

# Multi-Processing on Small FPGA's

## Introduction

A Bit of Background

Core Concept

Programmer's View

# Multi-Processing on Small FPGA's

## A Bit of Background

What is small?

Altera MAX 10© 10M02

2000 4 input luts

2000 FF's

12KB Memory

\$2.84 Quantity 1000

# Multi-Processing on Small FPGA's

## A Bit of Background

What kind of multi-processing?

14 8 Bit Processors  
333 MIPS Aggregate  
No Memory Conflicts

# Multi-Processing on Small FPGA's

## A Bit of Background

Speed is determined by time to access memory.

Memory can be read in the Max10©  
10M02 in 3 ns so an instruction can be  
read 333 million times per second giving  
333 MIPS.

# Multi-Processing on Small FPGA's

## A Bit of Background

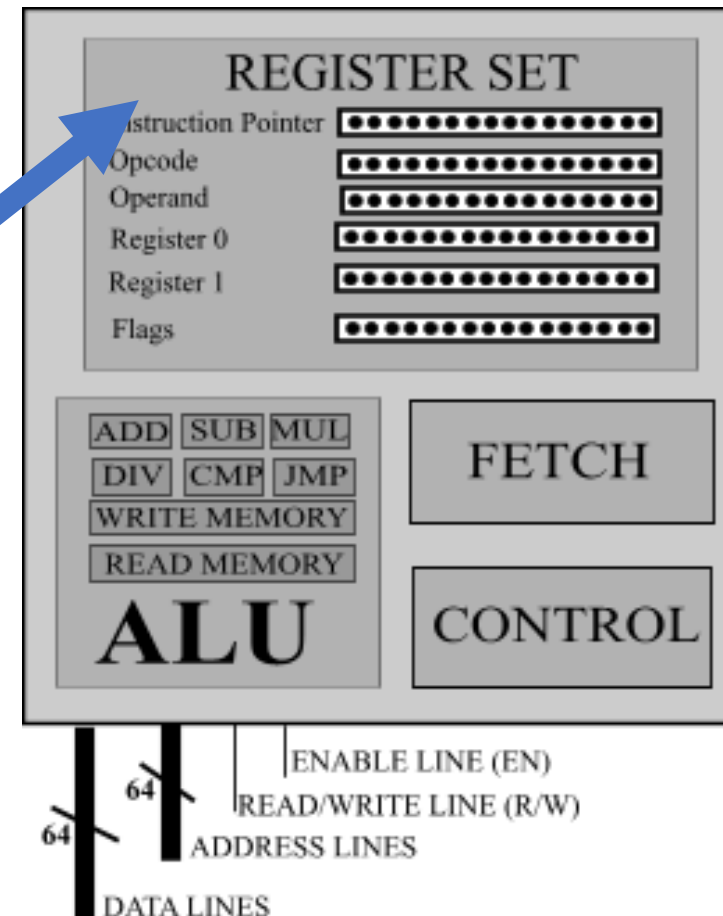
How?

