C and ARM Assembly Program

Peng-Sheng Chen

Fall, 2017

Outline

- ARM assembly program calls C function
- C function calls ARM assembly code

Use printf() Function

- R0: the address of the format
- R1: the number

```
printf(“number is %d\n”, 100);
```

```
        .data
format:
        .ascii "number is %d\n\000“

        .text
address_format:
        .word format
…
ldr    r0, address_format
mov    r1, #100
bl     printf
```

```
        .data
format:
        .ascii "number is %d\n\000"

        .text
address_format:
        .word format
main:
        mov     ip, sp
        stmfd   sp!, {fp, ip, lr, pc}
        sub     fp, ip, #4
        …
        ldr     r0, address_format
        mov     r1, #100
        bl      printf

        ldmea   fp, {fp, sp, pc}
```

遵守APCS規則

Use strcmp() Function

- R0: the address of the string1
- R1: the address of the string2

```
int result = strcmp("aa", "bb");
```

```
        .data
.str1:
        .ascii "aa\000"
.str2:
        .ascii "bb\000"

        .text
address_str1:
        .word .str1
address_str2:
        .word .str2
…
ldr   r0, address_str1
ldr   r1, address_str2
bl    strcmp
```

```
        .data
.str1:
        .ascii "aa\000"
.str2:
        .ascii "bb\000"
        .text
address_str1:
        .word .str1
address_str2:
        .word .str2
main:
        mov     ip, sp
        stmfd   sp!, {fp, ip, lr, pc}
        sub     fp, ip, #4
        …
        ldr     r0, address_str1
        ldr     r1, address_str2
        bl      strcmp

        ldmea   fp, {fp, sp, pc}
```

遵守APCS規則

其他Function如何使用？

- 寫一個test.c，其中使用了C function: func
- 使用arm-elf-gcc -S –O0 test.c產生ARM assembly code
- **觀察此assembly code**
- 得知如何使用 function func

C Program Calls ARM Assembly (1)

```
#include <stdio.h>
extern int square(int);

int main(void)
{
   int i;

   for (i=0; i<10; i++)
      printf("Square of %d is %d\n", i, square(i));

   return 0;
}
```

Argument1 => Register r0

遵守APCS規則

C Program Calls ARM Assembly (2)

square function

```
        .section .text
        .global square
        .type square,%function
square:
      mov     ip, sp
      stmfd   sp!, {fp, ip, lr, pc}
      sub     fp, ip, #4

      mul       r1, r0, r0
      mov       r0, r1

      ldmea   fp, {fp, sp, pc}
```

Get argument

Return value

C Program Calls ARM Assembly (3)

```
#include <stdio.h>
extern int* sort(int*, int);

int main(void)
{
    int array[2] = {1, 2};
    int* result;
    result = sort(array, 2);

    for (i=0; i<2; i ++)
        printf("%d\n", result[i]);

    return 0;
}
```

C Program Calls ARM Assembly (4)

```
        .section .text
        .global sort
        .type sort,%function
sort:
      mov    ip, sp
      stmfd  sp!, {fp, ip, lr, pc}
      sub    fp, ip, #4

      /* r0 <= the address of array */
      /* r1 <= the size of array    */

      /* do sorting */
      ...
      /* r0 <= the address of result array */
      ldmea  fp, {fp, sp, pc}
```

Program Compilation

- arm-elf-gcc -g -O0 call.c sort.s
- Use default’s link scripter
- -O0: 防止call.c太過簡單，而被compiler最佳化移除重要部分