ΕΠΛ221: Οργάνωση Υπολογιστών και Συμβολικός Προγραμματισμός

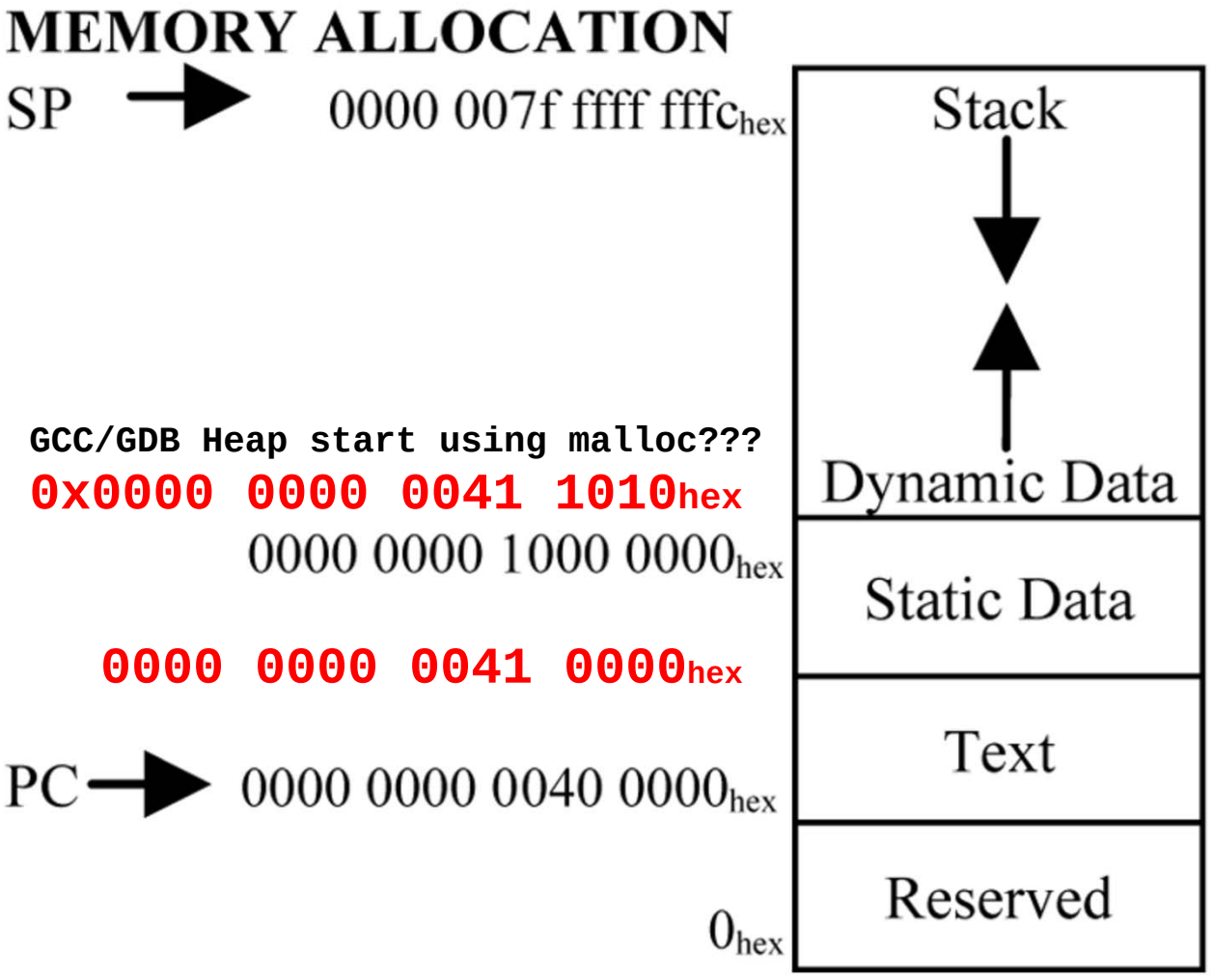
Εργαστήριο Αρ. 7

Εισαγωγή στην Αρχιτεκτονική **ARMv8**

Non-Leaf Functions/Recursion and fopen, fclose, fscanf, fprintf, feof, stdio, fgets

