CS245 – Lecture 7

Reconfigurable Computing

Tony Givargis

Introduction

- Reconfigurable computing, a new paradigm for system design
  - Based on FPGA technology
    - Larger gate count
    - Faster clock
    - Cheaper cost
  - Post fabrication hardware computation
- Underlying technology enables swapping digital circuits:
  - In real-time (i.e., matter of seconds or milliseconds)
  - In circuit
  - On-the-fly (i.e., during execution)
- Application domains
  - Telecon
  - Automotive
  - Settop boxes
  - Space
  - Desktop
  - Mobile

Speedup

- DNA matching
  - FPGA vs. Sparc => 4300x
- RSA crypto
  - FPGA vs. Alpha => 17.8x
- Ray casting
  - FPGA vs. Pentium => 33.8x
- FIR filter
  - FPGA vs. DSP => 17.9

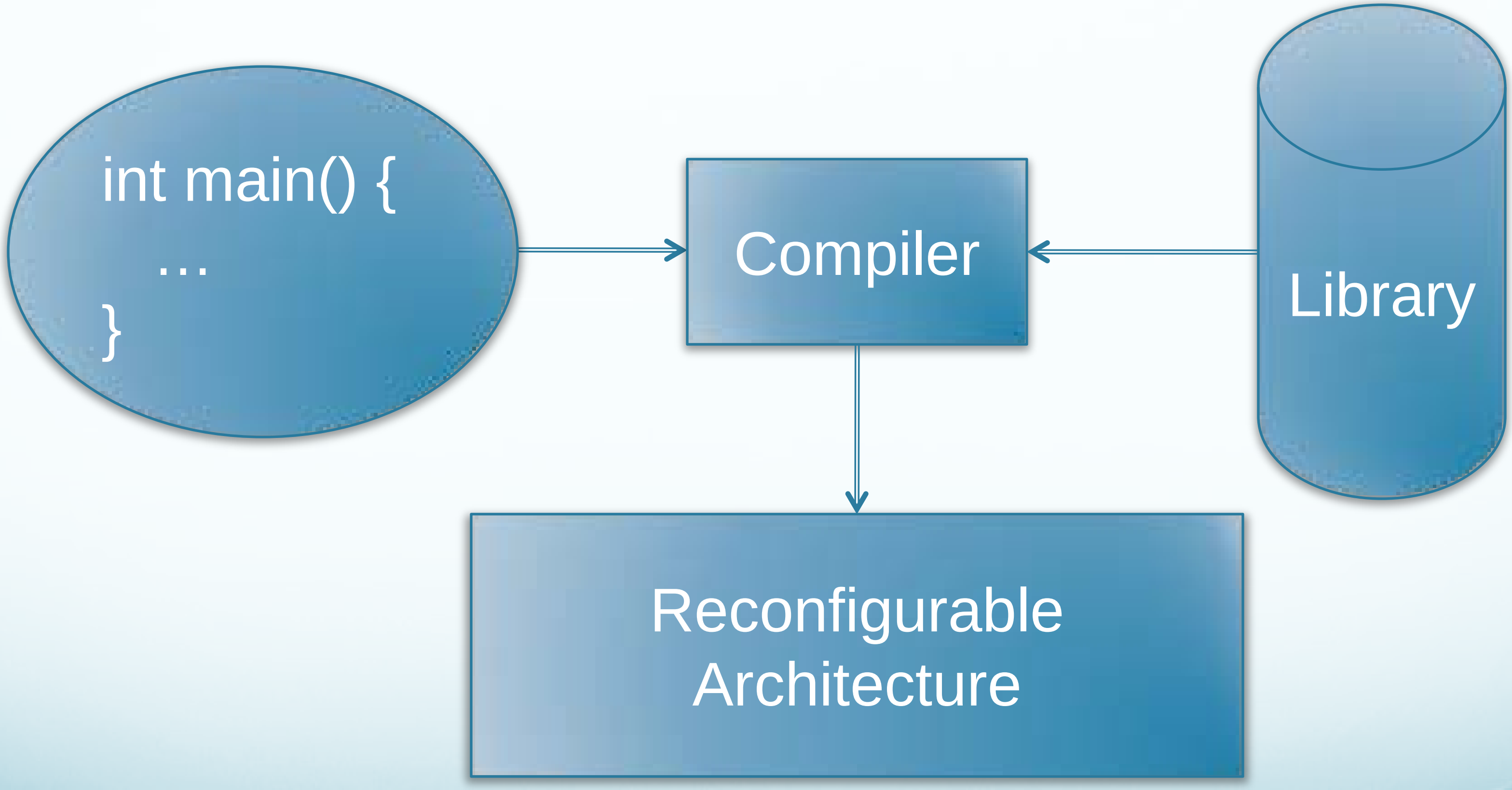
Why R.C.

