The Quiet Giant in RISC-V

RISC-V Virtual Hackathon

Softmax Challenge for NNs

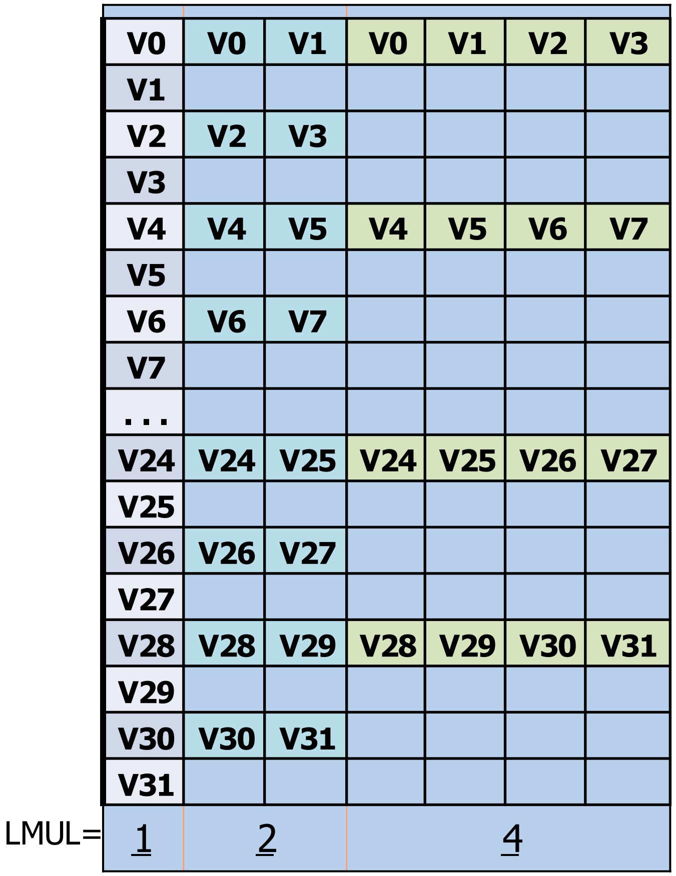
- Background Knowledge

Agenda

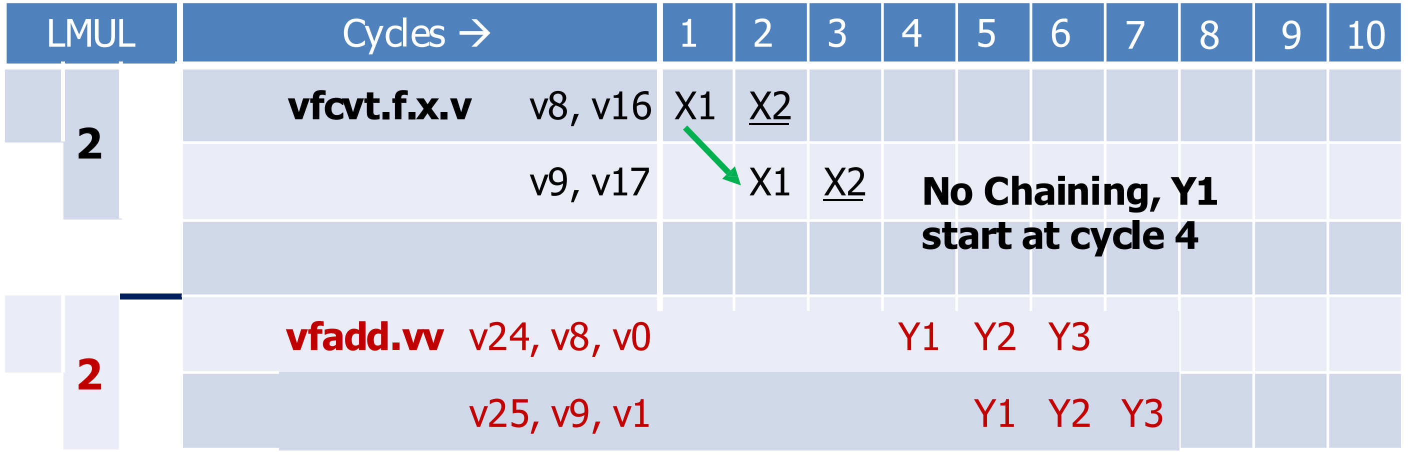
- RISC-V Vector Instruction/Extension(RVV)
- Intrinsic Function
- Andes Custom Extensions (ACE)
- Debug

Vector Registers Chaining

- **32 Vector Registers (VR)**
  - Each with VLEN bits, depending on HW config.
- **Data formats:**
  - SEW (standard element width): 8,16,32,64-bit
  - int8, int16, int32, int64, fp16, fp32, fp64, bf16
- **EX: VLEN=512, LMUL=1, each VR has**
  - **16** elements when SEW=32 (int32/fp32)
  - **32** elements when SEW=16 (int16/fp16/bf16)
  - **64** elements when SEW=8 (int8)
- **LMUL (Length Multiplier): VR combining**
  - Can be set to 1, 2, 4, 8 or 1/2, 1/4, 1/8 at runtime by SW
  - Example 1
    - VLEN=512 and LMUL=8
    - v0 represents v0~v7, or effectively a **4096-bit** (512-bitx8) register with 128 fp32 data
  - Example 2
    - For VLEN=128 and LMUL=1/4
      - 4 elements when SEW=8
      - 2 elements when SEW=16

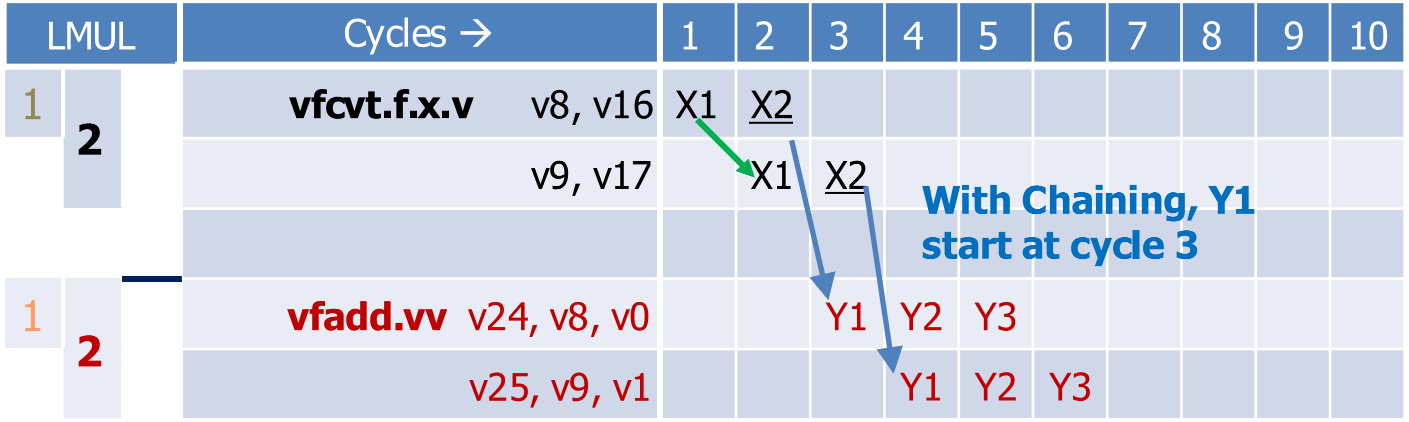


AX45MPV : Chaining with LMUL=2



LMUL		Cycles →		1	2	3	4	5	6	7	8	9	10
	2	**vfcvt.f.x.v**	v8, v16	X1	X2								
			v9, v17		X1	X2							
	2	**vfadd.vv**	v24, v8, v0				Y1	Y2	Y3				
			v25, v9, v1					Y1	Y2	Y3			

