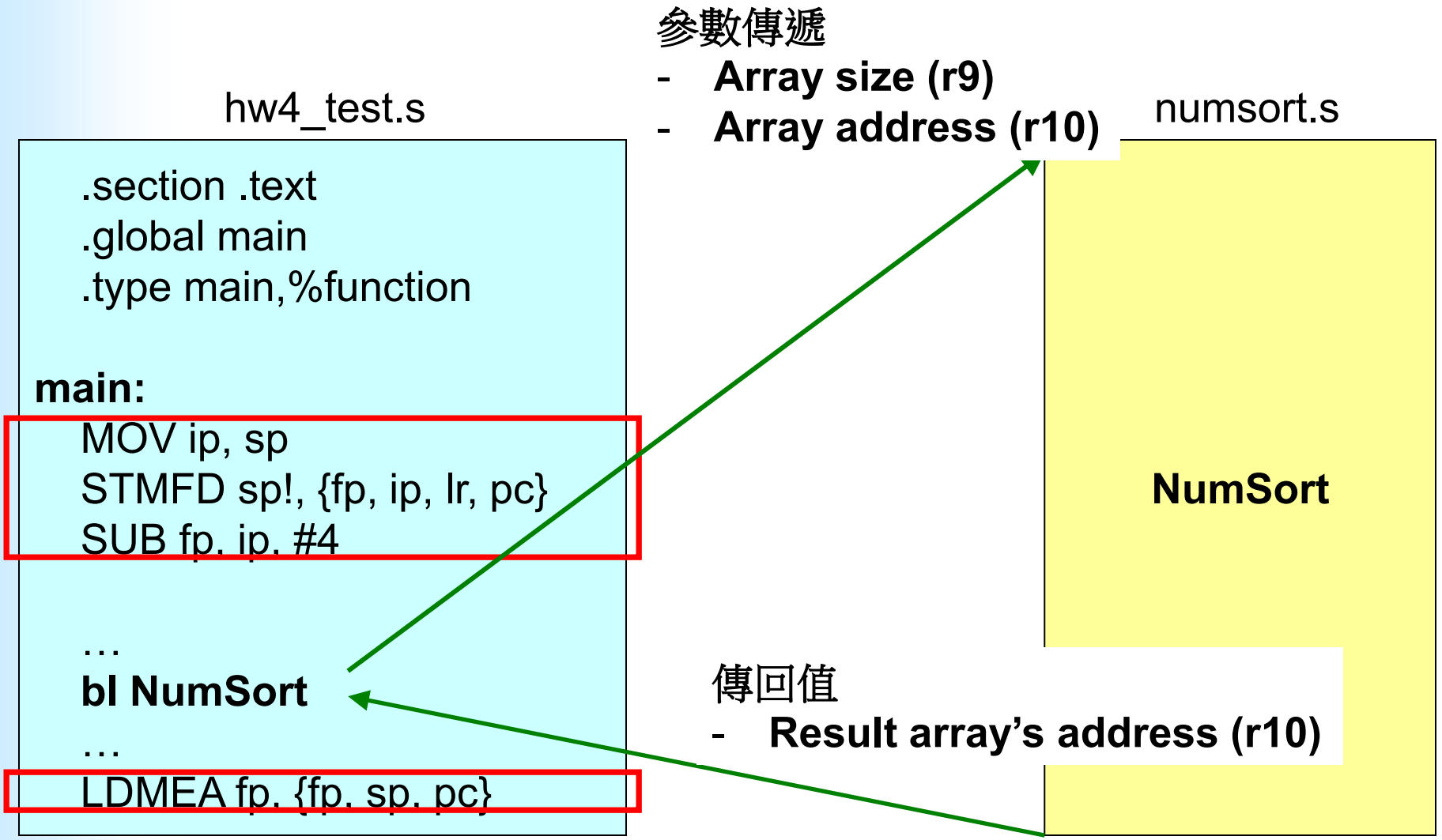
Homework #4 (1)

- Write a function called **NumSort** to sort an integer array from the smallest to the biggest.
- Two arguments will be passed into your function using registers
  - **Array size (r9)**
  - **The address of the first element in array (r10)**
- **The result of NumSort**
  - **Array size最大為100個elements。**
  - The result array in which each element is sorted from the smallest to the biggest. (原來的integer array裡的值沒有被修改，只是讀取原integer array，排序好的結果存放於**result array**)
  - Register r10 will have the address of the result array.

