A Microcontroller for Everyone

Maker Faire
May 21-22, 2011

Chen-Hanson Ting
Silicon Valley Forth Interest Group

Summary

- DIY microcontroller
- CPU hardware design in FPGA
- Operating system and programming language in Forth
- Demonstrations

DIY Microcontroller

- Lattice XP2 Brevia Kit is a complete FPGA development system for $49.
- The FPGA chip on Lattice Brevia Kit lets us design and implement your own microcontrollers.
- A 32-bit Forth microcontroller eP32 is demonstrated here.

Microcontroller for Everyone

- For $49, you can design, build and test your own microcontroller.
- The on-board FPGA chip, LatticeXP5E, can host a complete 32-bit microcontroller system.
- eP32 is implemented in VHDL, with Forth operating system and programming language.

Microcontroller in FPGA

- Brevia Kit has all hardware components for a microcontroller system.
- LatticeXP2-5E 6TN144C FPGA chip has enough gates and RAM/ROM memory to host a complete 32-bit microcontroller.
- Lattice provides free hardware design tools.
- eForth provides an operating system and Forth programming language.

LatticeXP2 Brevia Kit

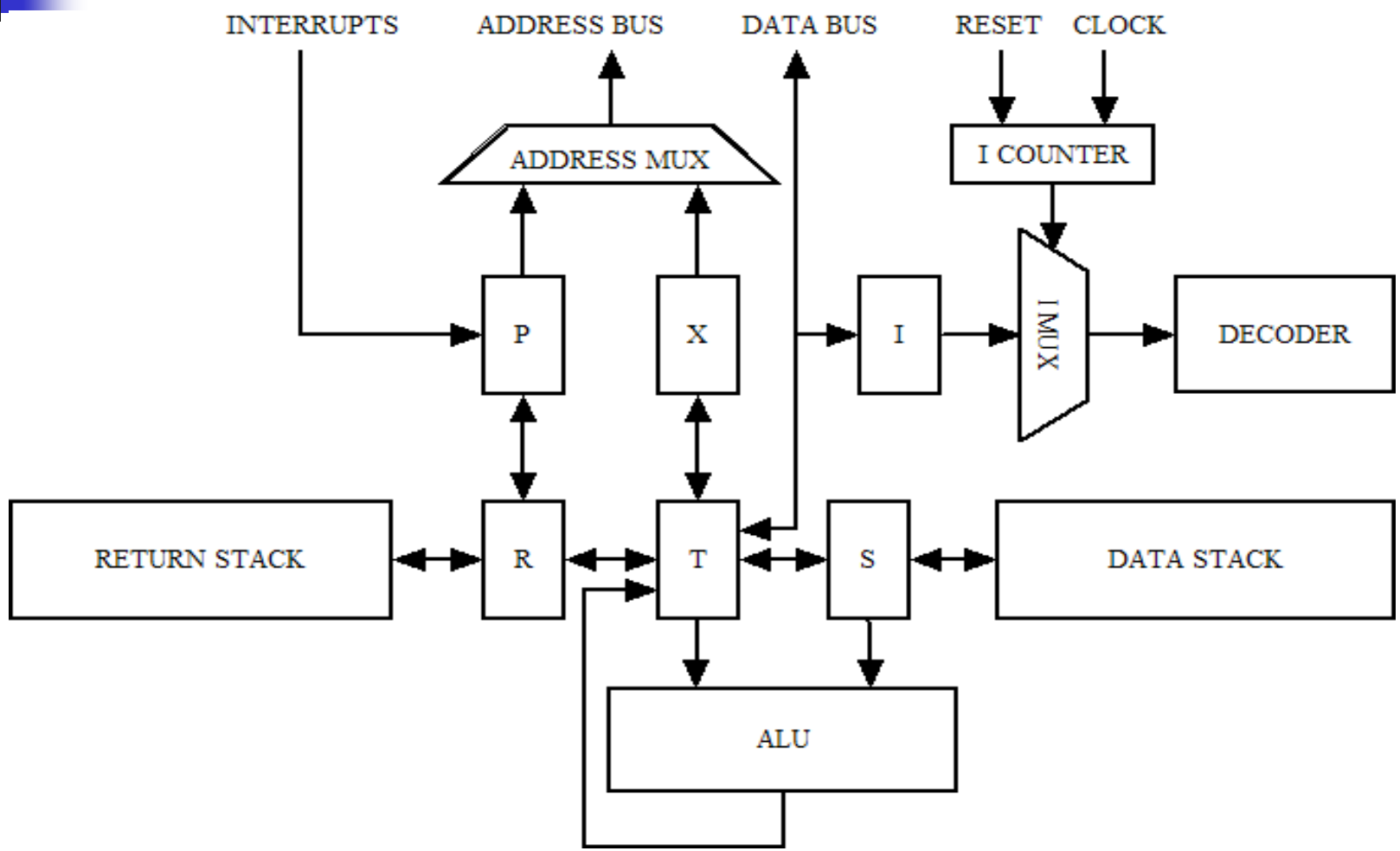
Free Hardware Design Tools

- Synthesis: Synplify
- Simulation: Active-HDL from Aldec
- Layout: Design Planner
- Flash Programming: ispVMR
- Tracing and Debugging: ispReveal