- Changes in application requirements
  - Most new features are enabled by software!
  - Changing standards
- Cost
  - Time-to-market & NRE
- Performance
  - Power & speed
- Application demands are not excel or word, but multimedia!

R.C. Performance

- Increased compute density
  - Gates dedicated to computing rather than control logic
- Matched data word size
- Parallelism
  - Instruction level
  - Bit level
  - Pipeline

The Basic Idea

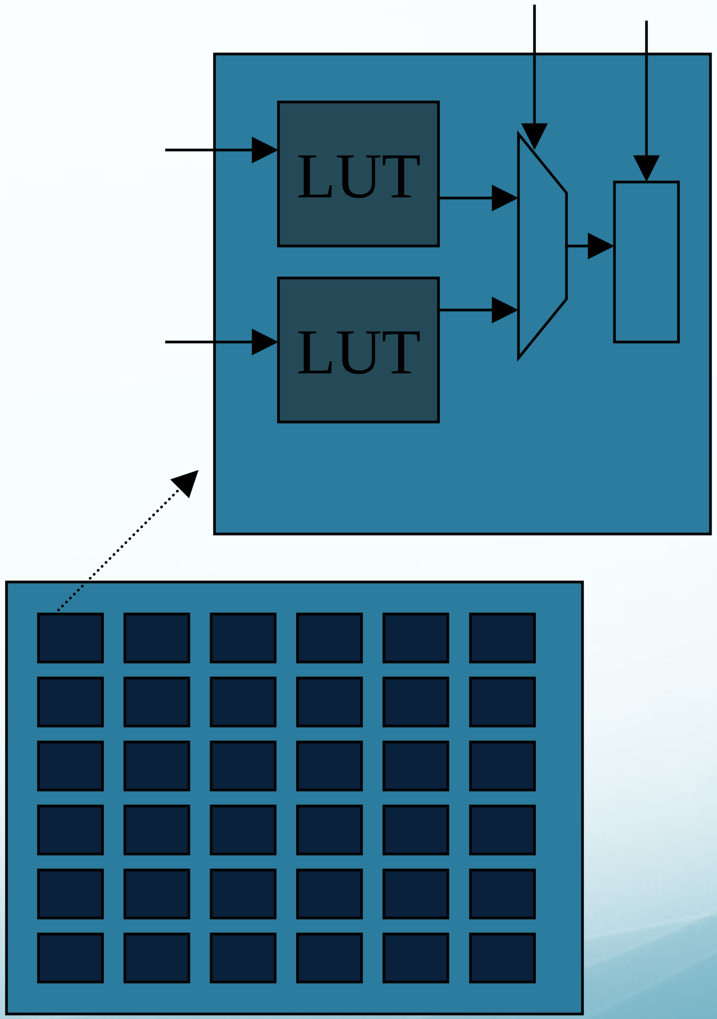


FPGA Basics

- Programmable logic blocks
- Programmable connections between logic blocks
- Mostly digital (but some mixed signal devices available)
- Bit stream programmable: SRAM, EEROM, Flash …
- As of 2010
  - Millions of gates (equivalent) capacity
  - Up-to 1GHz (maximum) clock speed
  - $3B market share
  - 100K design starts

FPGA Internals

- Look up tables (LUTs) form the basic logic generation
  - 3 to 4 inputs wide
  - Multiple LUTs / cell
- Programmable communication grid
- Recent trends:
  - Incorporate memory and processor on chip
- FPGA bit stream