SuperCore

# Multi-Processing on Small FPGA's

## Core Concept

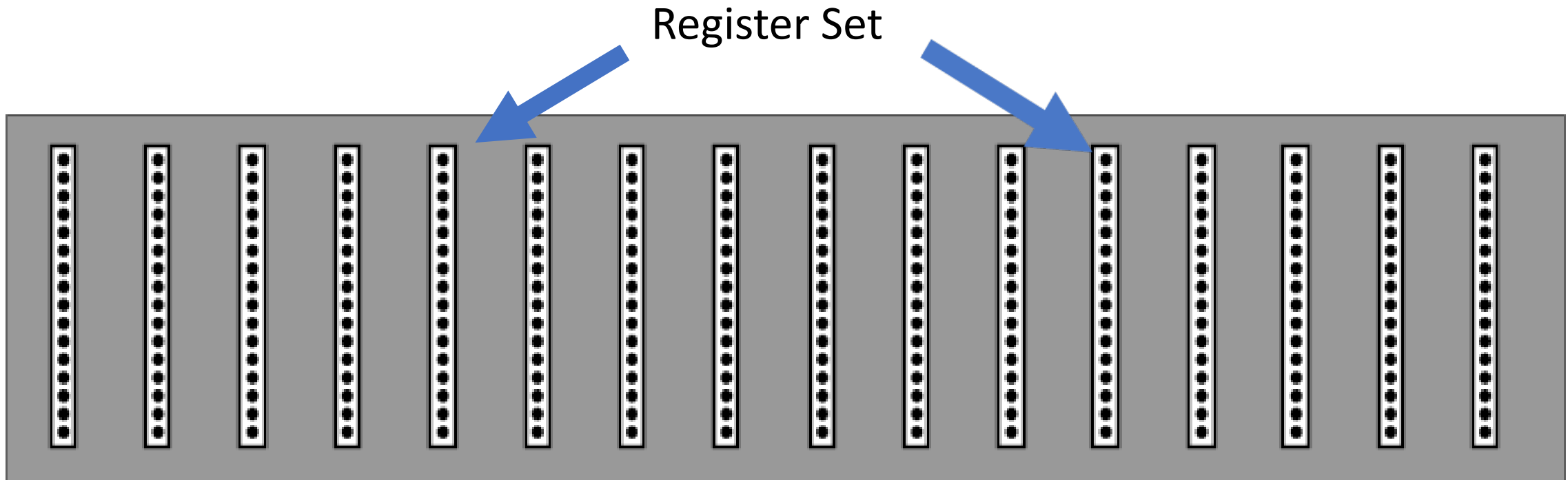
A core computer consists of a set of hardware that contains a set of registers and logic that manifest a single processor computer



# Multi-Processing on Small FPGA's

## Core Concept

A super core computer consists of a similar set of hardware that contains virtual sets of registers that can manifest from 1 to thousands of processors.



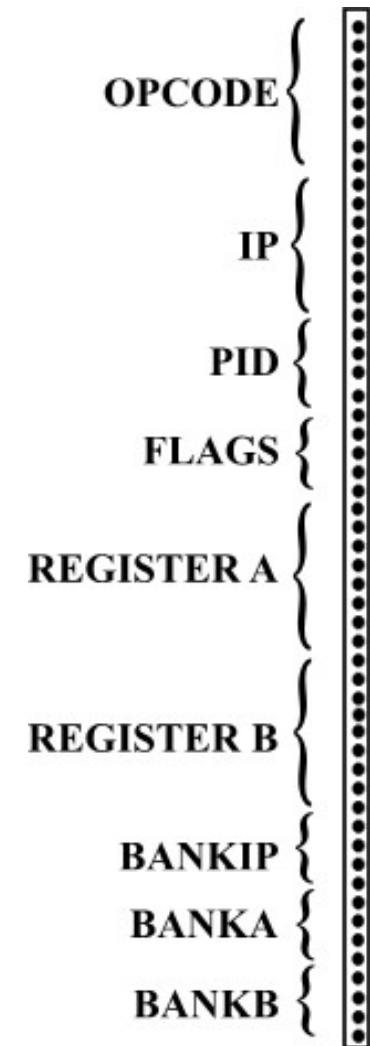
# Multi-Processing on Small FPGA's

## Core Concept

A super core register set.

This is the essence of a processor.

There are many of these in a super core.