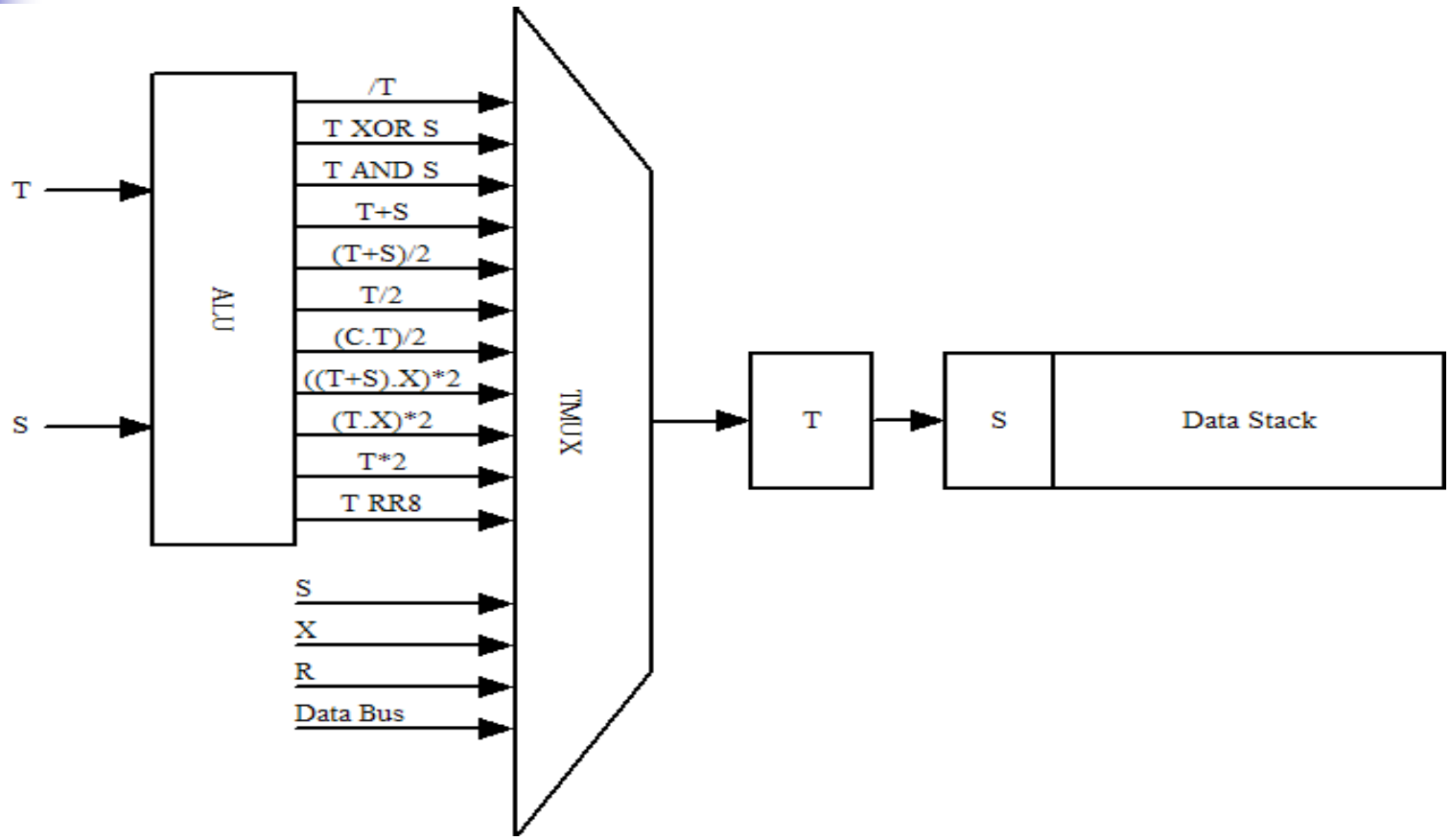
Architecture of eP32 CPU

- A 32-bit CPU (Central Processing Unit)
- Two stacks to support Forth Language:
  - Return stack for nested return addresses
  - Parameter stack to pass parameters among nested subroutines
- Minimal instruction set
- Single clock cycle execution time
- Optimized subroutine call and return