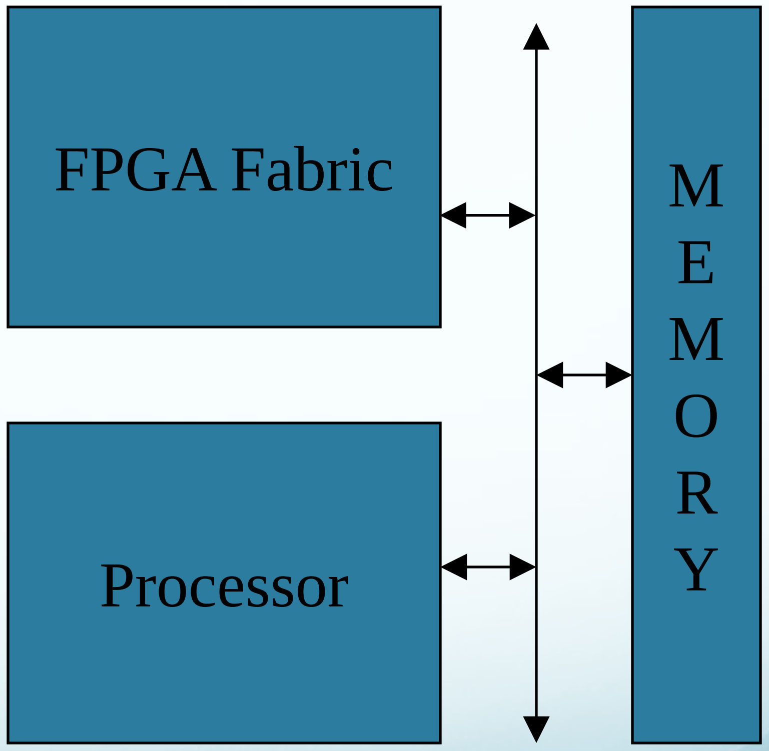


R.C. Model of Computation

- One or more FPGA contexts (bit streams) pre-synthesized B_0, B_1, B_2 … B_n
- Cost for reconfiguration $C_{reconfig}$
- FPGA holds K contexts at a time
  - *$K < n$: Dynamic scheduling*
  - *$K >= n$: Static scheduling*
- Notion of scheduling
  - Processing stops during reconfiguration
  - Next context being configured while current context is executing (double buffering)
  - NP complete problem (similar to partitioning/scheduling in codesign)

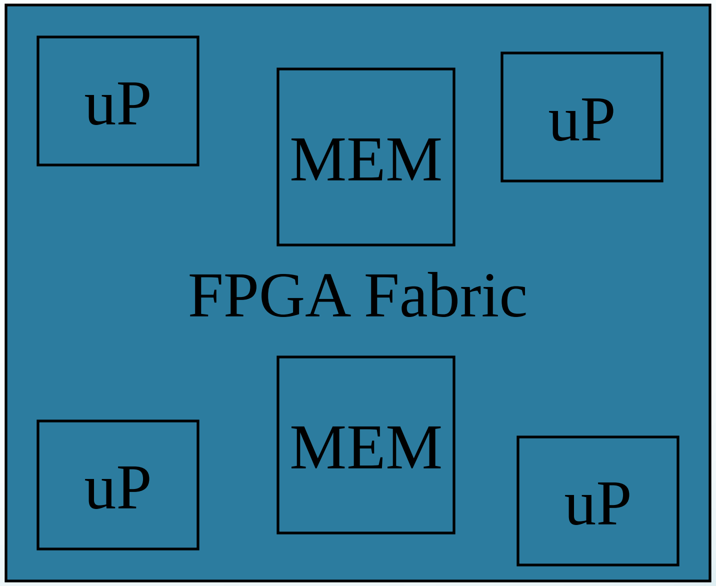
R.C. Basic Architecture

- Processor implements most functions
- FPGA fabric implements some compute intensive functions
- Shared memory (dual port or DMA)
- Shared bus
  - bottleneck



R.C. Extreme Architecture

- One or more processors on FPGA chip
- One or more memory blocks on FPGA chip
- FPGA fabric used for
  - Communication
  - Arbitration
  - Coherency
  - Compute