No Chaining, Y1 start at cycle 4



LMUL		Cycles →		1	2	3	4	5	6	7	8	9	10
1	2	**vfcvt.f.x.v**	v8, v16	X1	X2								
			v9, v17		X1	X2							
1	2	**vfadd.vv**	v24, v8, v0			Y1	Y2	Y3					
			v25, v9, v1				Y1	Y2	Y3				

With Chaining, Y1 start at cycle 3

Setting in CSR vtype:

- **LMUL=1 → 512 bits**
- **LMUL=2 → 1024 bits**
  - **V0~V1 group 0**
  - **V2~V3 group 1**
  - **V4~V5 group 2**
  - **. . .**
  - **V30~V31 group 15**

Xn, Yn: execution stages

LMUL →

Chaining →

RISC-V Vector(RVV) Instruction/Extension

- For vector instructions, each instruction will work on each element on the vector register. That means it’s a SIMD operation.
- For more details, please refer to the RISC-V Vector Extension spec, i.e. riscv-v-spec-1.0-rc2.pdf.

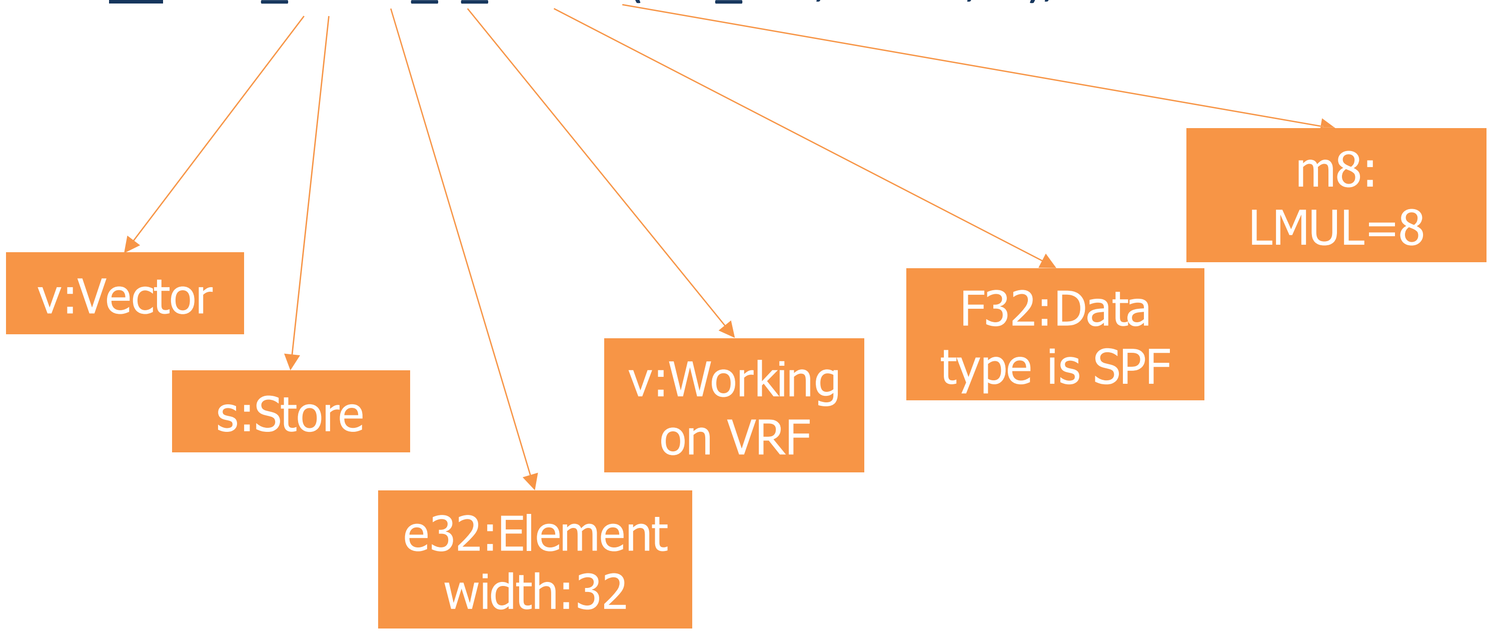
Intrinsic Function

- The intrinsic functions are for users who don’t want to program in assembly. They cover all the operations which compiler cannot generate.
- It avoids the overhead of a function call and allows efficient machine instructions to be emitted for that function.
- These functions available in a given language whose implementation is handled specially by the compiler.
- Intrinsic function is usually inserted inline.
- For more details about the vector intrinsic functions, please refer to RISC-V_Vector_(V)_Extension_Intrinsics_UM231_V1.5.pdf.

- NOTE: Be sure to use the correct signedness for arguments and return values when calling intrinsic functions.

RVV Intrinsic Function

- __riscv_vse32_v_f32m8(out_vec, vData, vl);



Agenda

- RISC-V Vector Instruction/Extension(RVV)
- Intrinsic Function
- Andes Custom Extensions (ACE)
- Debug

ACE

- ACE: Andes Custom Extension. Andes provides ACE package for customers to create custom instruction. The tool is COPILOT. The input of the tool is an .ace file which describes the name, input operands, output operand, and csim behavior model of the instruction.
- ace_user.h is one of the output file of COPILOT. The file includes the intrinsic functions of the custom instructions, pre-fixed with ace_ for each instruction.
- libacesim.so and libacetool.so are generated libraries from COPILOT for Sid simulator and toolchain respectively.