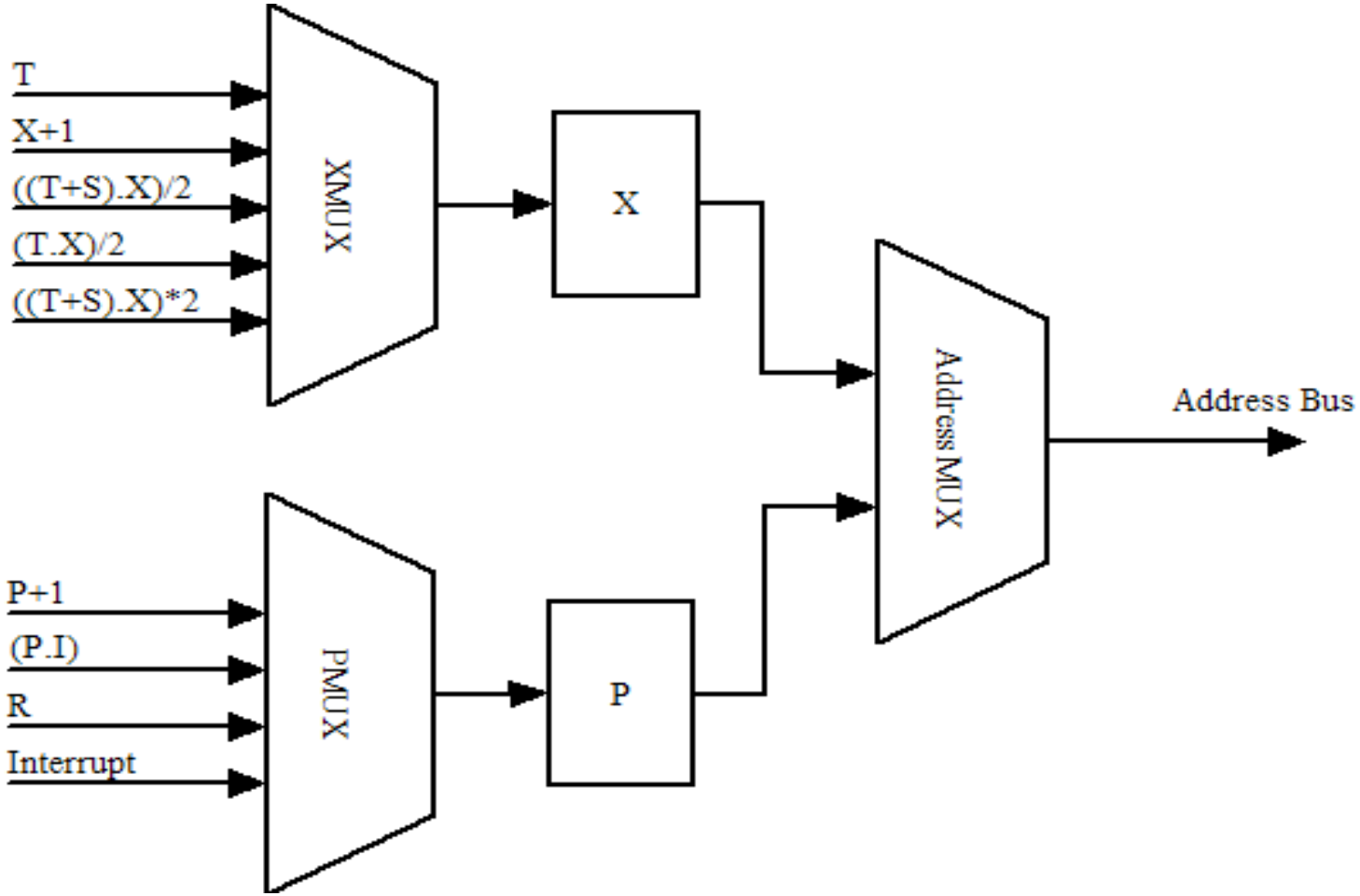
Block Diagram of eP32 CPU



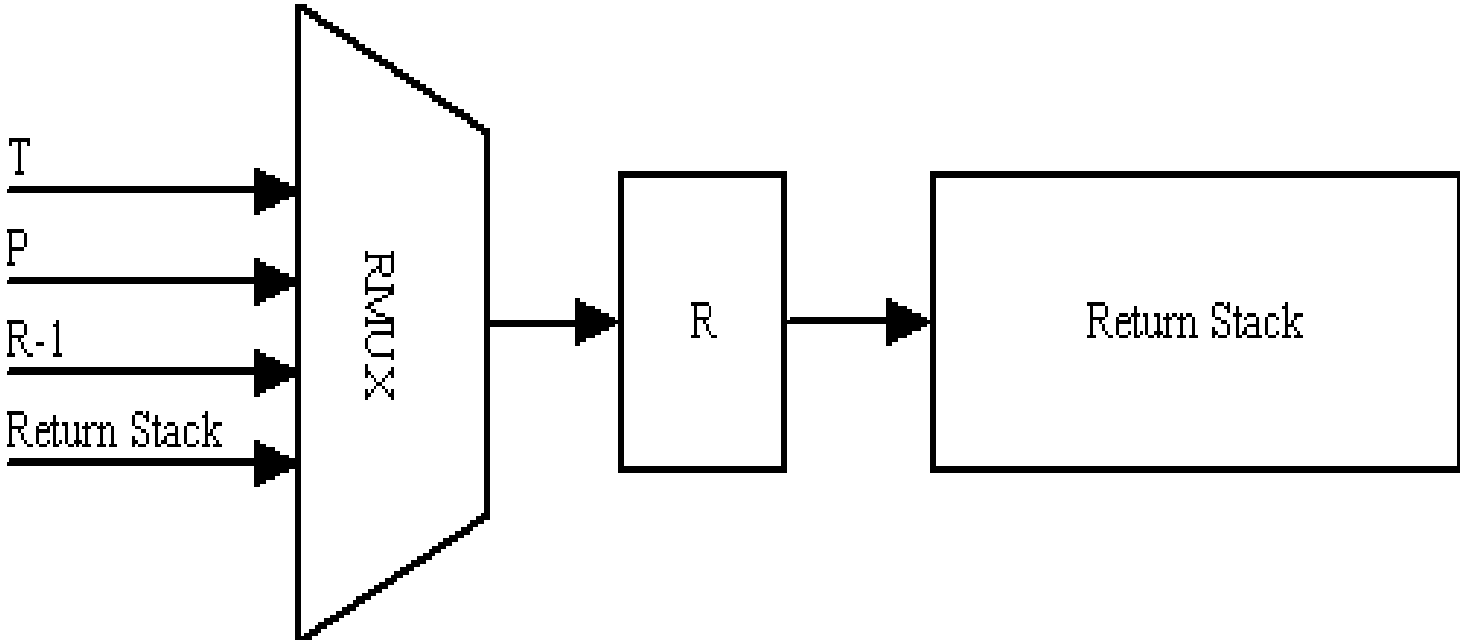
ALU and Data Processing Unit



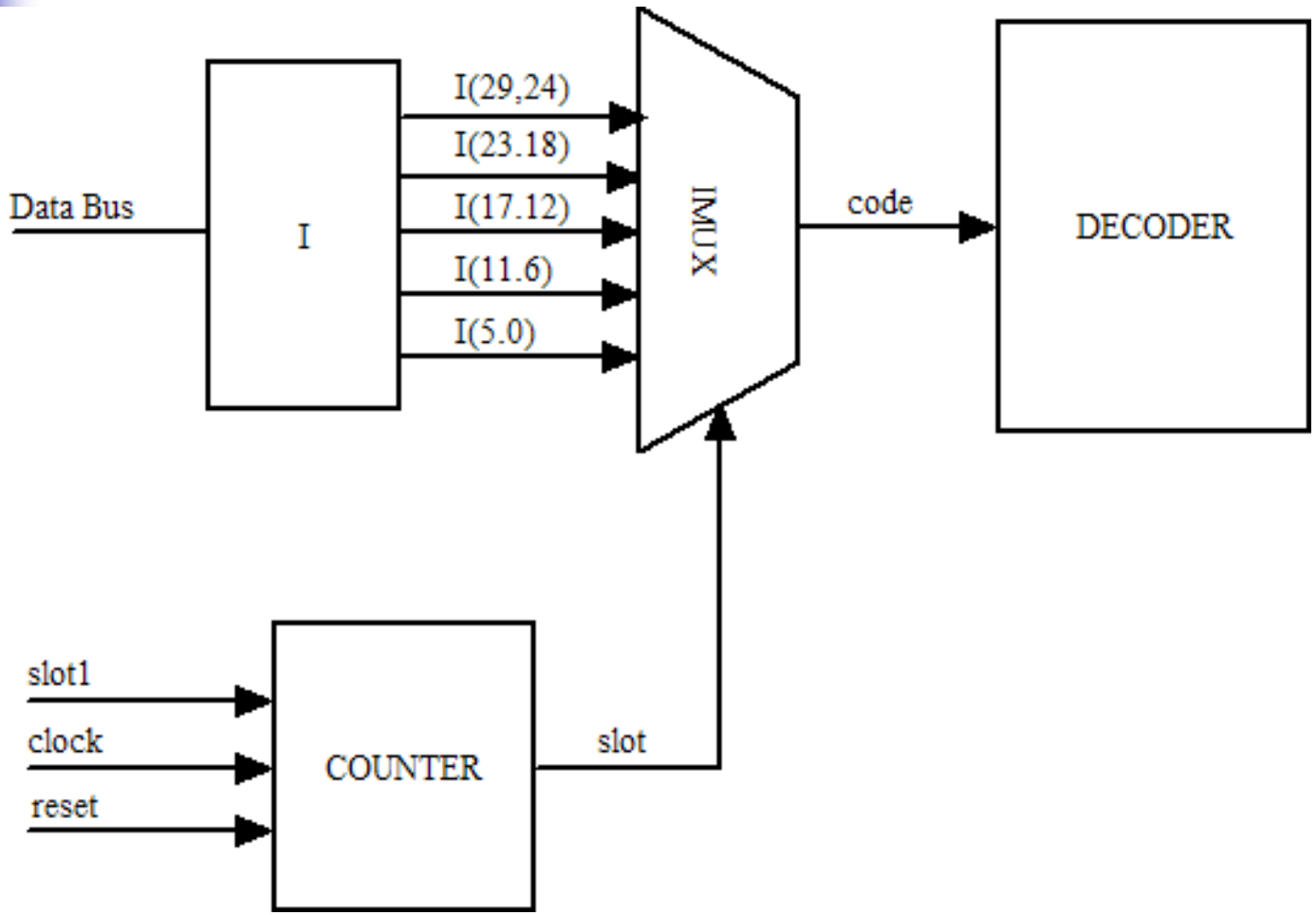
Program and Data Memory Unit



Return Address Processing Unit

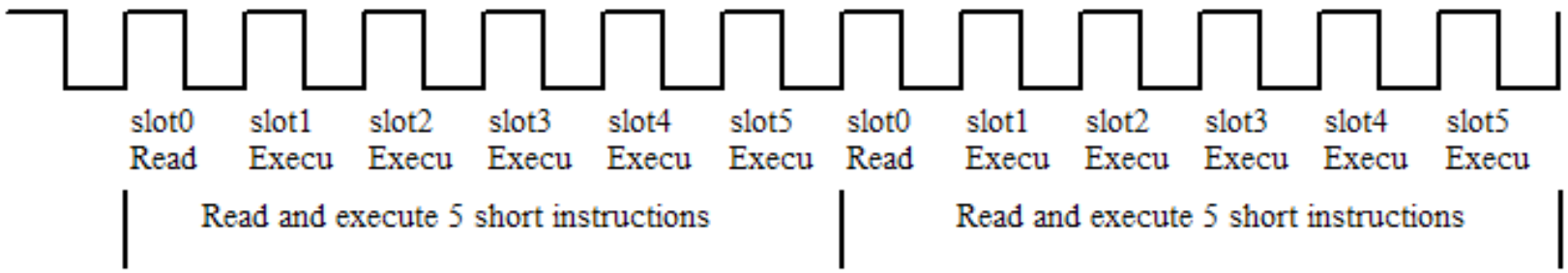


Instruction Execution Unit

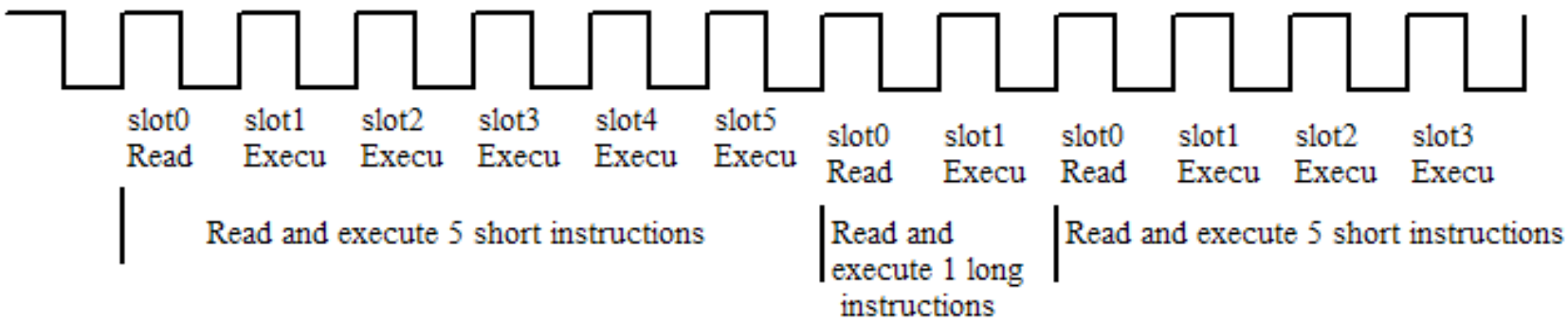


Instruction Execution Timing

Execution Cycles of Short Instructions