Homework #4 (2)

- Ex: an integer array=[1,10,6,3,20,40,9]
  - Result: **1, 3, 6, 9, 10, 20, 40**
- Ex: an integer array=[12,4,2,45,23,8,50,67]
  - Result: **2, 4, 8, 12, 23, 45, 50, 67**

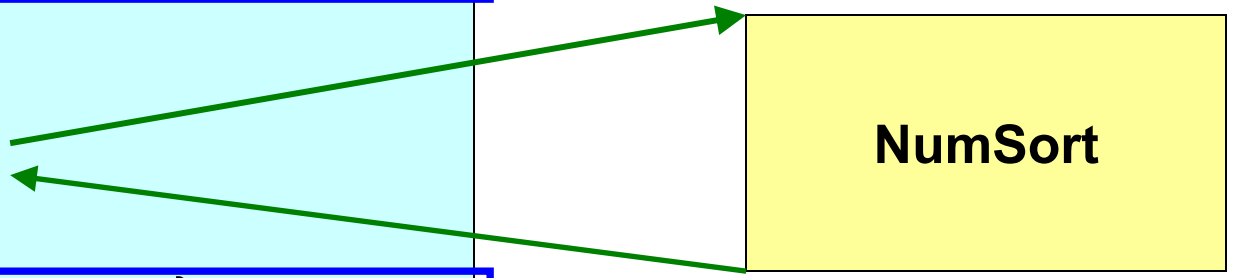

參數傳遞
- Array size (r9)
- Array address (r10)
hw4_test.s
numsort.s
.section .text
.global main
.type main,%function
main:
MOV ip, sp
STMFD sp!, {fp, ip, lr, pc}
SUB fp, ip, #4
…
bl NumSort
…
LDMEA fp, {fp, sp, pc}
NumSort
傳回值
- Result array's address (r10)

Homework #4 (3)

```
    .section .text
    .global main
    .type main,%function

main:
    MOV ip, sp
    STMFD sp!, {fp, ip, lr, pc}
    SUB fp, ip, #4

    …
    bl NumSort
    …
    LDMEA fp, {fp, sp, pc}
```

為了確保由NumSort() return後，main()仍可以正常執行，我們需要:

- 在執行NumSort()之前，把register的值暫存起來。
- 在執行完NumSort()之後，把暫存的值回復。
- 使用stack來儲存相關的值。



Stack

The mapping between the stack and block copy views of the multiple load and store instructions

定址方式	說明	POP	=LDM	PUSH	=STM
FA	遞增滿	LDMFA	LDMDA	STMFA	STMIB
FD	遞減滿	LDMFD	LDMIA	STMFD	STMDB
EA	遞增空	LDMEA	LDMDB	STMEA	STMIA
ED	遞減空	LDMED	LDMIB	STMED	STMDA

hw4_test.s

```
        .section .text
        .global main
        .type main,%function
main:
        MOV ip, sp
        STMFD sp!, {fp, ip, lr, pc}
        SUB fp, ip, #4

      /* prepare input array */
      …


        /* put array size into r9 */
        /* put array address into r10 */

        bl  NumSort
        /* --- end of your function --- */
        nop
        LDMEA fp, {fp, sp, pc}
        .end
```

- array size => r9
- array address => r10

- 當執行到nop時，r10的值為result array's address。

Homework #4 (4)