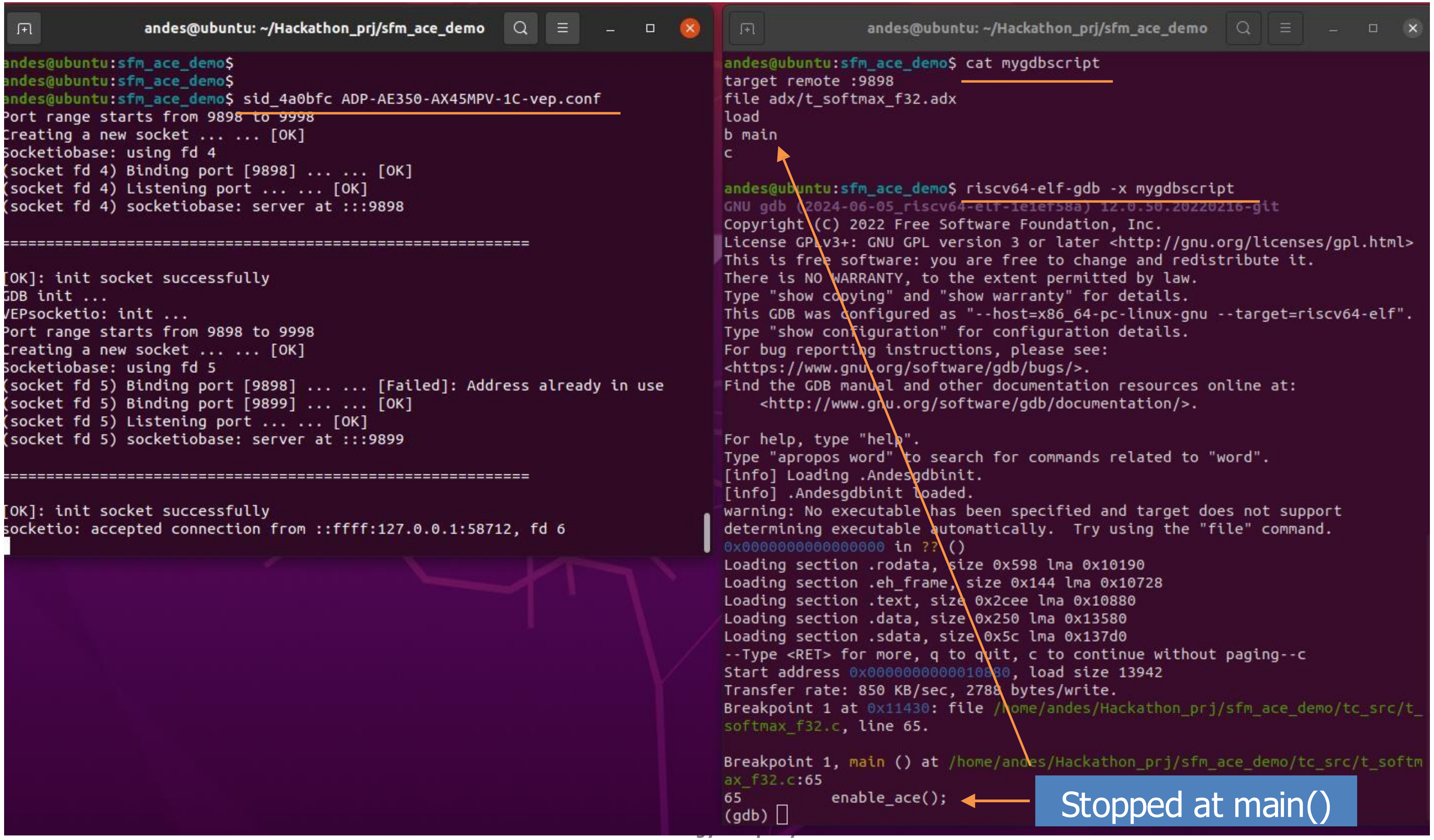
ACE_RVV

- ACE_RVV: is to create vector instructions working on the VPU.
- For more details, please refer to Andes_Custom_Extension_Programmer's_Manual.pdf

Agenda

- RISC-V Vector Instruction/Extension(RVV)
- Intrinsic Function
- Andes Custom Extensions (ACE)
- Debug

GDB Debug Example – In AndeSim Simulator



andes@ubuntu: ~/Hackathon_prj/sfm_ace_demo

```
andes@ubuntu:sfm_ace_demo$
andes@ubuntu:sfm_ace_demo$
andes@ubuntu:sfm_ace_demo$ sid_4a0bfc ADP-AE350-AX45MPV-1C-vep.conf
Port range starts from 9898 to 9998
Creating a new socket ... ... [OK]
Socketiobase: using fd 4
(socket fd 4) Binding port [9898] ... ... [OK]
(socket fd 4) Listening port ... ... [OK]
(socket fd 4) socketiobase: server at :::9898

============================================================

[OK]: init socket successfully
GDB init ...
VEPsocketio: init ...
Port range starts from 9898 to 9998
Creating a new socket ... ... [OK]
Socketiobase: using fd 5
(socket fd 5) Binding port [9898] ... ... [Failed]: Address already in use
(socket fd 5) Binding port [9899] ... ... [OK]
(socket fd 5) Listening port ... ... [OK]
(socket fd 5) socketiobase: server at :::9899

============================================================

[OK]: init socket successfully
socketio: accepted connection from ::ffff:127.0.0.1:58712, fd 6
```

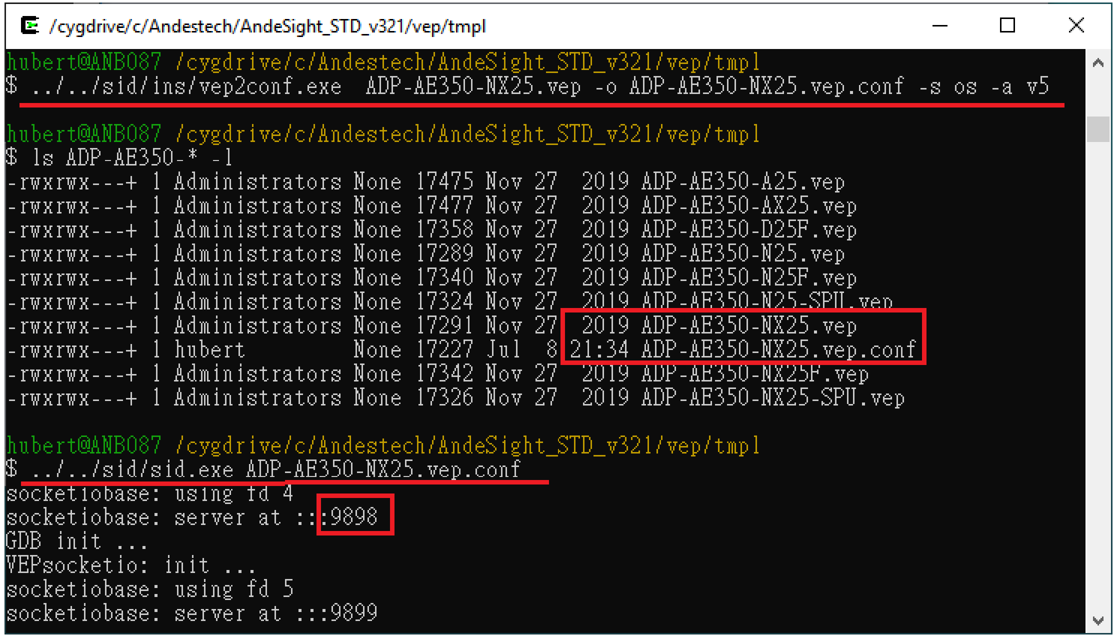
andes@ubuntu: ~/Hackathon_prj/sfm_ace_demo

```
andes@ubuntu:sfm_ace_demo$ cat mygdbscript
target remote :9898
file adx/t_softmax_f32.adx
load
b main
c

andes@ubuntu:sfm_ace_demo$ riscv64-elf-gdb -x mygdbscript
GNU gdb (2024-06-05_riscv64-elf-1e1ef58a) 12.0.50.20220216-git
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "--host=x86_64-pc-linux-gnu --target=riscv64-elf".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word".
[info] Loading .Andesgdbinit.
[info] .Andesgdbinit loaded.
warning: No executable has been specified and target does not support
determining executable automatically.  Try using the "file" command.
0x0000000000000000 in ?? ()
Loading section .rodata, size 0x598 lma 0x10190
Loading section .eh_frame, size 0x144 lma 0x10728
Loading section .text, size 0x2cee lma 0x10880
Loading section .data, size 0x250 lma 0x13580
Loading section .sdata, size 0x5c lma 0x137d0
--Type <RET> for more, q to quit, c to continue without paging--c
Start address 0x0000000000010880, load size 13942
Transfer rate: 850 KB/sec, 2788 bytes/write.
Breakpoint 1 at 0x11430: file /home/andes/Hackathon_prj/sfm_ace_demo/tc_src/t_
softmax_f32.c, line 65.

Breakpoint 1, main () at /home/andes/Hackathon_prj/sfm_ace_demo/tc_src/t_softm
ax_f32.c:65
65          enable_ace();
(gdb)
```

Stopped at main()

GDB Debug Example – In AndeSim Simulator

- All the examples are for 32 bit CPU. In the competition, since AX45MPV is a 64-bit CPU. Please change all “32” to “64”.
- Run sid with a config file: a gdb server is in the sid