Execution Cycles of Long Instructions



Instruction Set of eP32 CPU

- Minimum Instruction Set Computer (MISC)
- Only 27 Instructions, expandable to 64 instructions
- 4 Types of instructions:
  - Program transfer instructions
  - Memory access instructions
  - ALU instructions
  - Register and stack instructions

Program Transfer Instructions

- BRA Branch always
- RET Return from subroutine
- BZ Branch on zero
- BC Branch on carry
- CALL Call subroutine
- NEXT Loop until R is 0

Memory Access Instructions

- LDX — Load from memory
- LDXP — Load from memory and increment X register
- LDI — Load immediate value
- STX — Store to memory
- STXP — Store to memory and increment X register

ALU Instructions

- ADD Add S to T
- AND AND S to T
- XOR XOR S to T
- COM One's Complement of T
- SHR Shift T to right
- SHL Shift T to left
- RR8 Rotate T right by 8 bits
- MUL Multiplication step
- DIV Division step

Register and Stack Instructions

- DUP Duplicate T to S
- DROP Pop S to T
- PUSH Push T to R
- POP Pop R to T
- OVER Duplicate S over T
- TX Load X to T
- XT Store T to X
- NOP No operation

eP32 Microcontroller in VHDL

- eP32_chip.vhd, top level design
- eP32q_tb.vhd, test bench
- Modules
  - eP32.vhd, CPU core
  - Ram_memory.vhd, 4096x32 auto-initialized RAM memory
  - Uart.vhd, 115200 baud
  - Gpio.vhd, 16 bit bidirectional