Πέτρος Παναγή, PhD

Memory Allocation LEGv8



Parameters in general-purpose registers

For the purposes of function calls, the general-purpose registers are divided into four groups:

Argument registers (X0-X7)

These are used to pass parameters to a function and to return a result. They can be used as scratch registers or as caller-saved register variables that can hold intermediate values within a function, between calls to other functions. The fact that 8 registers are available for passing parameters reduces the need to spill parameters to the stack when compared with AArch32.

Caller-saved temporary registers (X9-X15)

If the caller requires the values in any of these registers to be preserved across a call to another function, the caller must save the affected registers in its own stack frame. They can be modified by the called subroutine without the need to save and restore them before returning to the caller.

Callee-saved registers (X19-X29)

These registers are saved in the callee frame. They can be modified by the called subroutine as long as they are saved and restored before returning.

Integer Registers used for Instructions

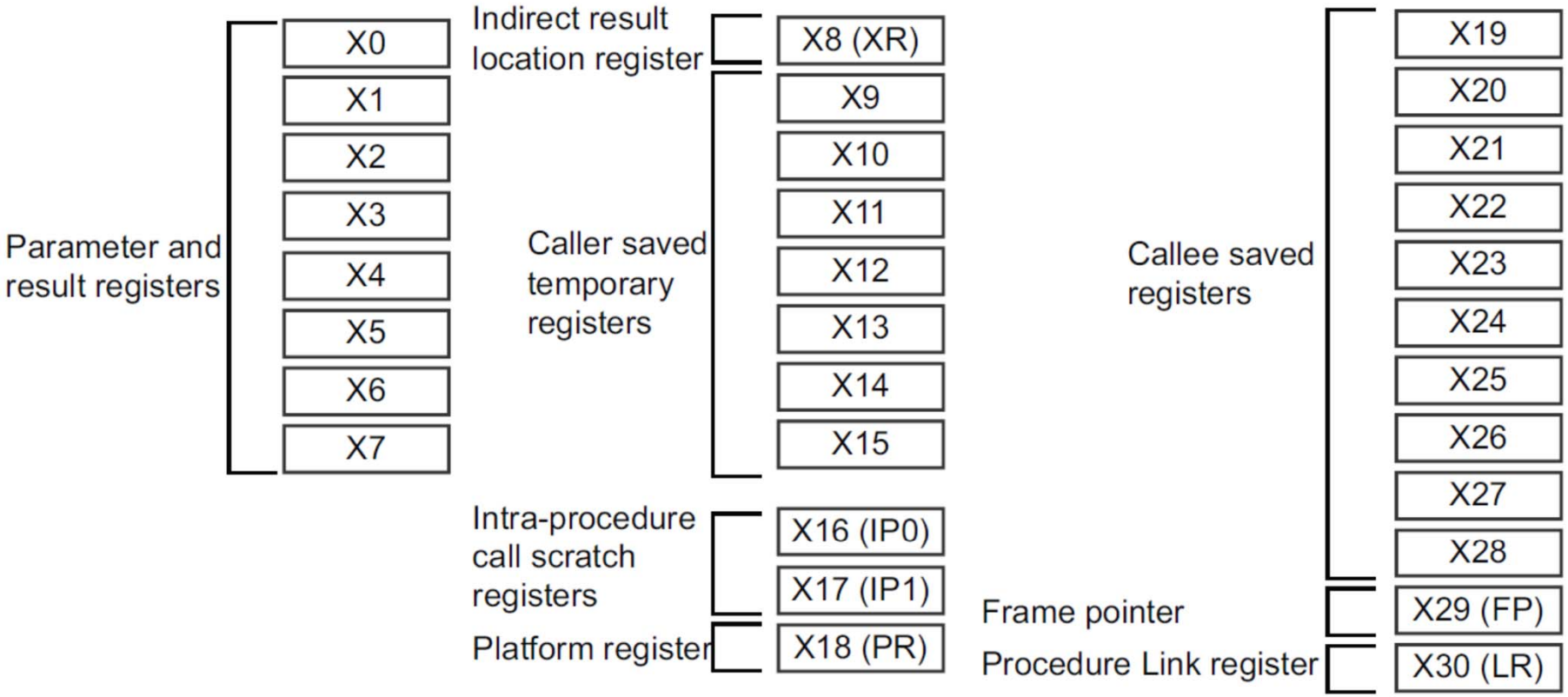


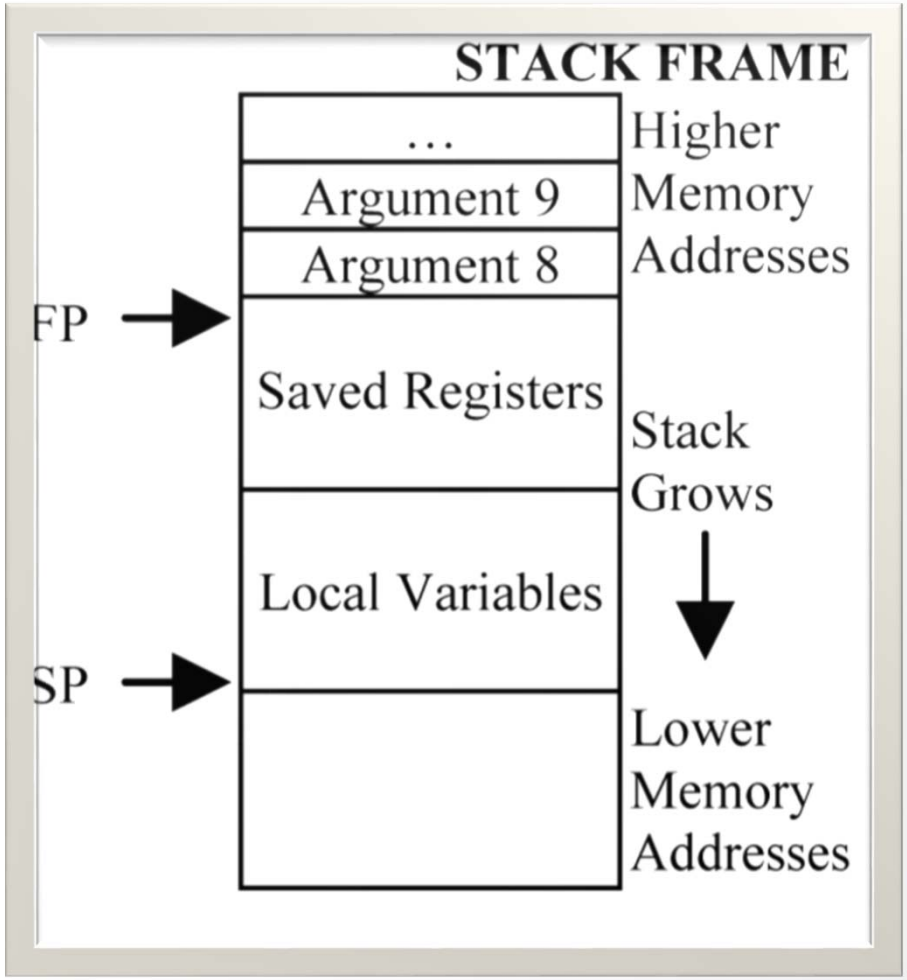
Figure 9-1 General-purpose register use in the ABI