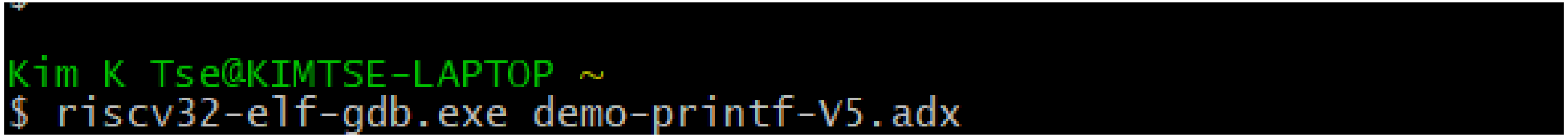


/cygdrive/c/Andestech/AndeSight_STD_v321/vep/tmpl

```
hubert@ANB087 /cygdrive/c/Andestech/AndeSight_STD_v321/vep/tmpl
$ ../../sid/ins/vep2conf.exe  ADP-AE350-NX25.vep -o ADP-AE350-NX25.vep.conf -s os -a v5

hubert@ANB087 /cygdrive/c/Andestech/AndeSight_STD_v321/vep/tmpl
$ ls ADP-AE350-* -l
-rwxrwx---+ 1 Administrators None 17475 Nov 27  2019 ADP-AE350-A25.vep
-rwxrwx---+ 1 Administrators None 17477 Nov 27  2019 ADP-AE350-AX25.vep
-rwxrwx---+ 1 Administrators None 17358 Nov 27  2019 ADP-AE350-D25F.vep
-rwxrwx---+ 1 Administrators None 17289 Nov 27  2019 ADP-AE350-N25.vep
-rwxrwx---+ 1 Administrators None 17340 Nov 27  2019 ADP-AE350-N25F.vep
-rwxrwx---+ 1 Administrators None 17324 Nov 27  2019 ADP-AE350-N25-SPU.vep
-rwxrwx---+ 1 Administrators None 17291 Nov 27  2019 ADP-AE350-NX25.vep
-rwxrwx---+ 1 hubert         None 17227 Jul  8 21:34 ADP-AE350-NX25.vep.conf
-rwxrwx---+ 1 Administrators None 17342 Nov 27  2019 ADP-AE350-NX25F.vep
-rwxrwx---+ 1 Administrators None 17326 Nov 27  2019 ADP-AE350-NX25-SPU.vep

hubert@ANB087 /cygdrive/c/Andestech/AndeSight_STD_v321/vep/tmpl
$ ../../sid/sid.exe ADP-AE350-NX25.vep.conf
socketiobase: using fd 4
socketiobase: server at :::9898
GDB init ...
VEPsocketio: init ...
socketiobase: using fd 5
socketiobase: server at :::9899
```

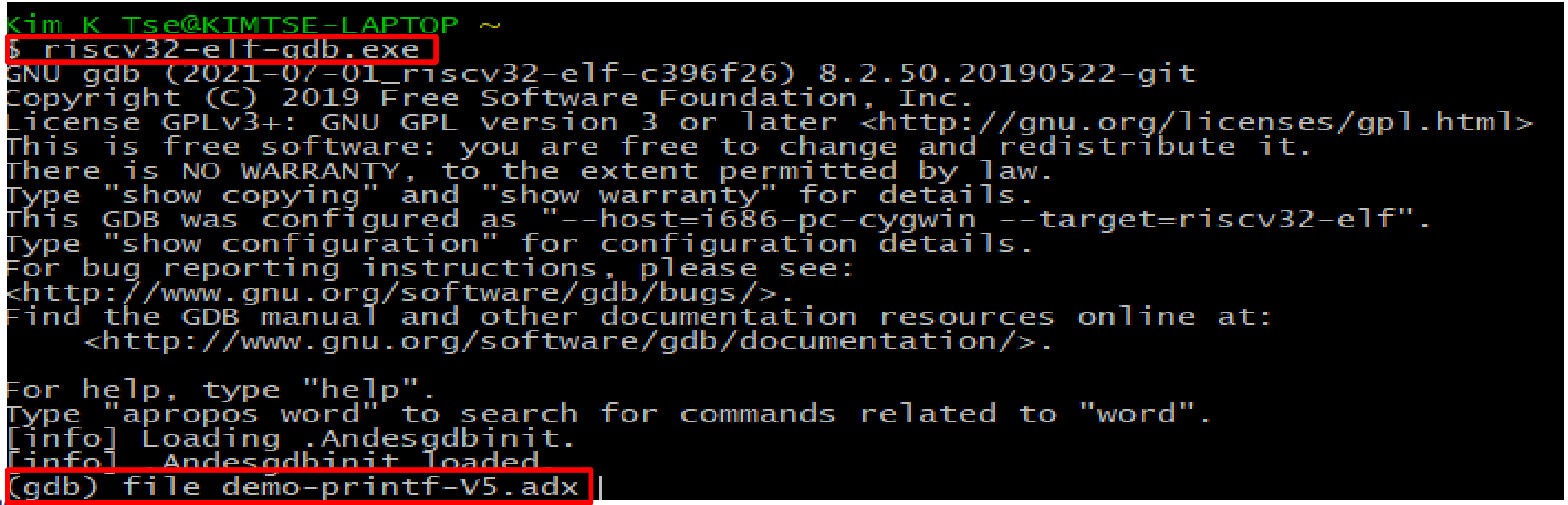
GDB Command – Select a Debug File

- GDB program – Debug program



```
Kim K Tse@KIMTSE-LAPTOP ~
$ riscv32-elf-gdb.exe demo-printf-V5.adx
```