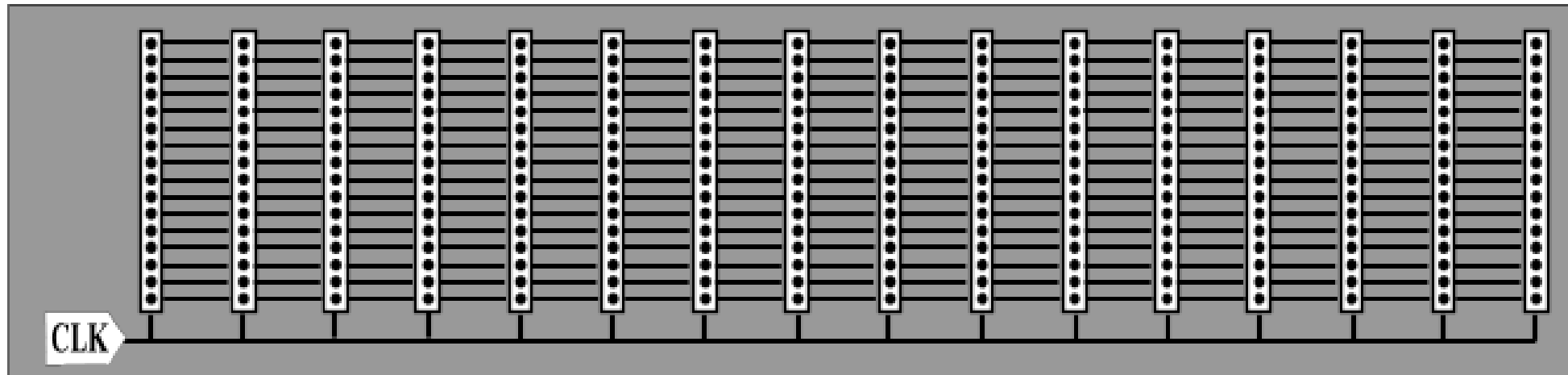




# Multi-Processing on Small FPGA's

## Core Concept

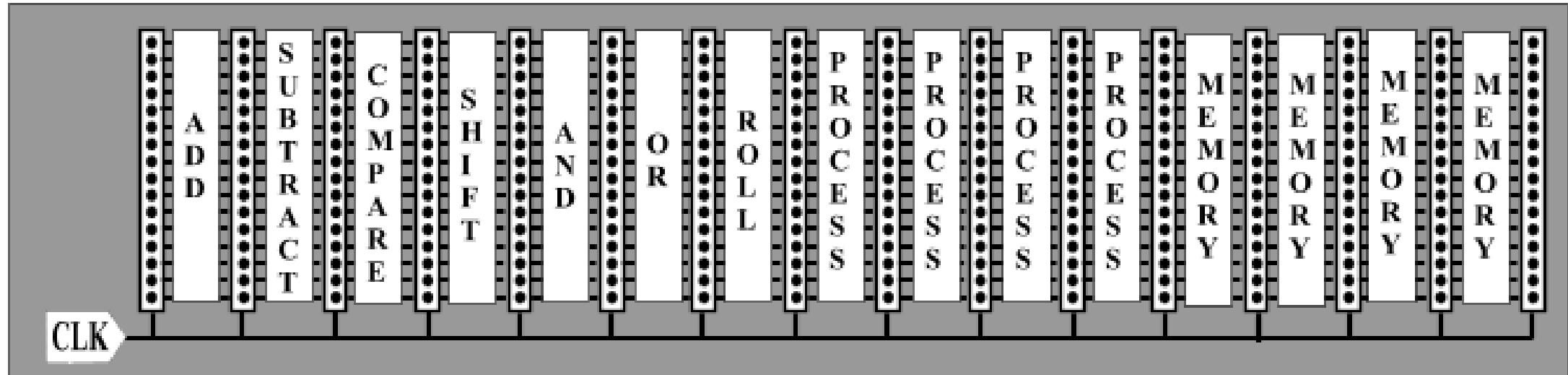
With each clock tick, the processor bits move from register to register.



# Multi-Processing on Small FPGA's

## Core Concept

As the processor bits move from stage to stage they encounter logic.

