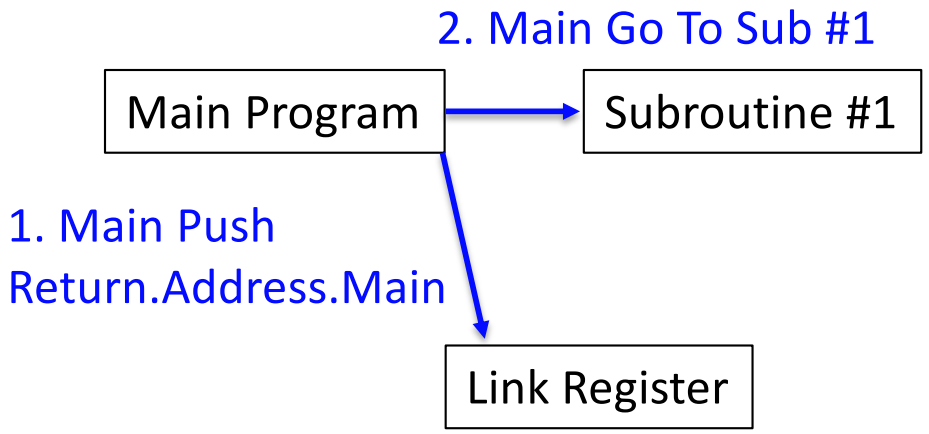
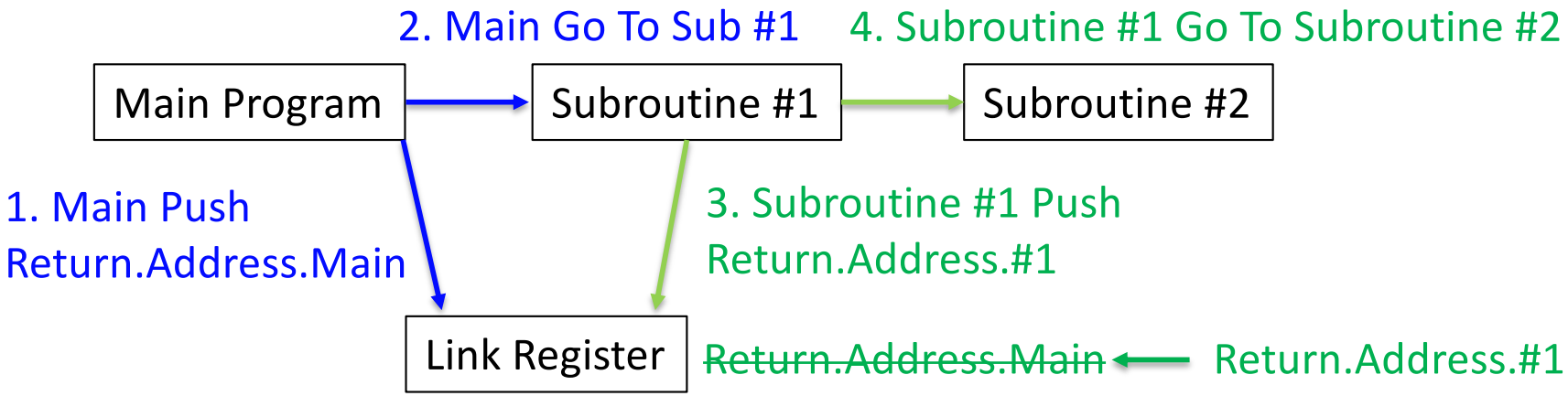
2.7.1 Subroutine Nesting

- **Subroutine nesting:**
  - A subroutine is allowed to call another subroutine
- Link register
  - Has next address of first call
  - Will be overwritten in subsequent calls