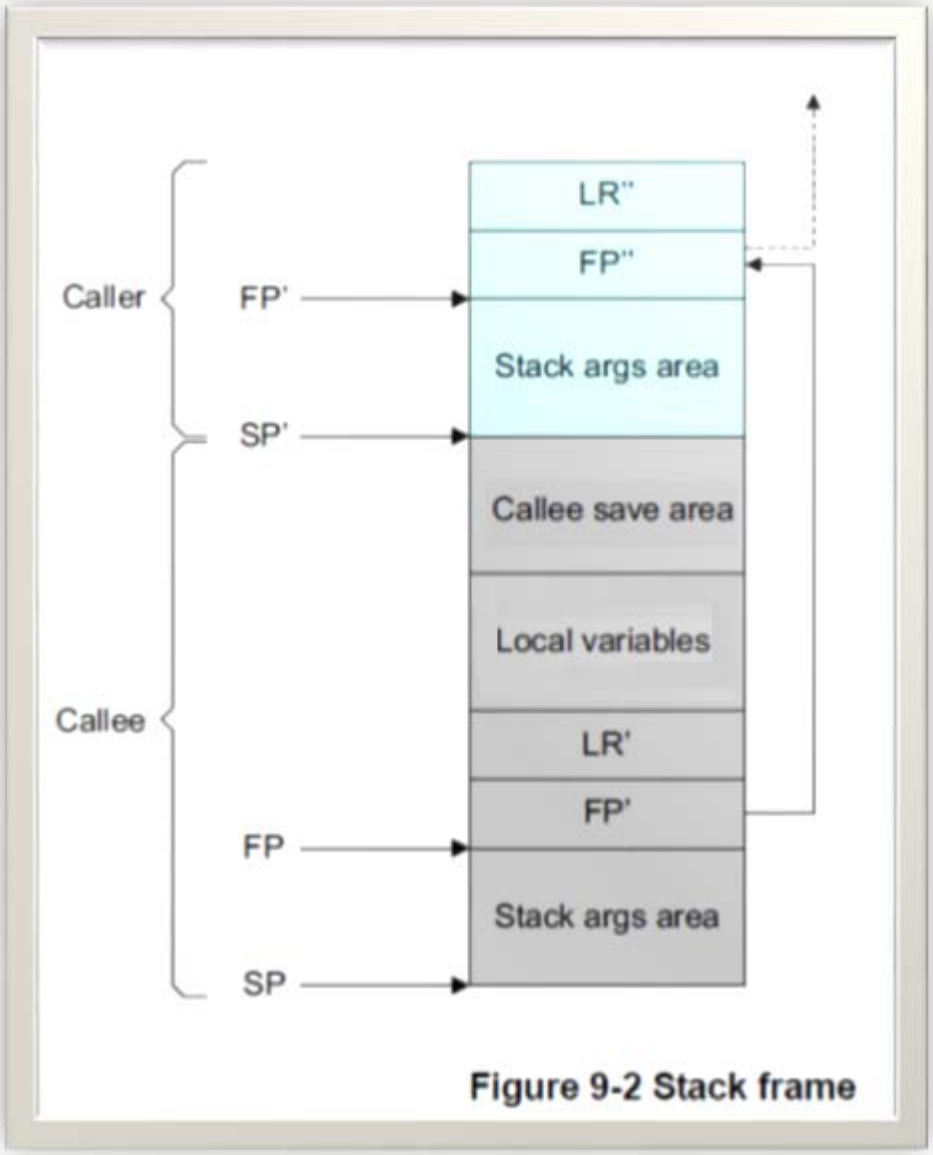
Integer Registers used for Instructions

Register	Special	Role in the procedure call standard
SP		The Stack Pointer.
r30	LR	The Link Register.
r29	FP	The Frame Pointer
r19…r28		Callee-saved registers
r18		The Platform Register, if needed; otherwise a temporary register. See notes.
r17	IP1	The second intra-procedure-call temporary register (can be used by call veneers and PLT code); at other times may be used as a temporary register.
r16	IP0	The first intra-procedure-call scratch register (can be used by call veneers and PLT code); at other times may be used as a temporary register.
r9…r15		Temporary registers
r8		Indirect result location register
r0…r7		Parameter/result registers

