The Libre-SOC Hybrid 3D CPU

Data-Dependent-Fail-First

FOSDEM2024

Sponsored by NLnet's PET Programme

January 11, 2024

Simple-V CMPI in a nutshell

```
function op_cmpi(BA, RA, SI) # cmpi not vector-cmpi!
  (assuming you know power-isa)
  int i, id=0, ira=0;
  for (i = 0; i < VL; i++)
    CR[BA+id] <= compare(ireg[RA+ira], SI);
    if (reg_is_vectorised[BA] ) { id += 1; }
    if (reg_is_vectorised[RA])  { ira += 1; }
```

- Above is oversimplified: predication etc. left out
- Scalar-scalar and scalar-vector and vector-vector now all in one
- OoO may choose to push CMPIs into instr. queue (v. busy!)

Load/Store Fault-First

- Problem: vector load and store can cause a page fault
- Solution: a protocol that allows optional load/store
- instruction *requests* a number of elements
- instruction *informs* the number actually loaded
- first element load/store is not optional (cannot fail)
- ARM SVE: https://arxiv.org/pdf/1803.06185.pdf
- more: wikipedia Vector processor page: Fault/Fail First

- Load/Store is Memory to/from Register, what about Register to Register?
- Register-to-register: "Data-Dependent Fail-First."
- Z80 LDIR: Mem-Register, CPIR: Register-Register

Data-Dependent-Fail-First in a nutshell

```
function op_cmpi(BA, RA, SI) # cmpi not vector-cmpi!
int i, id=0, ira=0;
for (i = 0; i < VL; i++)
    CR[BA+id] <= compare(ireg[RA+ira], SI);
    if (reg_is_vectorised[BA] ) { id += 1; }
    if (reg_is_vectorised[RA])  { ira += 1; }
    if test (CR[BA+id]) == FAIL: { VL = i + 1; break }
```

- Parallelism still perfectly possible ("hold" writing results until sequential post-analysis carried out. Best done with OoO)
- VL truncation can be inclusive or exclusive (include or exclude a NULL pointer or a string-end character, or overflow result)
- *Truncation can be to zero Vector Length*

Power ISA v3.1 vstribr

```
null_found <- 0
while(!null_found) do i = 0 to 15
     null_found <- (VSR[VRB+32].byte[15-i]=0)
   VSR[VRT+32].byte[15-i] <- VSR[VRB+32].byte[1
end
do j = i to 15
     VSR[VRT+32].byte[15-j] <- 0
end
if Rc=1 then
     CR.field[6] <- 0b00 || null_found || 0b0
```

- ironically this hard-coded instruction is identical to general-purpose Simple-V DD-FFirst...

Po

maxloc

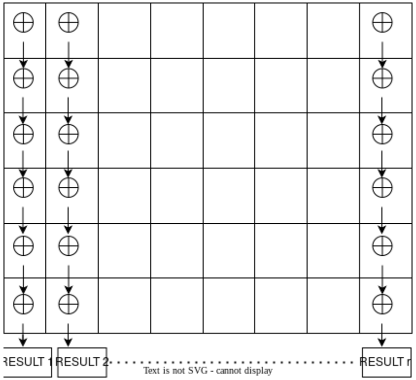
- "TODO

Pospopcount

- Positional popcount adds up the totals of each bit set to 1 in each bit-position, of an array of input values.
- Notoriously difficult to do in SIMD assembler: typically 550 lines
- https://github.com/clausecker/pospop

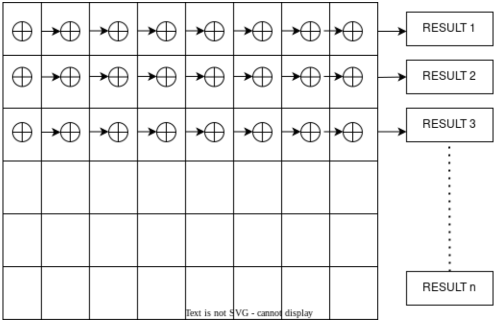
```
/*Copyright (c) 2020 Robert Clausecker fuz@fuz.su
  count8 reference implementation */
count8safe(counts *[8]int, buf []uint8)
  for i := range buf
    for j := 0; j < 8; j++
      counts[j] += int(buf[i] >> j & 1)
```

Pospopcount



- The challenge is to perform an appropriate transpose of the data (the CPU can only work on registers, horizontally), in blocks that suit the processor and the ISA capacity.