Synthesizing eP32

- Ep32.vhd, uart.vhd and gpio.vhd are synthesized without modification
- Ram_memory.vhd must be constructed to use RAM_Q modules in LatticeXP2-5E FPGA chip.
- Ep32_chip.vhd instantiates new ram_memory.vhd with all other modules

ispLEVER Project Navigator - [D:\ispTool\...\project\ep32q_xp2_4\ep32_xp2.syn]

File View Source Process Options Tools Window Help

Sources in Project:

- LFXP2-5E-5TN144C
  - ep32q_tb.vhd
  - **[ep32_chip (ep32_chip.vhd)]**
    - ep32 (ep32.vhd)
    - uart (uart.vhd)
    - ram_memory (ram_memory.vhd)
    - gpio (gpio.vhd)

Modules | Files

Processes for current source:

- Build Database
  - Build Database Report - HTML (ep32_xp2.html)
  - Build Database Report (ep32_xp2.drp)
  - Design Planner (Pre-Map)
  - Edit Preferences (ASCII)
- Map Design
  - Map Report - HTML (ep32_xp2.html)
  - Map Report (ep32_xp2.mrp)
  - Map TRACE Report (ep32_xp2.tw1)
  - Design Planner (Post-Map)
- Map Timing Checkpoint
- Place & Route Design
  - Report Summary - HTML (ep32_xp2.html)
  - Place & Route Report (ep32_xp2.par)
  - PAD Specification File (ep32_xp2.pad)
  - Place & Route TRACE Report (ep32_xp2.twr)
  - I/O Timing Report (ep32_xp2.ior)
  - I/O SSO Analysis Report (ep32_xp2.sso)
  - IBIS Model
  - Reentrant Route Design
  - Memory Initialization

```
-- Restoring VHDL parse-tree ieee.std_logic_arith from d:/isptool/cae_library/vhdl_pa
-- Restoring VHDL parse-tree ieee.std_logic_misc from d:/isptool/cae_library/vhdl_pac
-- Restoring VHDL parse-tree synopsys.attributes from d:/isptool/cae_library/vhdl_pac
-- Restoring VHDL parse-tree ieee.std_logic_unsigned from d:/isptool/cae_library/vhdl
-- Analyzing VHDL file ram_memory.vhd
-- Analyzing VHDL file uart.vhd
-- Analyzing VHDL file ep32.vhd
-- Analyzing VHDL file ep32_chip.vhd
-- Elaborating ep32_chip
Done: completed successfully.
```

Automake Log

Ready | Pure VHDL | Auto Read | Symplify

Simulating eP32

- Active-HDL simulation tools are supplied by Aldec.
- It needs a test bench module for functional simulation of eP32 chip:

  ep32q_tb.vhd

Active-HDL 8.2 (work ,work) - D:\ispTool\Demo_LatticeXP2_Brevia_SoC_VHDL\Demo_LatticeXP2_Brevia_SoC\pr...

File Edit Search View Workspace Design Simulation Waveform Tools Window Help

Design Brows...

testbench (behavior)

testbench (beha
uut : ep32_chi
cpu1 : ep3
line__
line__
line__
line__
line__

Name
s_stack(4)
s_stack(3)
s_stack(2)
s_stack(1)
s_stack(0)
r_stack
slot

F... St... R...

Name	V	S	20	40	60	80	100	120	140	160	180	200	220	240	260 ns
sclk	1														
aclk	1														
arst	1														
interrupt_i	..		UU												
ioport	..		ZZZZ												
uart_i	1														
uart_o	1														
addr	..		00000000			00000001		000006BD		000006BB		00000640		00000641	
data_i	..			000006BD			00000000		04000640		04000332		0A29B79B		000
code	..		1B			00		1B		04		1B		0A	

ep32_chip... ep32.vhd waveform ...

