State of LLDB and Deeply Embedded RISC-V

An update on D62732

Simon Cook

Rationales

- Complete “LLVM only” tool chain
  - Have compiler/assembler/linker/compiler-rt, now need a debugger
- Debug bare metal applications
  - Set breakpoints, step/continue, backtrace
  - Load new binaries into memory
- Testcase for our Debug Server

Current Status

- Have initial `ABISysV_riscv` class implemented
  - Supports RV32/RV64
  - Default unwind plan for previous frame
- Extend `DisassemblerLLVMMC` to support RISC-V
  - Assuming `amfdc` extensions for disassembly
- Add support for software breakpoints
  - Currently assumes that compressed instructions are available
- Start of (non-JIT) `PrepareTrivialCall` implementation

Current Status

```
simon@hartnell$ ./build-llvm/bin/lldb fasta

(lldb) target create "fasta"
Current executable set to '/home/simon/work/rvlldb/fasta' (riscv32).

(lldb) gdb-remote 51000
Process 1 stopped
* thread #1, stop reason = signal SIGTRAP
frame #0: 0x00010604 fasta`__addsf3(a=0, b=0) at addsf3.c:45:3

(lldb) bt
* thread #1, stop reason = signal SIGTRAP
* frame #0: 0x00010604 fasta`__addsf3(a=0, b=0) at addsf3.c:45:3
frame #1: 0x000102c6 fasta`accumulate_probabilities(genelist_in=0x00011894,
genelist_out=0xfffffb60, len=15) at libfasta.c:101:12
frame #2: 0x00010558 fasta`benchmark at libfasta.c:216:7
frame #3: 0x000101b0 fasta`main(argc=0, argv=0xfffffc24) at main.c:44:12
frame #4: 0x000100b0 fasta`_start + 60
```

Current Status

```
simon@hartnell$ ./build-llvm/bin/lldb a.out

(lldb) target create "a.out"
Current executable set to '/home/simon/work/rvlldb/a.out' (riscv32).

(lldb) gdb-remote 51000
Process 1 stopped
* thread #1, stop reason = signal SIGTRAP
    frame #0: 0x00010188 a.out`main at test.c:8:8

(lldb) list foo
File: /home/simon/work/rvlldb/testcase/test.c
   1 int foo() {
   2 return 2;
   3 }

(lldb) call foo()
(int) $0 = 2
```

EMBECOSM®

Todo/Open Questions

- Better subtarget detection
  - Could this be done via `qHostInfo` or Target XML?
- Calling functions loaded on target
  - Start of implementation – but can't see how to set `m_can_interpret_function_calls` for the non-JIT case?
- Loading binaries
  - I couldn't find any way of doing this other than running a new process – Do I have to support `A` packet in my server?
- Testing

Questions?

www.embecosm.com