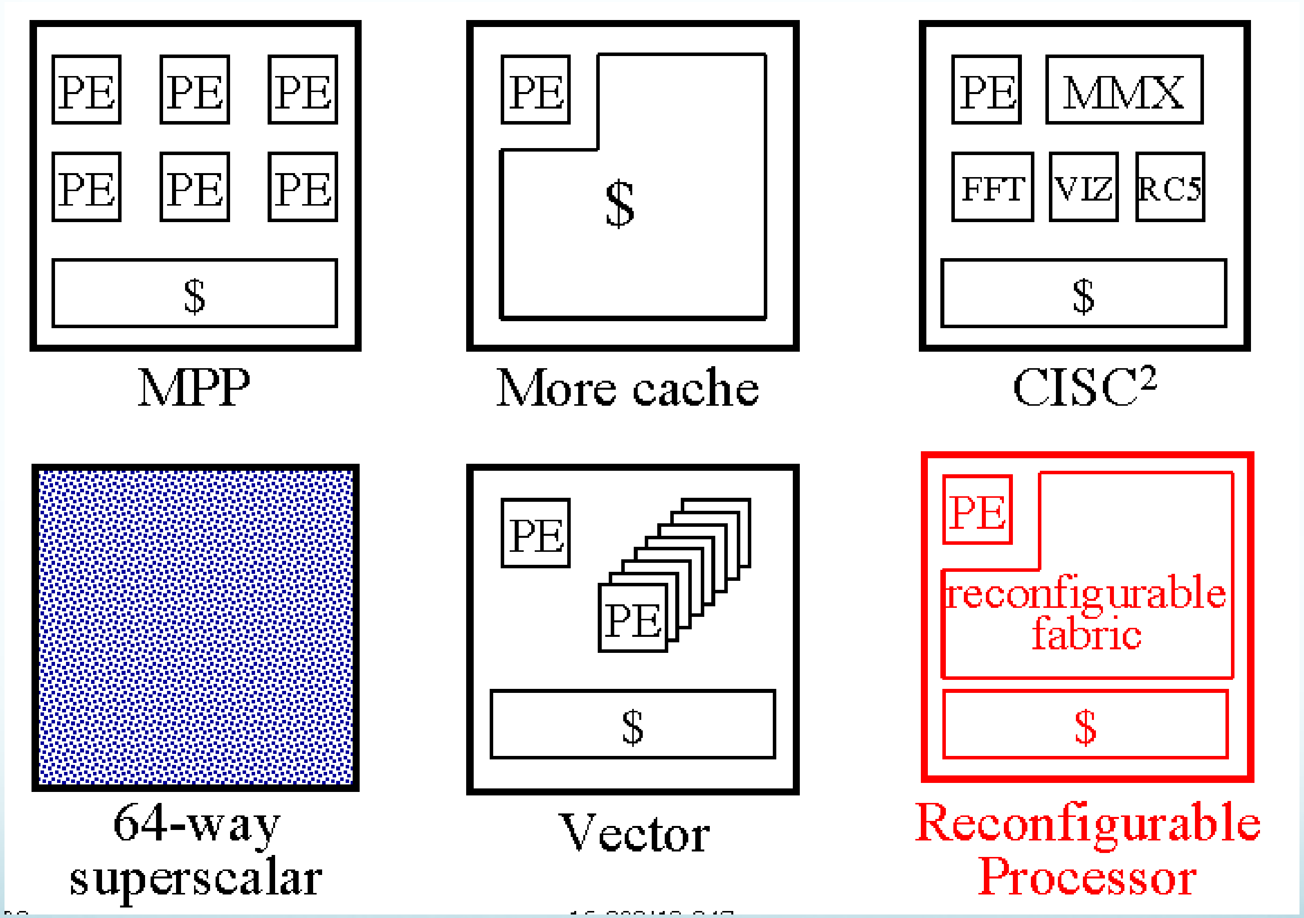
Coarse Grain Architectures

- A grid of programmable devices (mini-processors)
- A central CPU handling context scheduling and sequencing
- Devices
  - Perform arithmetic operations
  - Data steering & forwarding
- Highly specialized compilers
- Large hand-programmed kernels

Synthesis in the Loop R.C.

- The tools necessary to generate the FPGA context is included in the firmware
- Dynamic profiling of application discovers hot-spots
- Synthesis/compiler tool invoked to generate FPGA hardware
- Application code is patched to redirect execution to hardware
- Repeate

Summary


PE
$
MPP
PE
$
More cache
PE
MMX
FFT
VIZ
RC5
$
CISC²
64-way superscalar
PE
PE
$
Vector
PE
reconfigurable fabric
$
Reconfigurable Processor