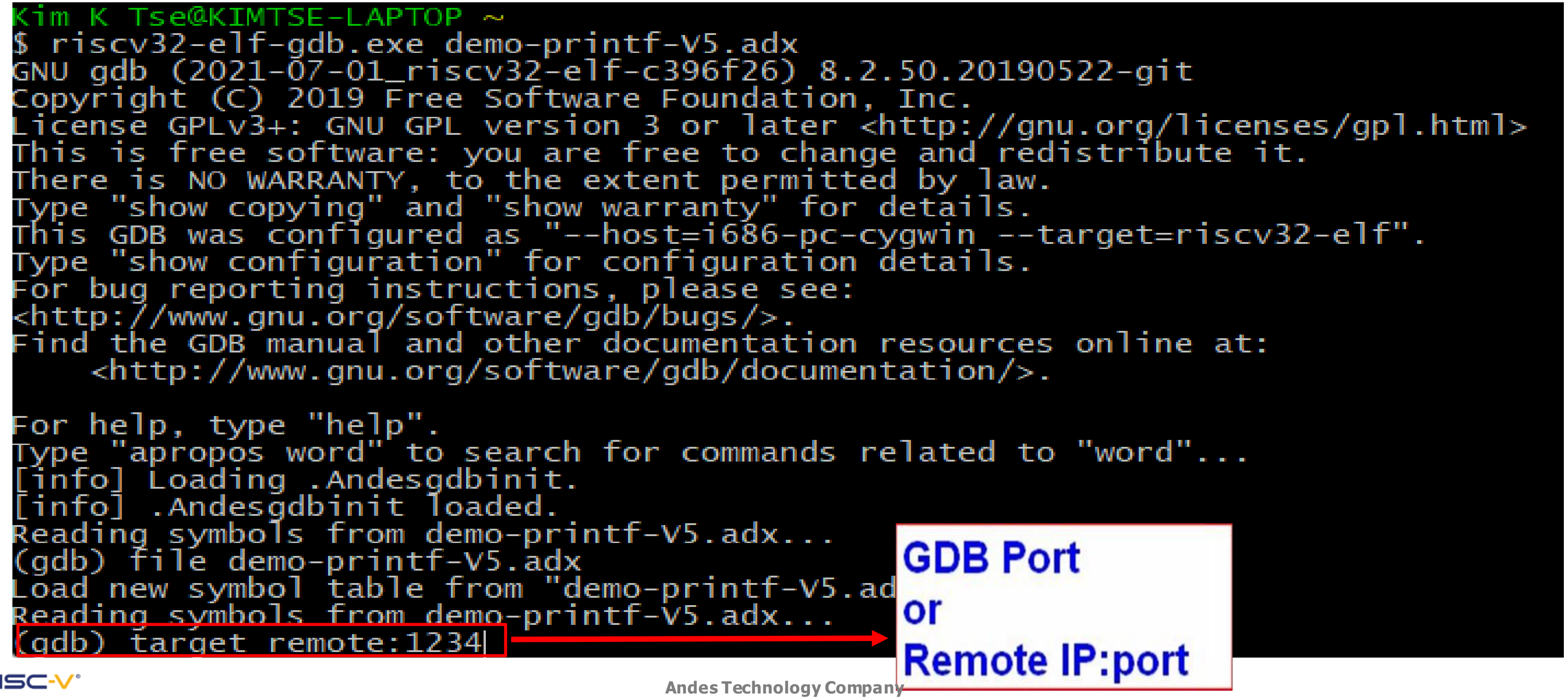
- (gdb) file [file] – Use file for symbols & executable



```
Kim K Tse@KIMTSE-LAPTOP ~
$ riscv32-elf-gdb.exe
GNU gdb (2021-07-01_riscv32-elf-c396f26) 8.2.50.20190522-git
Copyright (C) 2019 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "--host=i686-pc-cygwin --target=riscv32-elf".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word".
[info] Loading .Andesgdbinit.
[info] .Andesgdbinit loaded.
(gdb) file demo-printf-V5.adx
```

GDB Connect to Target Platform

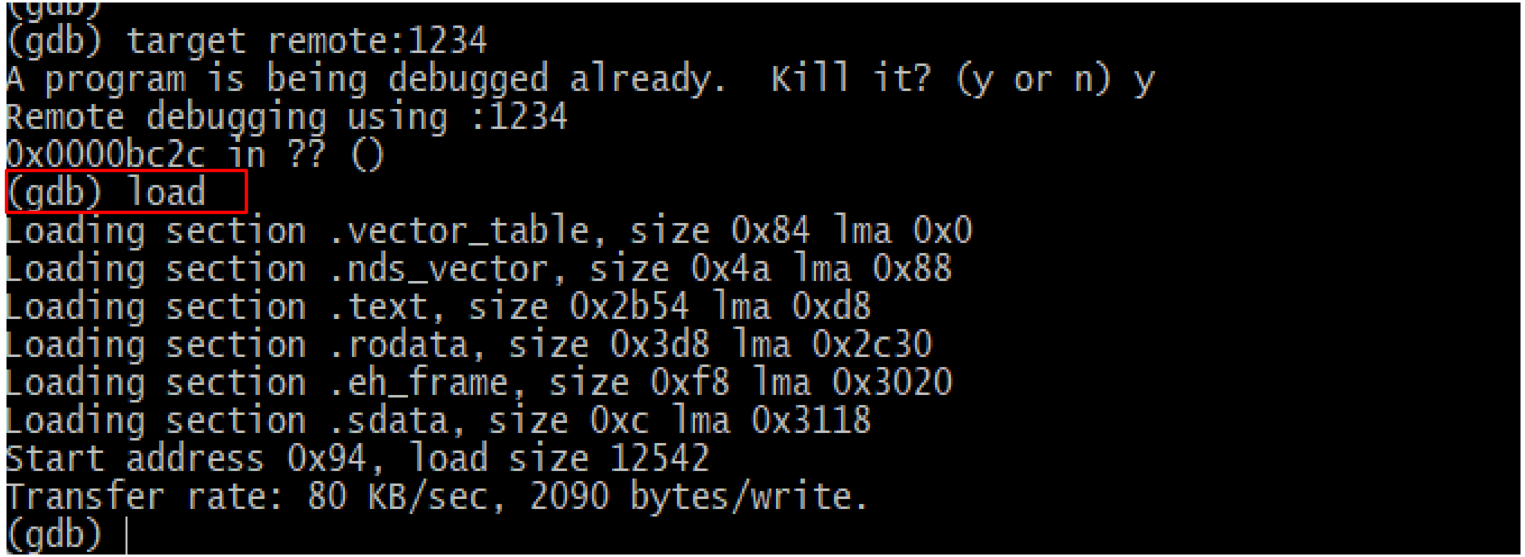
- (gdb) target remote :port
- (gdb) target remote host_ip:port – Remote debugging



```
Kim K Tse@KIMTSE-LAPTOP ~
$ riscv32-elf-gdb.exe demo-printf-V5.adx
GNU gdb (2021-07-01_riscv32-elf-c396f26) 8.2.50.20190522-git
Copyright (C) 2019 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "--host=i686-pc-cygwin --target=riscv32-elf".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
[info] Loading .Andesgdbinit.
[info] .Andesgdbinit loaded.
Reading symbols from demo-printf-V5.adx...
(gdb) file demo-printf-V5.adx
Load new symbol table from "demo-printf-V5.ad
Reading symbols from demo-printf-V5.adx...
(gdb) target remote:1234
```

GDB Command – Load Code

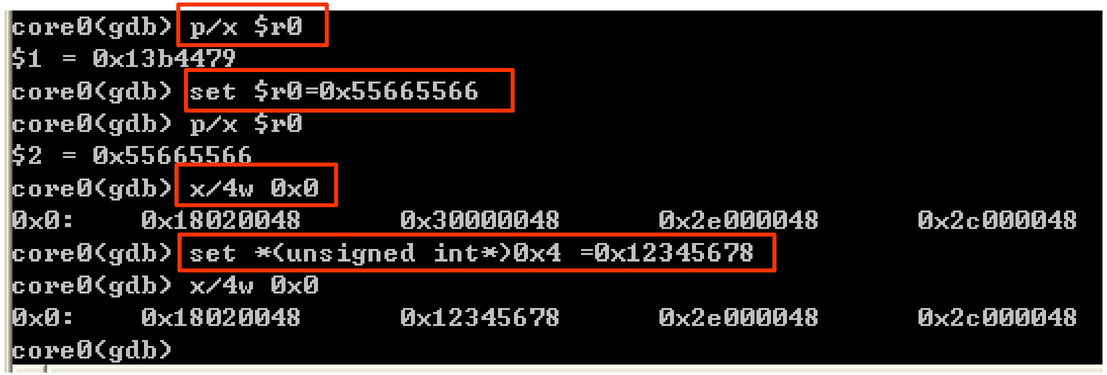
- (gdb) load



```
(gdb) target remote:1234
A program is being debugged already.  Kill it? (y or n) y
Remote debugging using :1234
0x0000bc2c in ?? ()
(gdb) load
Loading section .vector_table, size 0x84 lma 0x0
Loading section .nds_vector, size 0x4a lma 0x88
Loading section .text, size 0x2b54 lma 0xd8
Loading section .rodata, size 0x3d8 lma 0x2c30
Loading section .eh_frame, size 0xf8 lma 0x3020
Loading section .sdata, size 0xc lma 0x3118
Start address 0x94, load size 12542
Transfer rate: 80 KB/sec, 2090 bytes/write.
(gdb) 
```

