Linux-Ready RV-GC AndesCore™ with Architecture Extensions

Charlie Su, Ph.D.
CTO and SVP
2018/05/09
Introduction to Andes

- Asia-based IPO Company
- 13 years in the pure-play CPU IP business
- Before Andes adopted RISC-V technologies:
  - AndeStar™ V1-V3 architecture
  - 10 active AndesCore™ (V3): 2-8 stage pipeline, 1- and 2-issue
  - Upstreamed (NDS32) GNU tools, OpenOCD, U-Boot, Linux, etc.
  - >140 licensees, >2.5B Andes-Embedded SoCs
- Rich experience in customer diversified needs:
  - Interrupt sources: who needs >16? 2-stage core needing >100.
  - Interrupt latencies: Some never ask; others care very much.
  - Efficiency: DSP+SIMD based on existing integer resources (i.e. GPR+MUL)
  - Performance: scalable vector (or SIMD) with dedicated resources
  - Need caches to be write-back and write-through
  - Loading RO-data from icache!
  - Some think GNU/LLVM the most popular; others think otherwise
Andes Approach to RISC-V

- **RISC-V has a good start**
  - Concise (RV-I), modular (RV-MACFD and more)
- **But, one size doesn’t fit all**
  - Extensible: leave more space for custom extensions
  - Architecture profiles are the right way to go
- **AndeStar V5 architecture:**
  - Adopt RISC-V as its subset
  - Add pre-defined useful Andes extensions
  - Provide custom-extension frameworks for DSA (or ASIP)
  - Push “features” important for compatibility thru Task Groups
- **Goal: Taking RISC-V Mainstream**
  - X86 for PC/servers
  - ARM for smart phones/tablets
  - RISC-V for all computing devices!
AndeStar V5 Extensions

- **Baseline extension instructions:**
  - Path length reduction with similar computation complexity
  - E.g. Memory accesses and branches with fewer instructions
  - Code size reduction on top of C-extension (CoDense™)

- **DSP/SIMD based on GPR → P-extension proposal**

- **Custom instructions (more later)**
  - Powerful tool simplifying new instruction creation
  - For SoC architects/designers, with or without CPU background

- **Non-instruction extensions: CSR-based**
  - Vectored PLIC with priority preemption → Fast interrupts proposal
  - Stack protection mechanism (StackSafe™)
  - Power management (PowerBrake)
  - Cache management in finer granularity
  - Simultaneous support for write-back and write-thru
Baseline V5 AndesCore™: N25/NX25

- Fast-n-small cores for control tasks in storage, networking, AI, and more.
- **N25**: 32-bit, **NX25**: 64-bit
  - From scratch for the best PPA
- **AndeStar V5m ISA**
  - Superset of RV-IMAC
- **5-stage pipeline**
- **Configurable multiplier**
  - Sequential or parallel
- **Optional branch prediction**
- **Flexible memory subsystem**
  - I/D Local Memory (LM): to 16MB
  - I/D caches: to 64KB
  - Optional parity or ECC
- **Bus interface**
- **JTAG debug module**

- **Sample N25 config. @ 28HPC:**
  - Small: 37K gates, 1GHz (worst)
  - Large: 159K gates, 1.15GHz (worst)
New V5 AndesCores Coming This Summer

- **4 New 25-series**: maintain the similar frequency
  - **N25F/NX25F**: N25/NX25 + FP support (RV-GC and more)
  - **A25/AX25**: N25F/NX25F + MMU + S-mode

- **High-performance FP support**:  
  - **RV-FD**  
  - Multiply, add/sub, multiply-accumulate:  
    - 1-cycle issue rate, 4-cycle latency  
  - Divide/sqrt: 15 cycles for SP, 29 cycles for DP  
    - Run in the background  
  - **Half-precision** load/store for machine learning

- **MMU support**:  
  - Supporting SV\{32, 39, 48\}  
  - Page sizes: \{Kilo, Mega, Giga, Tera\} page  
  - 4- or 8-entry microTLBs (ITLB,DTLB)  
  - 4-way 32~128-entry Shared-TLB (STLB)
Performance Efficiency For Low Power

- **Efficient pipeline:**
  - General performance: >20% higher (e.g. 3.49 CM/MHz)
  - Hit-under-miss caches: optimize performance with minor additional logic
    - Continue execution when a miss fill is ongoing
  - Half-precision FP load/store (with auto-conversion): popular in ML/DL
    - Reduce memory footprint and cache misses
    - Lead to better performance and power reduction
  - HW support for misaligned accesses:
    - Good for porting existing SW
    - Without it, >100 cycles are needed in the exception handler