Pospopcount



- The draft gbbd instruction implements the transpose (shown above), preparing the data to use standard popcount. (gbbd is based on Power ISA vgbbd, v3.1 p445)

Pospopcount.s

```
 mtspr 9, 3                  # move r3 to CTR
 setvl 3,0,8,0,1,1           # MVL=8, VL=r3=MIN(MVL,CTR)
# load VL bytes (update r4 addr) at width=8 (dw=8)
 addi 6, 0, 0                # set all 64-bits of r6=0
 sv.lbzu/pi/dw=8 *6, 1(4)
 gbbd 8,6 # gbbd performs the transpose
# now bits are turned around, popcnt and sum them
 setvl 0,0,8,0,1,1            # set MVL=VL=8
 sv.popcntd/sw=8 *24,*8       # do (transposed) popcnt
 sv.add *16,*16,*24           # accumulate in results
# branch back if CTR non-zero works even when VL=8
 sv.bc/all 16, *0, -0x28 # reduces CTR by VL
```

strncpy

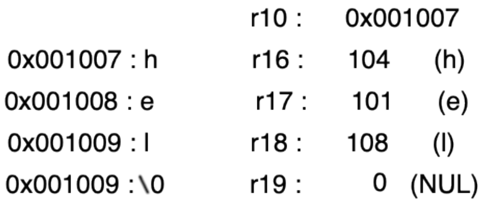
```
for (i = 0; i < n && src[i] != chr(0); i++)
    dest[i] = src[i];
for ( ; i < n; i++)
    dest[i] = chr(0);
```

- "TODO

strncpy assembler

```
     mtspr 9, 3  # move r3 to CTR
     addi 0,0,0  # initialise r0 to zero
L1: # chr-copy loop starts here:
     setvl 1,0,64,0,1,1 # VL,r1 = MIN(CTR,MVL=64)
    # load VL bytes (update r10 addr)
     sv.lbzu/pi *16, 1(10)
     sv.cmpi/ff=eq/vli *0,1,*16,0 # cmp 0, chop VL
    # store VL bytes (update r12 addr)
     sv.stbu/pi *16, 1(12)
     sv.bc/all 0, *2, L1  # stop if cmpi failed
L2: # zeroing loop starts here:
     setvl 1,0,64,0,1,1 # VL,r1 = MIN(CTR,MVL=64)
    # store VL zeros (update r12 addr)
     sv.stbu/pi 0, 1(12)
     sv.bc 16, *0, L2 # dec CTR by VL
```

sv.lbz/ff=RC1/vli *16,1(10)



	r10 :	0x001007	
0x001007 : h	r16 :	104	(h)
0x001008 : e	r17 :	101	(e)
0x001009 : l	r18 :	108	(l)
0x001009 :\0	r19 :	0	(NUL)

- r10 points to memory address 0x001007
- sv.lbz (Power ISA load byte immediate) multiplies immediate offset by element step index, to get Effective Address (EA)
- LD/ST has no Rc=1 so Data-Dependent Fail-First specified as "ff=RC1". Not LD/ST Fault First! vli: VL inclusive
- Test done after each load. Fails at Memory contents 0x001009. Inclusive Mode: VL is truncated to 5 (FIVE) not 4

linked-list walking

- "TODO

Summary

- Goal is to create a mass-volume low-power embedded SoC suitable for use in netbooks, chromebooks, tablets, smartphones, IoT SBCs.
- No way we could implement a project of this magnitude without nmigen (being able to use python OO to HDL)
- Collaboration with OpenPOWER Foundation and Members absolutely essential. No short-cuts. Standards to be developed and ratified so that everyone benefits.
- Riding the wave of huge stability of OpenPOWER ecosystem
- Greatly simplified open 3D and Video drivers reduces product development costs for customers
- It also happens to be fascinating, deeply rewarding technically challenging, and funded by NLnet

How can you help?

- Start here! https://libre-soc.org
  Mailing lists https://lists.libre-soc.org
  IRC Freenode libre-soc
  etc. etc. (it's a Libre project, go figure)
- Can I get paid? Yes! NLnet funded
  See https://libre-soc.org/nlnet/#faq
- Also profit-sharing in any commercial ventures
- How many opportunities to develop Libre SoCs exist, and actually get paid for it?
- I'm not a developer, how can I help?
  - Plenty of research needed, artwork, website
  - Help find customers and OEMs willing to commit (LOI)

The end

Thank you

Questions?

- Discussion: http://lists.libre-soc.org
- Freenode IRC #libre-soc
- http://libre-soc.org/
- http://nlnet.nl/PET
- https://libre-soc.org/nlnet/#faq