GDB Command – Read/Write Register and Memory

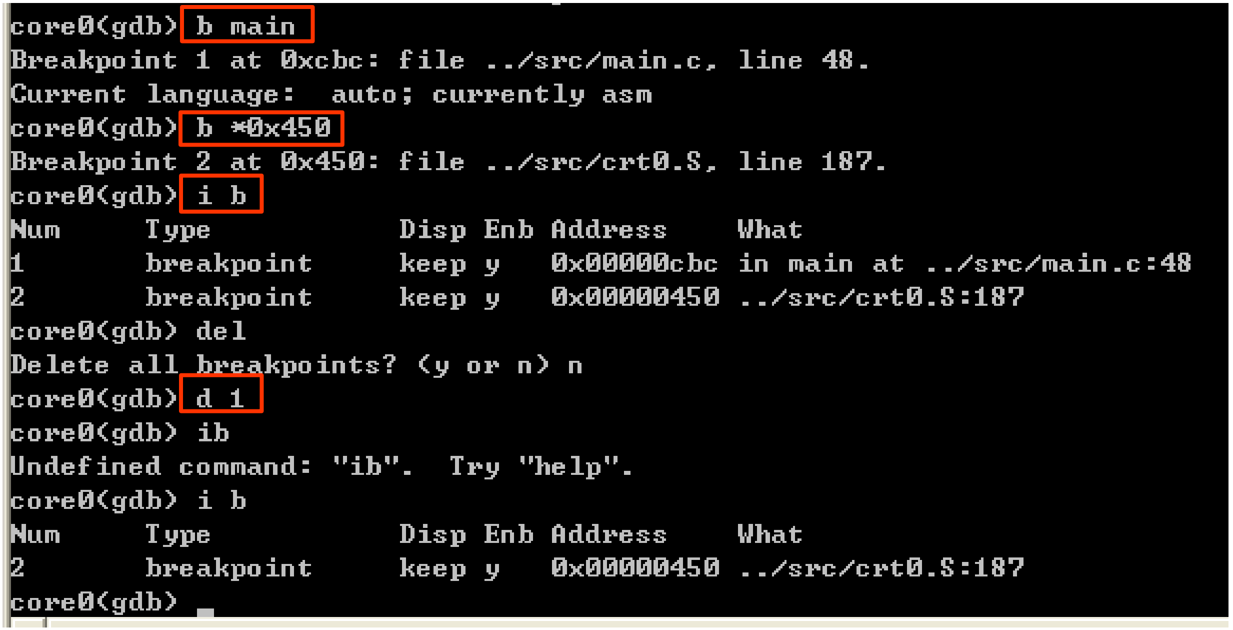
- (gdb) p/x $r0 – Print register
- (gdb) set $r0=0x55665566 – Set register
- (gdb) x/4w 0x0
  – Examine memory (4w→4 words, 0x0→address)
- (gdb) set *(unsigned int*) 0x4=0x12345678
  – Set memory
- (gdb) p variable – Print variable



```
core0(gdb) p/x $r0
$1 = 0x13b4479
core0(gdb) set $r0=0x55665566
core0(gdb) p/x $r0
$2 = 0x55665566
core0(gdb) x/4w 0x0
0x0:    0x18020048      0x30000048      0x2e000048      0x2c000048
core0(gdb) set *(unsigned int*)0x4 =0x12345678
core0(gdb) x/4w 0x0
0x0:    0x18020048      0x12345678      0x2e000048      0x2c000048
core0(gdb)
```

GDB Command – Set Breakpoint (1)

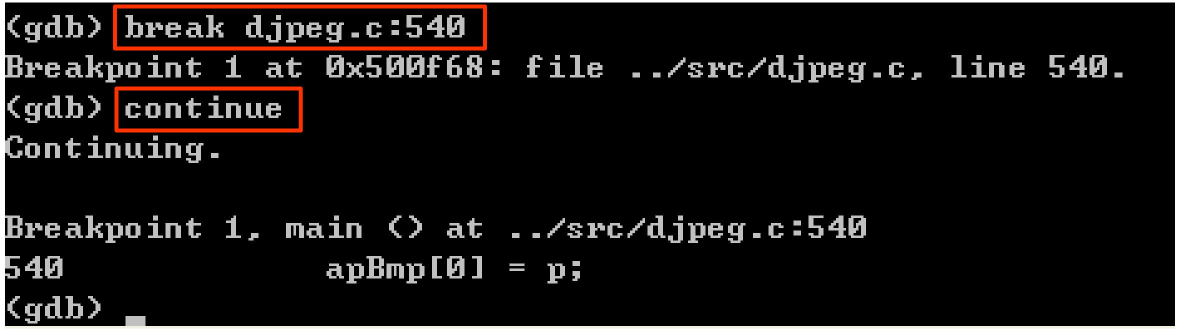
- (gdb) break *address – Set a breakpoint at address “address”.
- (gdb) break function – Set a breakpoint at entry of function “function”.



```
core0(gdb) b main
Breakpoint 1 at 0xcbc: file ../src/main.c, line 48.
Current language:  auto; currently asm
core0(gdb) b *0x450
Breakpoint 2 at 0x450: file ../src/crt0.S, line 187.
core0(gdb) i b
Num     Type           Disp Enb Address    What
1       breakpoint     keep y   0x00000cbc in main at ../src/main.c:48
2       breakpoint     keep y   0x00000450 ../src/crt0.S:187
core0(gdb) del
Delete all breakpoints? (y or n) n
core0(gdb) d 1
core0(gdb) ib
Undefined command: "ib".  Try "help".
core0(gdb) i b
Num     Type           Disp Enb Address    What
2       breakpoint     keep y   0x00000450 ../src/crt0.S:187
core0(gdb) _
```

GDB Command – Set Breakpoint (2)

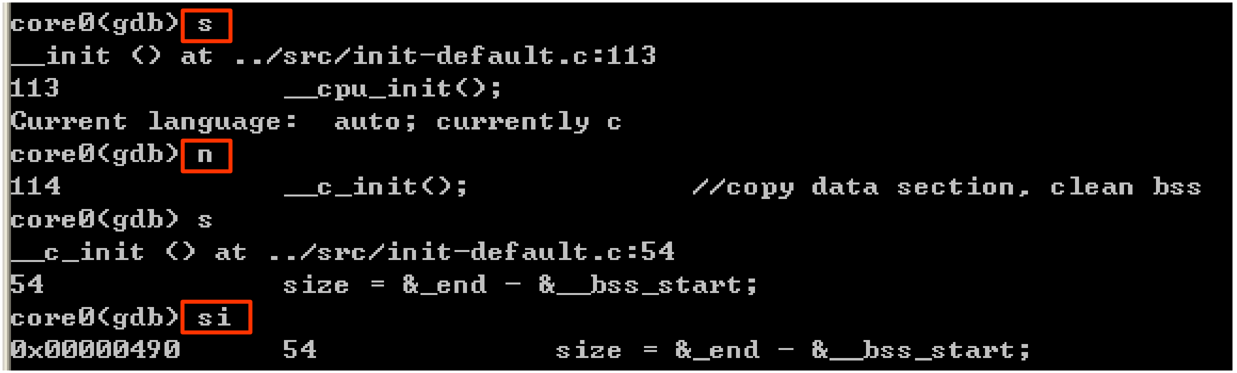
- (gdb) break filename:linenum – Set a breakpoint at line linenum in source file filename.
- (gdb) hbreak args – Set a HW breakpoint (Trigger Module).
- (gdb) tbreak args – Set a temporary breakpoint only stopping once.
- (gdb) continue (or c) – Continue means resuming program execution your program completes normally.



```
(gdb) break djpeg.c:540
Breakpoint 1 at 0x500f68: file ../src/djpeg.c, line 540.
(gdb) continue
Continuing.

Breakpoint 1, main () at ../src/djpeg.c:540
540                 apBmp[0] = p;
(gdb) _
```