- **Caches for low power:**
  - Only single-port SRAMs to reduce its area and power
  - Designed for fast logic power-down and wakeup

- **RTL design for high clock-gating synthesis (98%)**
# Andes Custom Extension™ (ACE)

<table>
<thead>
<tr>
<th>Items</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Instructions</strong></td>
<td></td>
</tr>
<tr>
<td>scalar</td>
<td>single-cycle, or multi-cycle</td>
</tr>
<tr>
<td>vector</td>
<td><strong>for</strong> loop, or <strong>do-while</strong> loop</td>
</tr>
<tr>
<td>background option</td>
<td>retire immediately, and continue execution in the background. Applicable to scalar and vector.</td>
</tr>
<tr>
<td><strong>Operands</strong></td>
<td></td>
</tr>
<tr>
<td>standard</td>
<td>immediate, GPR, baseline memory (thru CPU)</td>
</tr>
<tr>
<td>custom</td>
<td>- ACR (ACE Register), ACM (ACE Memory)</td>
</tr>
<tr>
<td></td>
<td>- With arbitrary width and number</td>
</tr>
<tr>
<td>implied option</td>
<td>Implied operands don’t appear in mnemonic</td>
</tr>
<tr>
<td><strong>Auto-generation by COPILOT tool</strong></td>
<td>- Opcode assignment: automatic by default</td>
</tr>
<tr>
<td></td>
<td>- All required development tools, and simulators (C or SystemC)</td>
</tr>
<tr>
<td></td>
<td>- RTL code for instruction decoding, operand mapping, dependence checking, input accesses, output updates</td>
</tr>
<tr>
<td></td>
<td>- Waveform control file</td>
</tr>
</tbody>
</table>

**Fast turnaround time!**
Inner Product for 2 64 1B Data

reg CfReg {   //Coef Registers
    num= 4;
    width= 512;
}

ram VMEM {    //data memory
    interface= sram;
    address_bits= 3;   //8 elements
    width= 512;
}

insn innerp {
    op= {out gpr IP,
         in CfReg C, in VMEM V};
    csim= %{
        //multi-precision lib. used
        IP= 0;
        for(uint i= 0; i<64; ++i)
            IP+= ((C >> (i*8)) & 0xff) * ((V >> (i*8)) & 0xff);
    };
    latency= 3;   //enable multi-cycle ctrl
};

Intrinsic: long ace_innerp(CfReg_t, VMEM_t);

//ACE_BEGIN: innerp
assign IP= C[ 7:0] * V[ 7:0] 
   + C[15:8] * V[15:8] 
   ... 
   + C[511:504] * V[511:504];

//ACE_END

Speedup: 85x
New SW For The Community

- **64-bit ThreadX from Express Logic**
  - Ported and used by a storage customer

- **Qemu enhancement:**
  - Properly handle extensions, standard or custom
  - Andes AX25 platform ready

- **Linux not ready w/o supporting components**
  - U-Boot:
    - Upstreamed and Andes as maintainer
    - Used by openSUSE project to port UEFI on AX25 platform
  - Ftrace (function trace): upstreamed
  - Module (Loadable): upstreamed
  - Perf (system profiling): patchset submitted
openSUSE

openSUSE:
- One of the major Linux distributions
- Among the first for a full RV64 target, in latest rolling release, Tumbleweed
- Packages for almost all components and rootfs images available

UEFI/EBBR:
- SUSE is one of the initiating parties of the EBBR spec.
- EBBR: provide distributions with a common boot interface across all systems
  - all parties from OS developers to system integrators can work more efficiently.
- SUSE is helping to enable EBBR for the RISC-V platform as well, starting from U-Boot.

Links:
- openSUSE:
  - https://build.opensuse.org/project/show/openSUSE:Factory:RISCV
  - https://download.opensuse.org/repositories/openSUSE::Factory::RISCV/images/
- UEFI:
  - Embedded Base Boot Requirements (EBBR): https://github.com/glikely/ebbr-for-discussion
  - https://patchwork.ozlabs.org/cover/902762/
Concluding Remarks

- RISC-V is emerging as a major application platform
- Andes offers comprehensive RISC-V solutions
  - **V5 CPUs:**
    - N25/NX25: fast-n-small cores for control tasks
    - N25F/NX25F: RV-GC cores for FP computation and AI
    - A25/AX25: Performance-efficient application processors
  - **ACE for Domain-Specific Acceleration**
    - A separate option available for all V5 cores
  - **Strong tools and SW support**
  - Let us work with you for your next SoC projects