Table 2. General purpose registers and AAPCS64 usage

Stack Frame Layout





Stack Frame Layout

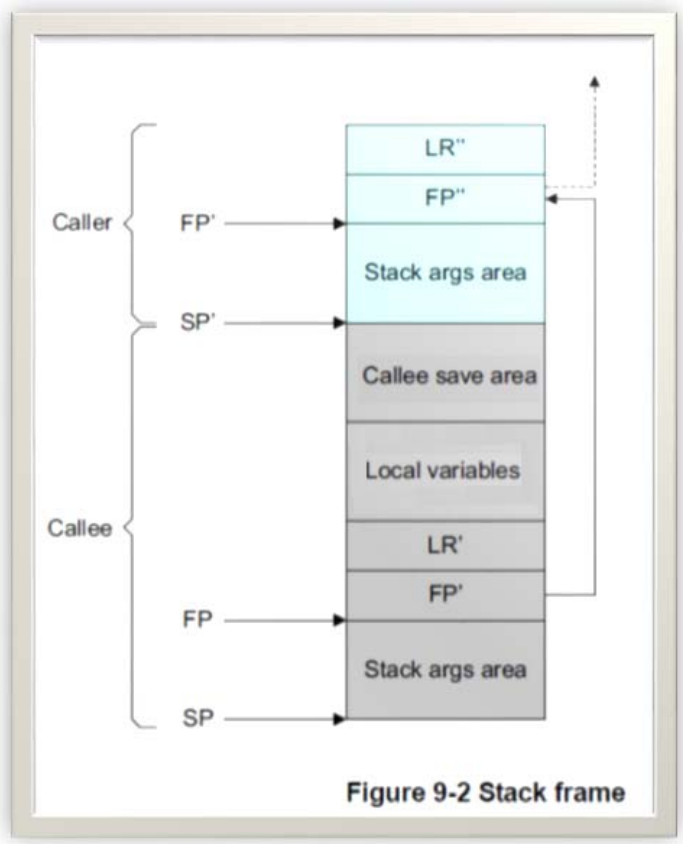


Figure 9-2 Stack frame

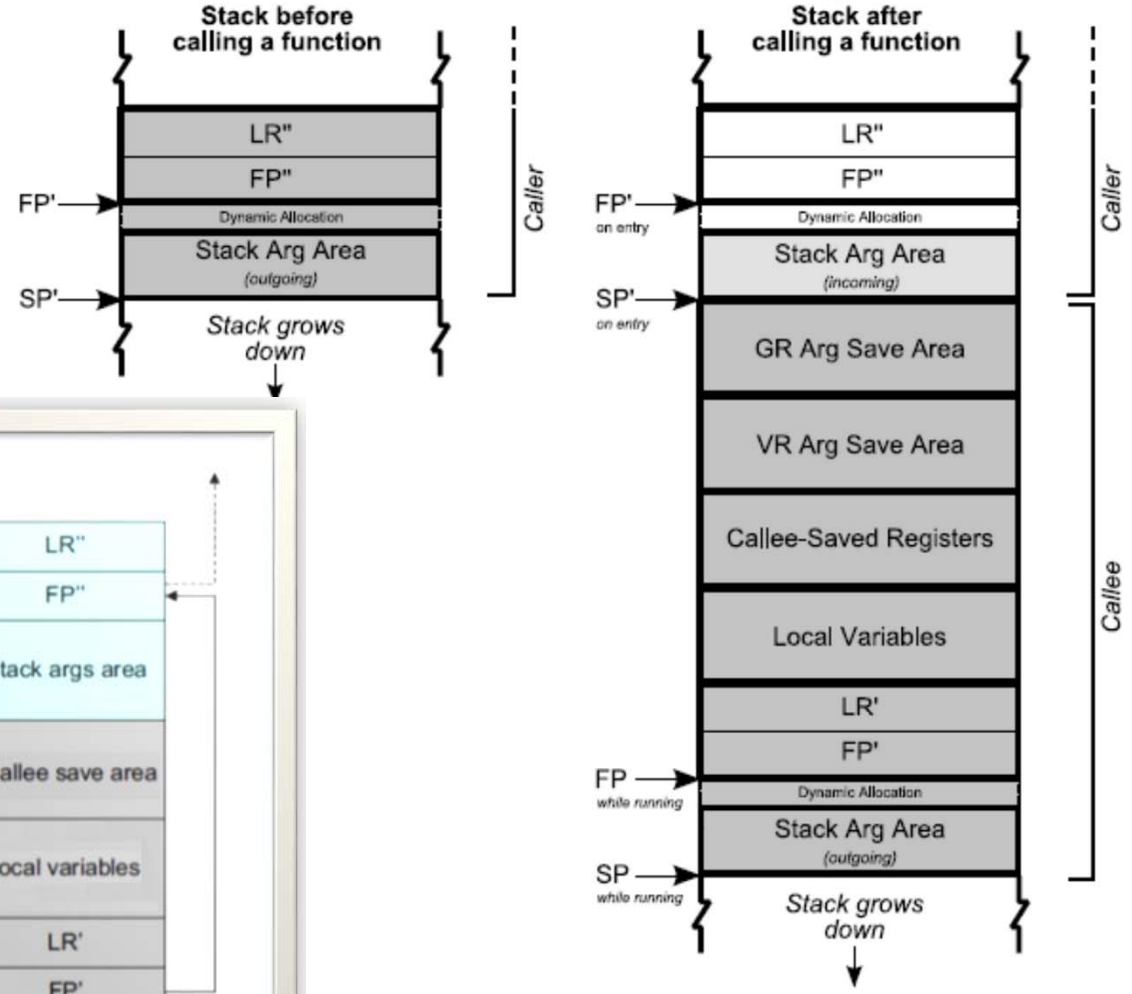


Figure 3, Example stack frame layout

MIPS Recursion Example - Factorial

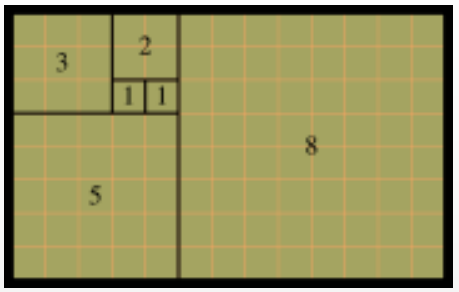
```
main ()
{
    printf ("The factorial of 10 is %d\n", fact (10));
}

int fact (int n)
{
    if (n < 1)
       return (1);
    else
       return (n * fact (n - 1));
}
```

MIPS Αναδρομή Εργαστηριακή Άσκηση

Αριθμοί Fibonacci:

$$F(n) := \begin{cases} 0 & \text{if } n = 0; \\ 1 & \text{if } n = 1; \\ F(n-1) + F(n-2) & \text{if } n > 1. \end{cases}$$



Να υπολογίσετε τον κάθε αριθμό (n) Fibonacci που δίνετε στο πρόγραμμα σας.
0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, 4181, 6765, 10946, 17711, 28657, 46368, 75025, 121393, .

MIPS Αναδρομή Εργαστηριακή Άσκηση

$$F(n) := \begin{cases} 0 & \text{if } n = 0; \\ 1 & \text{if } n = 1; \\ F(n-1) + F(n-2) & \text{if } n > 1. \end{cases}$$

Initial Moment	None
First	1
Second	1
Third	2
Fourth	3
Fifth	5
Sixth	8
Seventh	13
Eighth	21
Ninth	34
Tenth	55
Eleventh	89
Twelfth	144

Exercises

Recursive Factorial and Fibonacci Examples:

`Lab7_exercise1.s`

Exercises

Working with Files:

```
Lab7_exercise2.s
Lab7_exercise3.s
Lab7_exercise4.s
```

http://www.cplusplus.com/reference/cstdio/fopen/
http://www.cplusplus.com/reference/cstdio/fscanf/
http://www.cplusplus.com/reference/cstdio/fprintf/
http://www.cplusplus.com/reference/cstdio/fclose/
http://www.cplusplus.com/reference/cstdio/feof/