Reincarnate Historic Systems On FPGA with Novel Design Methodology

Naohiko Shimizu, Ph.D.
IP ARCH, Inc. / Tokai University
You may have questions.

- **Why** Reincarnate Historic Systems?
- **What is** Reincarnated Historic Systems?
- **How to** Reincarnate Historic Systems?
Preface (2/4)

• Why?
  – Good for education
    • Copying after good design affords good design discipline.
    • Open software but hardware...
    • Not too complex.
    • Students can examine every detail of design.

  – Fun!
    • It is like a puzzle to make them by ourselves with current technology.
Preface (3/4)

• What is?
  – Not a Clone!
  – Run the same software
    • Same instruction set architecture
    • Software compatible I/O subsystems
  – Renew the hardware design
    • Design from the specification.
    • Substitute new technology
      – FPGA for TTL/MPU
      – Digital processing for analog parts
      – Conversion for obsolete I/O
Preface (4/4)

• How to?
  – Select good system
    • Happy to manipulate
    • Specification available
    • Copyright cleared
    • patents expired
  – Collect software and specs.
  – Make CPU and I/O subsystems
  – Write BIOS or required software
  – Run
    • on simulation and on FPGA
to get sense of achievement
Agenda

• Our past decade projects
  – Case 1: Space Invader compatible
  – Case 2: PDP11 compatible with UNIX V6
  – Projects for 2009

• Design methodology

• Conclusion
Our past projects

• 2000 CP/M80 system with SSD
• 2002 Space Invaders compatible
• 2002 PDP11 compatible with UNIX v6
• 2003 i8086 compatible with FreeDOS
• 2004 VAX11 compatible without MMU
• 2006 DLX and GCC
• 2007 UML based 6502 CPU design
• 2009 UML based MIPS CPU
Case 1: Space Invaders

• Why?
  – Fun!
  – Famous!

• What is?
  – Video game born in 1978
  – i8080A CPU, NTSC video,
    Analog sound, colored tape

• How to?
  – In the next page
How to? : Space Invaders

- Assigned 2 undergraduate students.

  CPU debug
  I/O, timer
  NTSC video
  audio

Student
Shimizu
8080A CPU

Negotiate with TAITO for use of the program
How to? : Space Invaders

- Substitute digital for Analog circuits
  - Color tape to color NTSC signal
How to?: Space Invaders

• Substitute digital for Analog circuits
  – Op amp and Sound chip to FPGA
How to? : Space Invaders

• Make a piggy back PCB and PLAY.
Space Invaders
Case 2: pdp11 and UNIX

• Why?
  – Handy size REALLY USED OS.
  – Lions' book
  – Origin of C language

• What is?
  – Maybe it is obvious for people here.
  – License of ancient UNIX is set open by Caldera in 2002.
  – Without MMU 11/10, with MMU 11/40m
How to?: pdp11 and UNIX

- Assigned 1 undergraduate student.

**Student**
- Shimizu
- CPU wo MMU
- MMU / IO
- PCB
- dhrystone demo
- PCB
- 6502 sample
- Hex monitor
- Lions text lecture
- GCC/GAS
- RTOS

**Time**
How to?: pdp11 and UNIX

• pdp11/10: No MMU CPU
  – Fix GCC/binutil pdp11 port
  – wrote hex decimal monitor
  – on chip RAM as main memory
  – Port RTOS proc Real-Time Kernel by nilsen elektronikk as.
  • Run multi-task applications
How to? : pdp11 and UNIX

- pdp11/40m Lions' book CPU
  - I/O subsystem substitute
    IDE HDD for RK storage
  - PC for serial console
  - on chip Boot loader
How to? : pdp11 and UNIX

• Make a PCB to use with FPGA board.
  – Level converter
  – SRAM, Clock module
PDP11 Compatible CPU

7th semester

Student C
CPU Specification/Logic/Simulation GCC/RTOS porous MMU/UNIX

8th semester

POP-11/10 for Embedded Debug on FPGA POP-11/40*

Demonstration at Embedded Technology 2002 Parthenon Society, ASIC Design Contest

UNIX V6 boot and work

<table>
<thead>
<tr>
<th>CPU</th>
<th>Logic Cells</th>
<th>Max Freq.</th>
<th>FPGA</th>
</tr>
</thead>
<tbody>
<tr>
<td>POP-11/10</td>
<td>1678 LCs</td>
<td>9 MHz</td>
<td>EPF10K30EPC208-3</td>
</tr>
<tr>
<td>POP-11/40*</td>
<td>2687 LCs</td>
<td>20 MHz</td>
<td>EP1K100QC208-1</td>
</tr>
</tbody>
</table>
2009 Project
MIPS CPU from UML

• UML diagram to make the skeleton of CPU
Design Methodology

- **1996-2002**
  - HDL: SFL
  - EDA tool: PARTHENON
  - Much easier for students to make complex systems than VHDL or ...*

- **2003-2008**
  - HDL: SFL, my tool convert to Verilog
  - EDA tool: sfl2vl / Icarus Verilog

- **2009**
  - UML to SFL/NSL generator
  - HDL: NSL, successor of SFL, I designed.

  UML
Example of NSL

- UART output module core

```plaintext
function start seq {
  reg i[4], td[8];

  {td:=di; ack();}
  put_so(0b0); //start bit
  for(i:=0x0;i<0x8;i++) {
    put_so(td[i]);
  }
  put_so(0b1); // stop bit
  fin();
}

proc put_so {
  so = r;
  if(sck) finish;
}
```
LSI Design Flow with sfl2vl

Architecture level modeling

UML

SystemC prototype

SFL/NSL

sfl2vl

UMLtoNSL

SystemC Verification codes

Verification

Verification

SystemC

Verilog

ASIC

FPGA

Auto generate

Semi auto generate + manual entry

manual entry?
Conclusion

• Reincarnate historical systems on FPGA
  – some of examples explained
  – More examples in the proceeding

• New Design methodology with UML
  – Please read my paper for detail.
How to try now?

- I have USB memory includes
  - Logic and simulation codes of
    - PDP11 compatible with UNIX V6
    - 6502 compatible with Apple-I monitor
    - i8086 compatible with a small test code, etc.
  - Free EDA tools to compile/run
  - Cygwin based Execution environment
- You can copy/extract and run.
AISOC: Apple-I compatible

- 6502 PC with Woz's monitor.
  - Up to 38MHz with Spartan-3. 1208LUT.