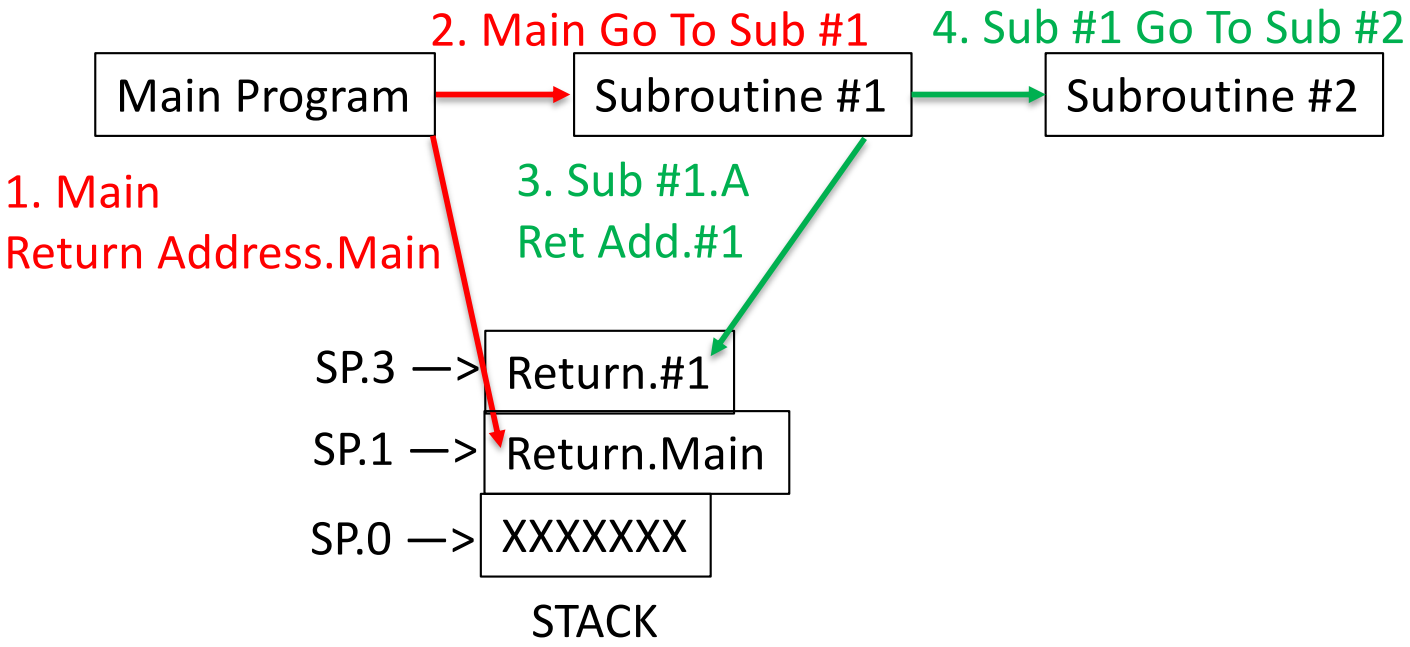
Nesting Problem



- Do not use the Link Register alone
- Use Stack Frame
- Or Address Stack
- With Link Register containing the latest return address