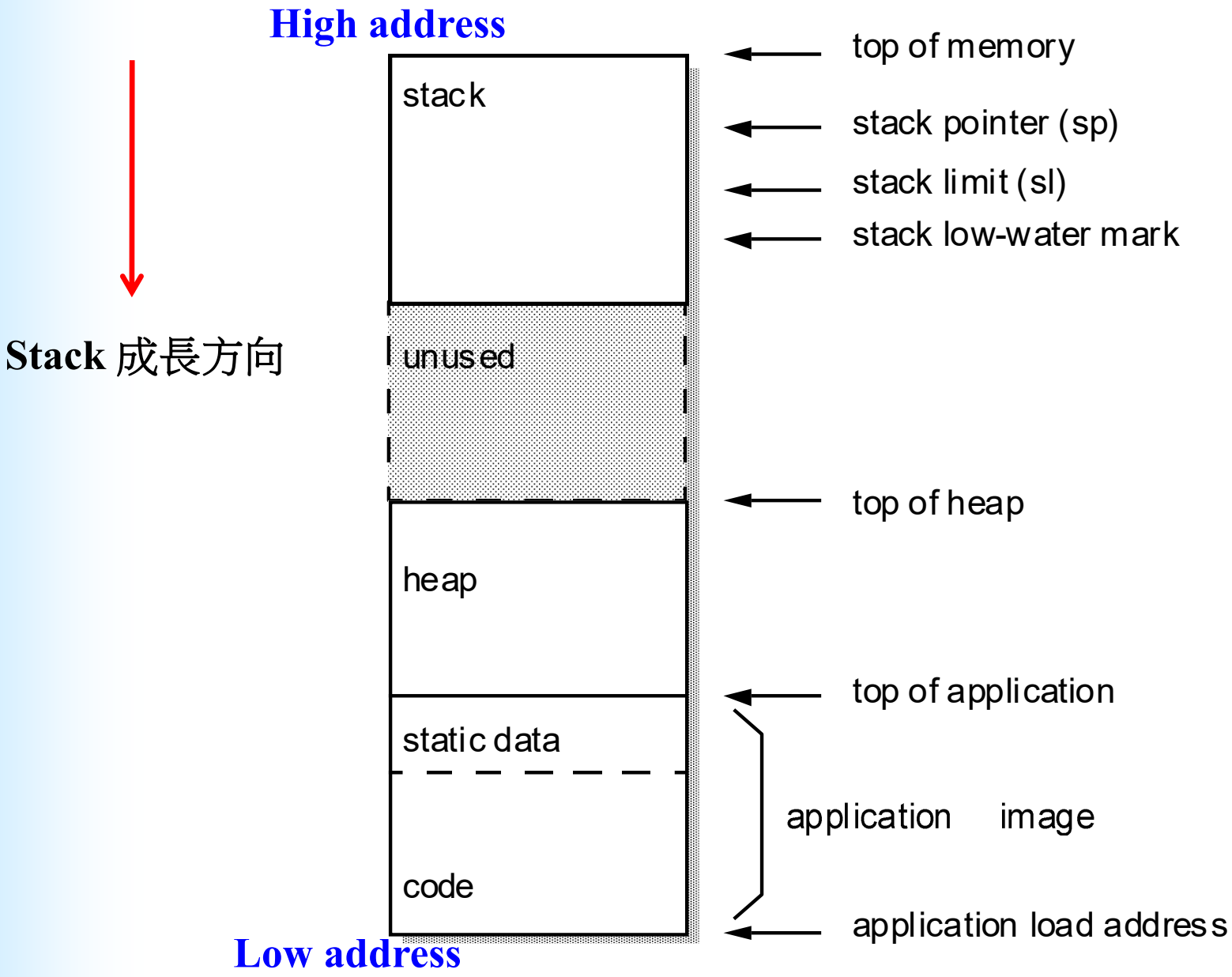
numsort.s

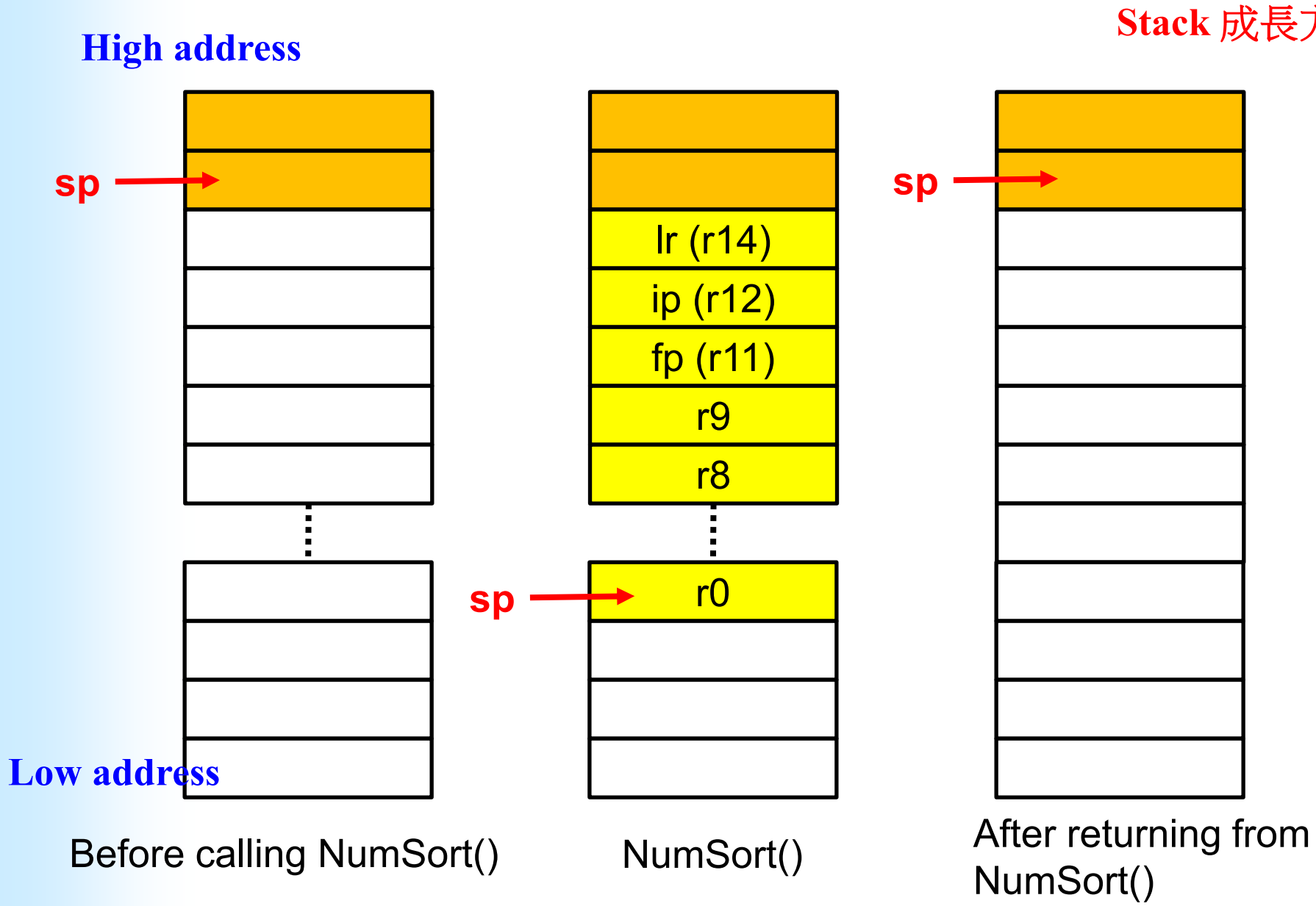
```
.section .text
.global NumSort
.type NumSort,%function

NumSort:
    /* function start */
    STMFD sp!, {r0-r9, fp, ip, lr}

    /* --- begin your function --- */
    /* Get array size from r9 */
    /* Get array address from r10 */

    /* DO NumSort */


    /* put result array's address into r10 */
    /* --- end of your function --- */
    /* function exit */
    LDMFD sp!, {r0-r9, fp, ip, pc}
    .end
```

參數傳遞

Write your function

傳回值


High address
Stack 成長方向
stack
unused
heap
static data
code
top of memory
stack pointer (sp)
stack limit (sl)
stack low-water mark
top of heap
top of application
application image
application load address
Low address


High address
Stack 成長方向
sp
lr (r14)
ip (r12)
fp (r11)
r9
r8
sp
r0
sp
Low address
Before calling NumSort()
NumSort()
After returning from NumSort()

How to Compile Your Program?

- $arm-none-eabi-gcc –g hw4_test.s numsort.s –o hw4.exe

Homework #4 (5)

- Program should be assembled and linked by gcc
  - 使用於作業一所安裝完成的cross toolchain.
- Program should be executed under **GDB ARM simulator**
- 程式中應有適當的說明（註解）
- You should turn in to **ECOURSE2**
  - "**README.txt**" file: 文字檔，描述你程式的內容、如何編譯程式、如何執行你的程式
  - Your ARM assembly procedure，**檔名為：numsort.s**
  - An ARM assembly program which uses your NumSort procedure，**檔名為：hw4_test.s**
  - Makefile / any file needed in your work
  - 請將欲繳交的檔案壓縮成 <**hw4_學號.tar.bz2**>，上傳壓縮檔
- **Deadline: November 29 (Monday), 2021**