GDB Command – Stepping (1)

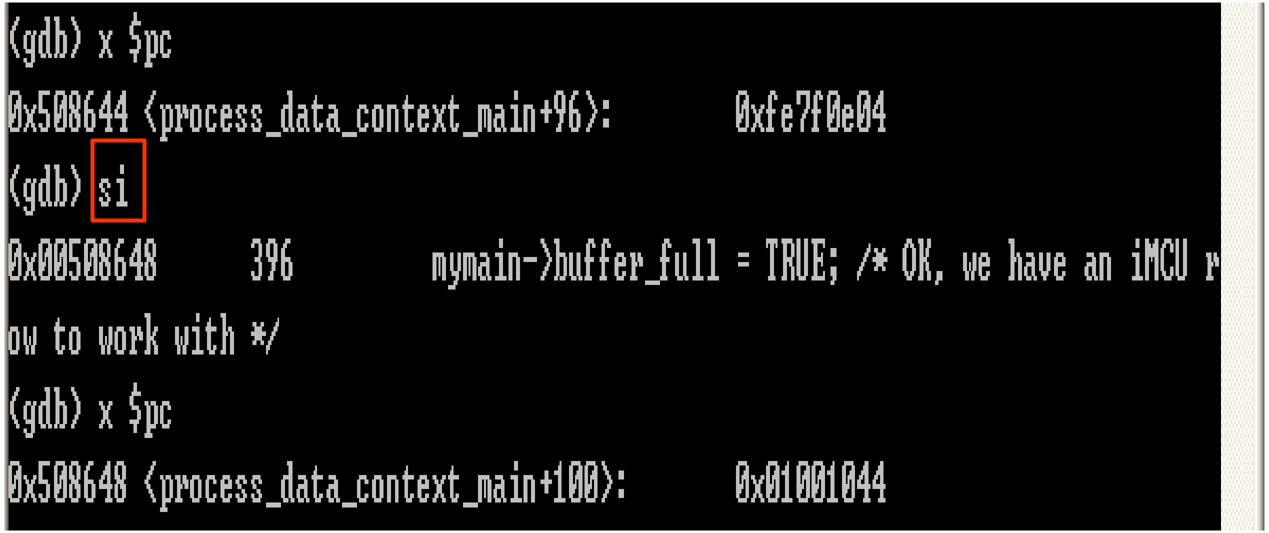
- Stepping means executing just one more "step" of your program, where "step" may mean either one line of source code, or one machine instruction.
- (gdb) step (s) – Execute a single statement. If the statement is a function call, just single step into the function.
- (gdb) next (n) – Execute a single statement. If the statement is a function call, execute the entire function and return to the statement just after the call; that is, step over the function.



```
core0(gdb) s
__init () at ../src/init-default.c:113
113             __cpu_init();
Current language:  auto; currently c
core0(gdb) n
114             __c_init();             //copy data section, clean bss
core0(gdb) s
__c_init () at ../src/init-default.c:54
54              size = &_end - &__bss_start;
core0(gdb) si
0x00000490      54              size = &_end - &__bss_start;
```

GDB Command – Stepping (2)

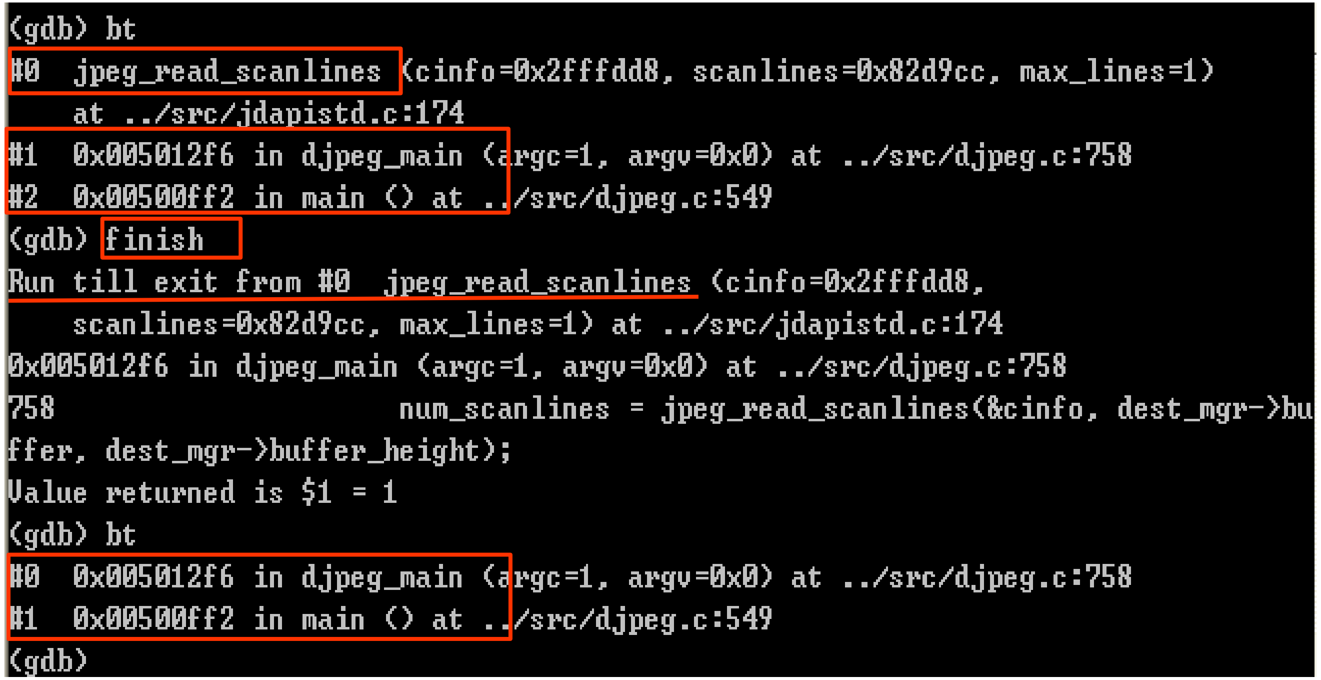
- (gdb) stepi (si) – Execute one machine instruction, then stop and return to the debugger.



```
(gdb) x $pc
0x508644 <process_data_context_main+96>:       0xfe7f0e04
(gdb) si
0x00508648      396             mymain->buffer_full = TRUE; /* OK, we have an iMCU r
ow to work with */
(gdb) x $pc
0x508648 <process_data_context_main+100>:      0x01001044
```

GDB Command – Stepping (3)