Console

```
# KERNEL: Time: 0 ps,  Iteration: 2,  Instance: /uut/cpu1,  Process: line__133.
# KERNEL: WARNING: There is an 'U'|'X'|'W'|'Z'|'-' in an arithmetic operand, the result will be 'X'(es).
# KERNEL: Time: 0 ps,  Iteration: 2,  Instance: /uut,  Process: line__197.
# KERNEL: WARNING: There is an 'U'|'X'|'W'|'Z'|'-' in an arithmetic operand, the result will be 'X'(es).
# KERNEL: Time: 0 ps,  Iteration: 2,  Instance: /uut,  Process: line__197.
```

Console

Layout of eP32

- Design Planner to assign following signals to physical pins:
  - External reset
  - External master clock
  - Interrupts
  - GPIO to LED and switches
  - UART transmit
  - UART receiver

Package View (ep32_xp2.lpf)

File Edit View Tools Help

Project: ep32_chip
- Design Signals
  - aclk @ 21
  - arst @ 19
  - interrupt_i(4:0)
    - interrupt_i_0 @ 58
    - interrupt_i_1 @ 57
    - interrupt_i_2 @ 56
    - interrupt_i_3 @ 55
    - interrupt_i_4 @ 54
  - ioport(15:0)
    - ioport_0 @ 46
    - ioport_1 @ 45
    - ioport_2 @ 44
    - ioport_3 @ 43
    - ioport_4 @ 40
    - ioport_5 @ 39
    - ioport_6 @ 38
    - ioport_7 @ 37
    - ioport_8 @ 53
    - ioport_9 @ 52
    - ioport_10 @ 50
    - ioport_11 @ 1
    - ioport_12 @ 2
    - ioport_13 @ 5
    - ioport_14 @ 6
    - ioport_15 @ 7
  - uart_i @ 110
  - uart_o @ 109
- Device: LFXP2-5E-TQFP144

(Bottom View)

Package Layout

Site 60:VCCAUX; Bank 99: 0 assigned/27 total

Programming XP25E FPGA

- Connect JTAG cable to printer port.
- Connect UART cable to COM port.
- Invoke ispVMR system to download ep32_xp2.jed file.
- eP32 eForth boots up and sends sign-on message to Hyperterminal console.

Software Development Tools

- A metacompiler written in Forth includes:
  - eP32 machine code assembler
  - A nucleus of eP32 commands
  - Text interpreter
  - Compiler for new command
  - Debugging tools
  - eP32 software Simulator

eForth System for eP32

- eForth is a complete Forth operating system produced by the metacompiler.
- Binary memory image is produced to be synthesized with eP32 CPU.
- Application software can be written, tested, and debugged under eForth.

Demonstration

- Boot up eForth on eP32
- Universal greeting “Hello, World!”
- Turn LED’s on and off
- Read switches and push buttoms

Demonstrations

115200 - 超級終端機

檔案(F) 編輯(E) 檢視(V) 呼叫(C) 轉送(T) 說明(H)

```
 OK
  OK

eP32q v2.05
 : TEST1 CR ." HELLO, WORLD!" ; OK
  OK
  OK
 TEST1
HELLO, WORLD! OK
  OK
  OK
 HEX OK
 FF E0000001 ! OK
  OK
 0 E0000000 ! OK
  OK
 FE E0000000 ! OK
  OK
 E0000002 ? FFFE OK
  OK
 E0000002 ? FDFE OK
  OK
  OK
```

連線 00:05:24 | VT100 | 115200 8-N-1 | SCROLL | CAPS | NUM | 擷 | 列印

eP32_xp2 Release

- All VHDL design files
- All eForth metacompiler files
- weForth system, an eForth implementation for Windows
- Documentation in ep32_xp2.pdf
- Available at http://www.offete.com for $25

Take Home Messages

- We can break the yokes of Intel and Microsoft.
- We can explore the complete hardware/software space for the best solutions to our computing needs.

Forth Organizations

- Forth Interest Group http://www.forth.org

  A repository of public domain Forth systems and Forth documentation
- FORTH, Inc. http://www.forth.com

  SwiftForth, SwiftX for many microcontrollers
- MicroProcessor Engineering, Ltd. http://www.mpeforth.com

  VFX Forth, Forth 7 Cross Compiler

SVFIG -- Silicon Valley Forth Interest Group

- SVFIG meets every month at Stanford University to discuss the Forth language and its applications.
- For meeting announcements, please check:

  http://www.forth.org/svfig/next.html
  http://www.meetup.com/SV-FIG/

Thank You.