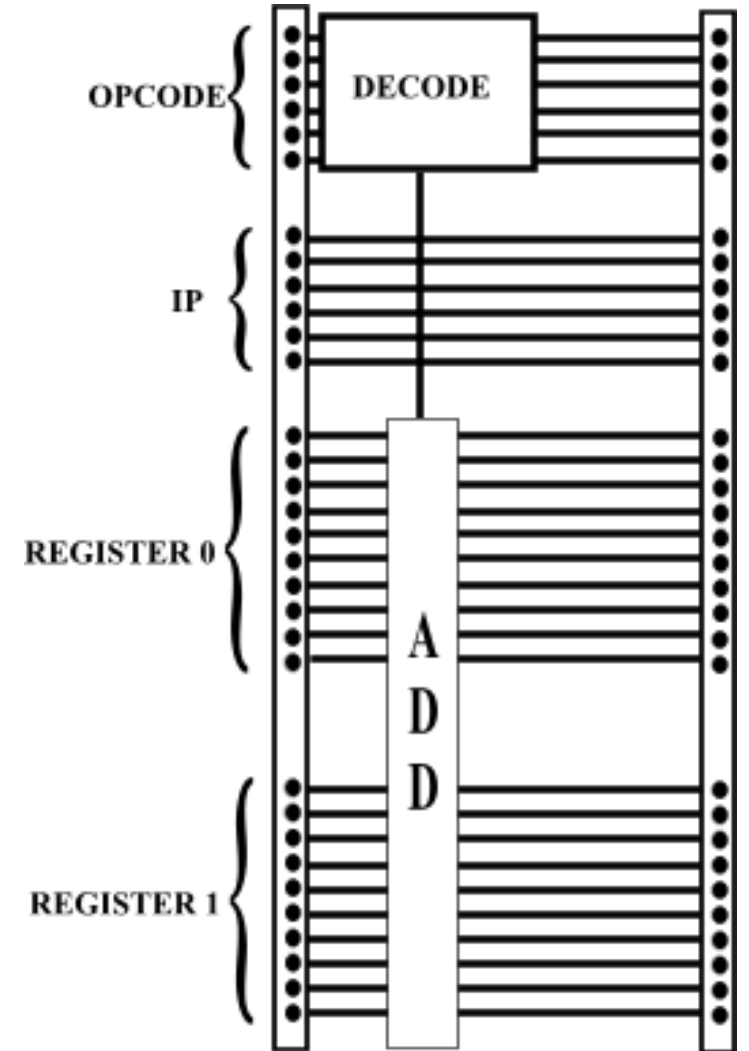


# Multi-Processing on Small FPGA's

## Core Concept

Each stage handles part of the processing task.

Decoding of all processor instructions happens at the same time.



# Multi-Processing on Small FPGA's Core Concept

That's It

On to Programmer's View

# Multi-Processing on Small FPGA's

## Programmer's View

Instruction structure

Always 4 bytes

OPCODE OPERANDA, OPERAND B, OPERAND C

# Multi-Processing on Small FPGA's

## Programmer's View

Note, operands are not registers but addresses in memory

The programmer is not aware of registers but manipulates memory directly

# Multi-Processing on Small FPGA's

## Programmer's View

As operands consist of 8 bits, there are bank instructions that enable each operand to extend memory range.

BANK BANK A, BANK#, NULL

ADD A, B, C

# Multi-Processing on Small FPGA's

## Programmer's View

A Jump is sensitive to Processor ID (PID).

```
JMPONPID (PID#, BANK#, A);
```



# Multi-Processing on Small FPGA's

## Programmer's View

Note, all processors access memory simultaneously.

So we have Mutex

GATECLOSE

GATEOPEN

# Multi-Processing on Small FPGA's

## Programmer's View

When execution begins.

The first 14 locations in memory would be  
JMP Some Address.

# Multi-Processing on Small FPGA's

## Programmer's View

The unique structure requires concepts that are a bit different from normal. Those shown represent a few of the different approaches.

# Multi-Processing on Small FPGA's

----- ***THE END*** -----

Thanks for watching.