Tutorial 7.1

File I/O, floating-point instructions

Lei Wang

lei.wang2@ucalgary.ca

File I/O

- The type of system call is put into *x8*
- The arguments are put into *x0-x5*
- Execute the system call with the instruction *svc*

x8	Service Request
56	openat
57	close
63	read
64	write

```
int fd = openat(int dirfd, const char *pathname, int flags, mode_t mode);
long n_read = read(int fd, void *buf, unsigned long n);
long n_written = write(int fd, void *buf, unsigned long n);
int status = close(int fd);
```

Write file/read file

- writelong.asm
  - write 1-100 into *output.bin*
- Readlong.asm
  - read *output.bin* and print

Floating-point arithmetic

- ARMv8 has 32 floating point registers
  - *d0-d32* are 64 bit double precision registers
  - *s0-s31* are the lower 32 bits of these registers and are used for single precision Floating-Point
  - *d8-d15* are callee saved registers
  - *d0-d7* and *d16-d31* may be overwritten by subroutines
  - *d0-d7* are used to pass floating point arguments into a function

Floating point arithmetic

- use %f as format strings to print floating point numbers
- use p/f or display/f to print the contents of a Floating-Point register
- Floating-point instructions
  - mov → fmov
  - add → fadd
  - sub → fsub
  - mul → fmul
  - div → fdiv
  - …
- you cannot use immediates in these floating-point instruction so you have to move a number into a FP register using fmov first.

Example

assembly code to divide 7.5 by 2.0 and print

- arguments for printf is set differently

```
// floating-pointer constant defined as external variables
        .data
x_m:    .double 0r7.5
y_m:    .double 0r2.0

        .text
fmt:    .string "%f divided by %f is %f\n"

        .balign 4
        .global main
main:   stp     x29, x30, [sp, -16]!
        mov     x29, sp

        // get x and set the 2nd arg of print
        adrp    x19, x_m
        add     x19, x19, :lo12:x_m
        ldr     d0, [x19]

        // get y and set the 3rd arg
        adrp    x19, y_m
        add     x19, x19, :lo12:y_m
        ldr     d1, [x19]

        // get the result and set it as the 4th arg
        fdiv    d2, d0, d1

        // set 1st arg and call printf
        adrp    x0, fmt
        add     x0, x0, :lo12:fmt
        bl      printf

end:    // ret
        ldp     x29, x30, [sp], 16
        ret
```

Code practice

- testread.asm
  - read input.bin in assignment6 and print

Code pratice

Reference

- lecture slides of Pro.Leonard
- https://www.dropbox.com/home/CPSC%20355?preview=Week+7.pptx