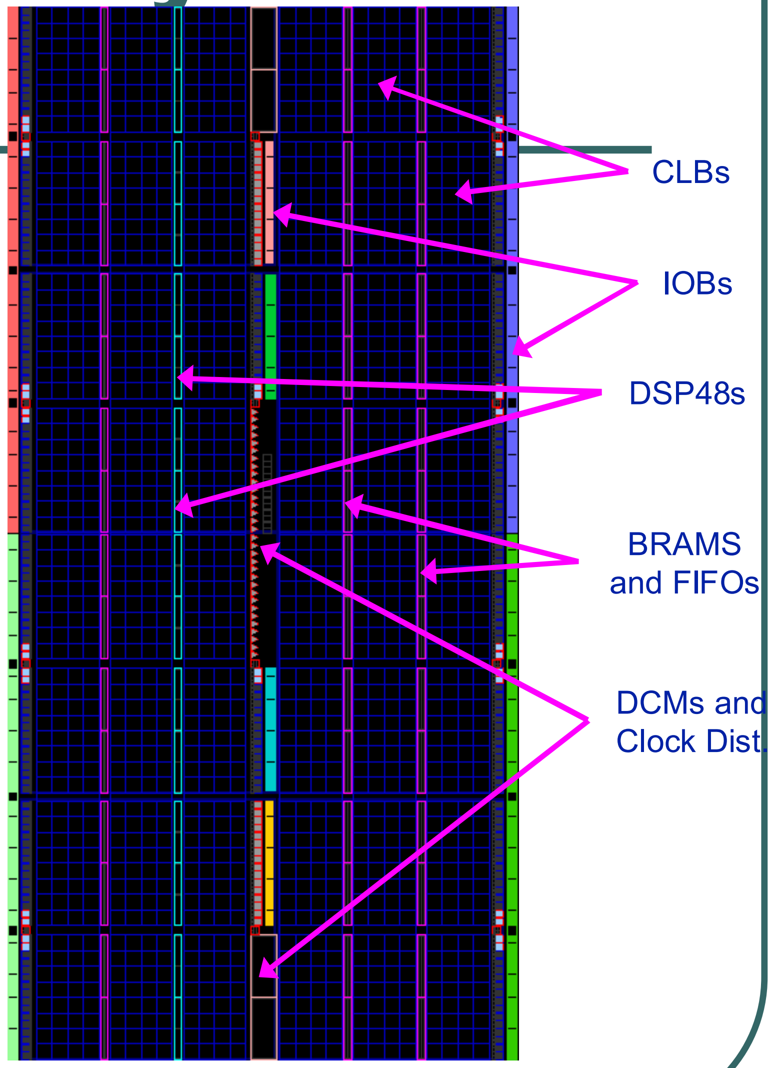


Figure 1. Virtex-4 LX15 FPGA layout

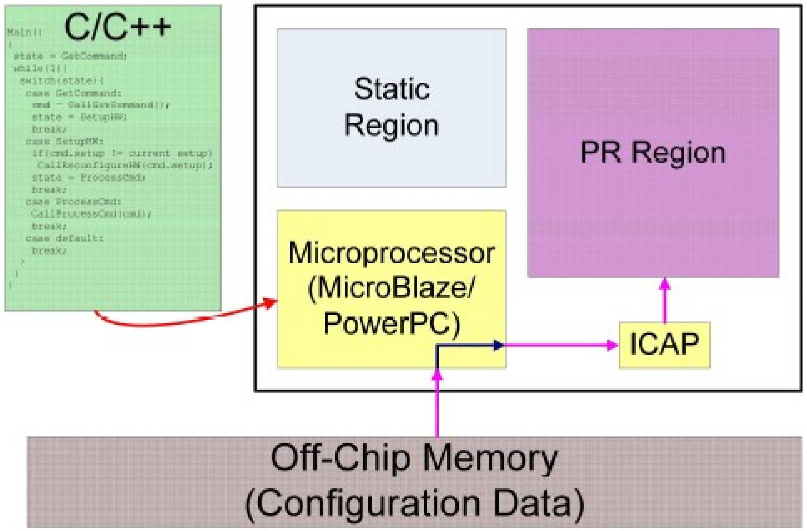
Virtex-4 Reconfiguration

- FPGA is reconfigured by writing bits into configuration Memory (CM)
- Configuration data is organized into frames that target specific areas of the FPGA through frame addresses
- To reconfigure any portion of that frame the partial bit-streams contain configuration data for a whole frame
- Reconfiguration times highly depend on the size and organization of the PR regions
  - Virtex-II allowed column based PR only
  - Virtex-4's allow arbitrarily sized PR regions
- Virtex-4 Frames
  - Composed of 41 32-bit words
  - The LX15 has 3,740 frames
- Methods of reconfiguring a device, each has applications where desirable
  - Externally
    - Serial configuration port
    - JTAG (Boundary Scan) port
  - Internally
    - Though the Internal configuration access port (ICAP) using an embedded microcontroller or state machine

Reconfiguration using an Embedded Microcontroller

- Many Xilinx FPGAs have Embedded hard processor cores
  - Processor core has the ability to process C/C++ code
  - Makes reconfigurable designs extremely flexible since no need for external control

- Reconfiguration Steps
  - Reconfiguration is triggered within the FPGA
  - Processor core loads the desired configuration data from external reconfiguration memory
    - This could be from ROM, Flash, static Ram loaded at startup or filled up by the FPGA itself
  - Processor reconfigures the PR region through the ICAP primitive



```
Main()
{
 state = GetCommand;
 while(1){
  switch(state){
   case GetCommand:
    cmd = CallGetCommand();
    state = SetupHW;
    break;
   case SetupHW:
    if(cmd.setup != current setup)
     CallReconfigureHW(cmd.setup);
    state = ProcessCmd;
    break;
   case ProcessCmd:
    CallProcessCmd(cmd);
    break;
   case default:
    break;
  }
 }
}
```

C/C++
Static Region
PR Region
Microprocessor (MicroBlaze/ PowerPC)
ICAP
Off-Chip Memory (Configuration Data)

Figure 2. PR design using embedded microcontroller

Conclusions

- Design time overhead involved when creating a PR design is acceptable but requires progressing though a slow learning curve before any results are obtained
- Full benefits of PR will not be evident until it becomes commonplace in industry and vendors place more resources on supporting the PR design flow and keeping documentation up to date
- Huge power and area is saved
- PR is slowly replacing processors especially in high performance computing systems

Thanks ☺