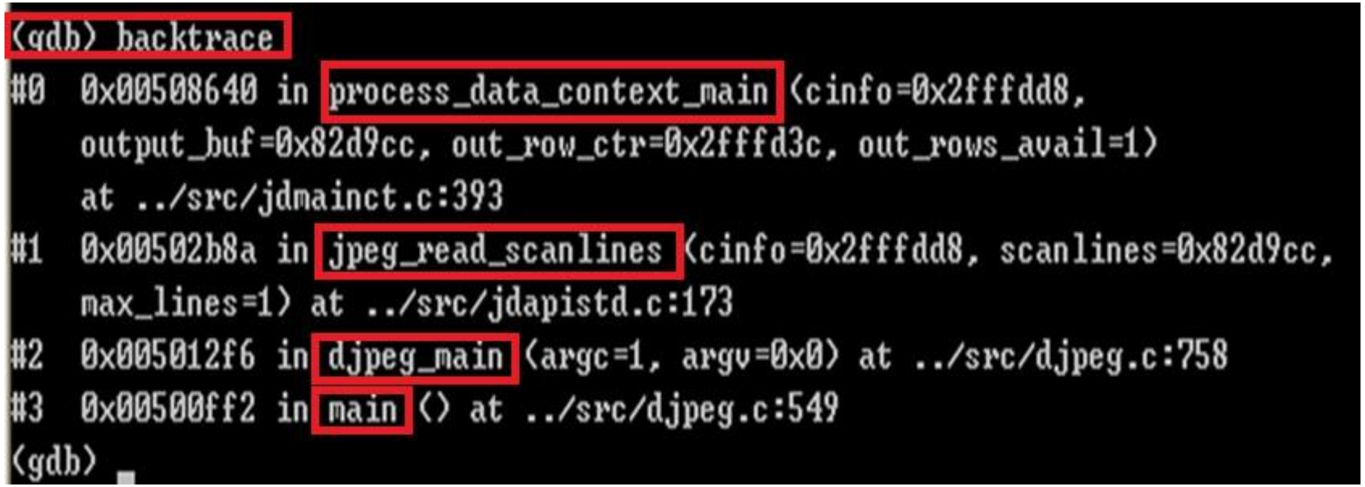
- (gdb) finish – Execute the rest of the current function; that is, step out of the function.



```
(gdb) bt
#0  jpeg_read_scanlines (cinfo=0x2fffdd8, scanlines=0x82d9cc, max_lines=1)
    at ../src/jdapistd.c:174
#1  0x005012f6 in djpeg_main (argc=1, argv=0x0) at ../src/djpeg.c:758
#2  0x00500ff2 in main () at ../src/djpeg.c:549
(gdb) finish
Run till exit from #0  jpeg_read_scanlines (cinfo=0x2fffdd8,
    scanlines=0x82d9cc, max_lines=1) at ../src/jdapistd.c:174
0x005012f6 in djpeg_main (argc=1, argv=0x0) at ../src/djpeg.c:758
758                       num_scanlines = jpeg_read_scanlines(&cinfo, dest_mgr->bu
ffer, dest_mgr->buffer_height);
Value returned is $1 = 1
(gdb) bt
#0  0x005012f6 in djpeg_main (argc=1, argv=0x0) at ../src/djpeg.c:758
#1  0x00500ff2 in main () at ../src/djpeg.c:549
(gdb)
```

GDB Command – Backtrace

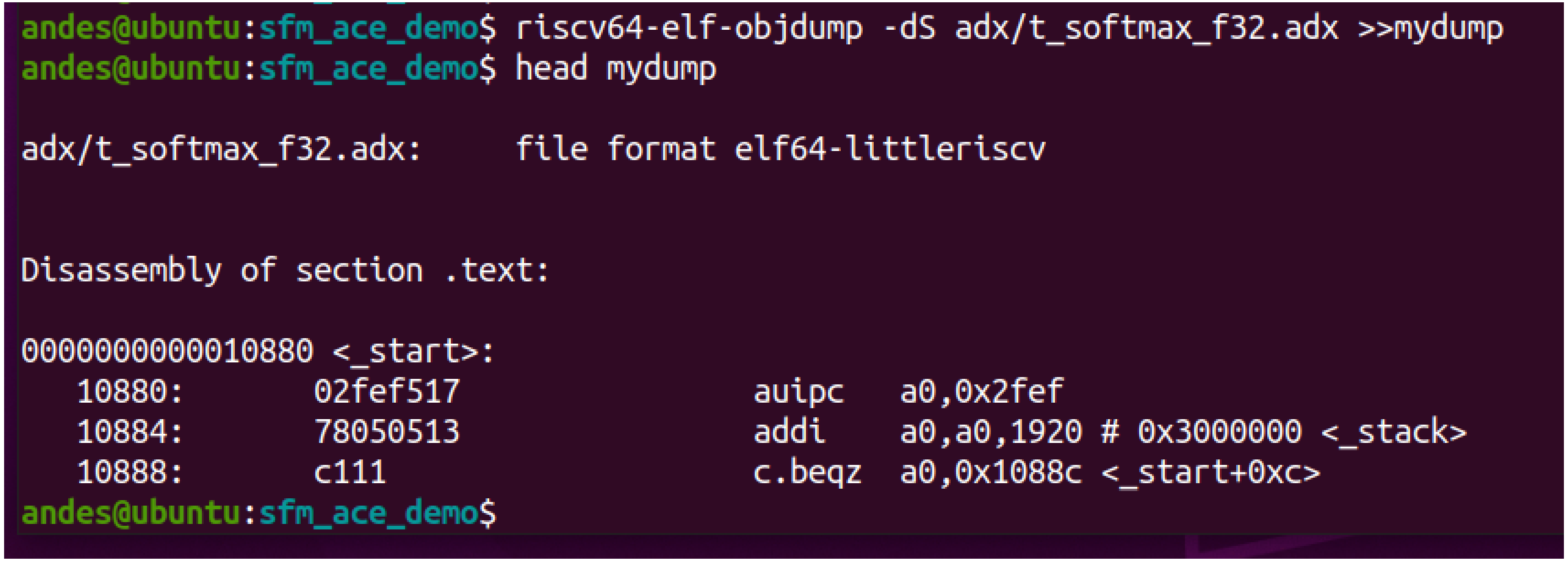
- A backtrace is a summary of how your program got where it is. It shows one line per frame, for many frames, starting with the currently executing frame (frame zero), followed by its caller (frame one), and on up the stack.
- (gdb) backtrace (bt) – Print a backtrace of the entire stack: one line per frame for all frames in the stack.



```
(gdb) backtrace
#0  0x00508640 in process_data_context_main (cinfo=0x2fffdd8,
    output_buf=0x82d9cc, out_row_ctr=0x2fffd3c, out_rows_avail=1)
    at ../src/jdmainct.c:393
#1  0x00502b8a in jpeg_read_scanlines (cinfo=0x2fffdd8, scanlines=0x82d9cc,
    max_lines=1) at ../src/jdapistd.c:173
#2  0x005012f6 in djpeg_main (argc=1, argv=0x0) at ../src/djpeg.c:758
#3  0x00500ff2 in main () at ../src/djpeg.c:549
(gdb) _
```

Tools

- **risc64-elf-objdump: dump the elf format file to a readable**
  - riscv64-elf-objdump -dS adx/t_softmax_f32.adx >> mydump



```
andes@ubuntu:sfm_ace_demo$ riscv64-elf-objdump -dS adx/t_softmax_f32.adx >>mydump
andes@ubuntu:sfm_ace_demo$ head mydump

adx/t_softmax_f32.adx:     file format elf64-littleriscv


Disassembly of section .text:

0000000000010880 <_start>:
   10880:       02fef517                auipc   a0,0x2fef
   10884:       78050513                addi    a0,a0,1920 # 0x3000000 <_stack>
   10888:       c111                    c.beqz  a0,0x1088c <_start+0xc>
andes@ubuntu:sfm_ace_demo$
```

ANDES

The Quiet Giant in RISC-V

Thank You!