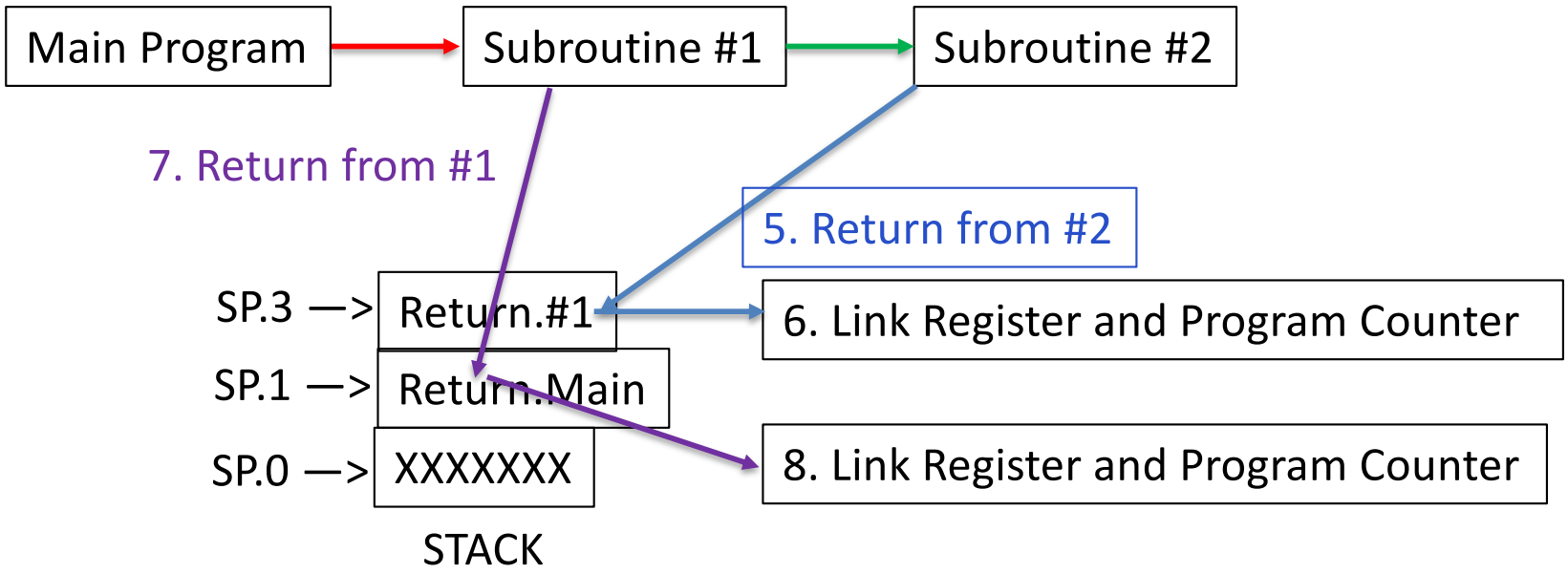
Nesting Stack: Calls

- Save return address on stack before call
- Restore link register (pop from stack) after return from subroutine



Nesting Stack: Returns

- Saves return address on stack before call
- Restores link register (pop from stack) after return from subroutine



Thumb Push and Pop Instructions

5.14.1 Operation

The instructions in this group allow registers 0-7 and optionally LR to be pushed onto the stack, and registers 0-7 and optionally PC to be popped off the stack.

The THUMB assembler syntax is shown in ◔*Table 5-15: PUSH and POP instructions*.

Note The stack is always assumed to be Full Descending.

L	R	THUMB assembler	ARM equivalent	Action
0	0	PUSH { Rlist }	STMDB R13!, { Rlist }	Push the registers specified by Rlist onto the stack. Update the stack pointer.
0	1	PUSH { Rlist, LR }	STMDB R13!, { Rlist, R14 }	Push the Link Register and the registers specified by Rlist (if any) onto the stack. Update the stack pointer.
1	0	POP { Rlist }	LDMIA R13!, { Rlist }	Pop values off the stack into the registers specified by Rlist. Update the stack pointer.
1	1	POP { Rlist, PC }	LDMIA R13!, { Rlist, R15 }	Pop values off the stack and load into the registers specified by Rlist. Pop the PC off the stack. Update the stack pointer.

Table 5-15: PUSH and POP instructions

Thumb Push Details

THUMB assembler	ARM equivalent	Action
PUSH { Rlist }	STMDB R13!, { Rlist }	Push the registers specified by Rlist onto the stack. Update the stack pointer.
PUSH { Rlist, LR }	STMDB R13!, { Rlist, R14 }	Push the Link Register and the registers specified by Rlist (if any) onto the stack. Update the stack pointer.

Thumb Pop Details

THUMB assembler	ARM equivalent	Action
POP { Rlist }	LDMIA R13!, { Rlist }	Pop values off the stack into the registers specified by Rlist. Update the stack pointer.
POP { Rlist, PC }	LDMIA R13!, { Rlist, R15 }	Pop values off the stack and load into the registers specified by Rlist. Pop the PC off the stack. Update the stack pointer.

Thumb Push then Pop Details

```
PUSH   {R0-R4,LR}      ; Store R0,R1,R2,R3,R4 and R14 (LR) at
                       ; the stack pointed to by R13 (SP) and
                       ; update R13.
                       ; Useful at start of a sub-routine to
                       ; save workspace and return address.

POP    {R2,R6,PC}      ; Load R2,R6 and R15 (PC) from the stack
                       ; pointed to by R13 (SP) and update R13.
                       ; Useful to restore workspace and return
                       ; from sub-routine.
```

Example: Nested Subroutines Program

Main program

```
                    :
2000        Move        PARAM2,–(SP)    Place parameterson stack.
2004        Move        PARAM1,–(SP)
2008        Call        SUB1
2012        Move        (SP),RESUIT     Store result.
2016        Add         #8,SP           Restorestack level.
2020        next instruction
                    :
```

Nested Subroutines Sub-1

```
2100   SUB1   Move          FP,–(SP)        Save frame pointer register.
2104          Move          SP,FP           Load the frame pointer.
2108          MoveMultiple  R0–R3,–(SP)     Save registers.
2112          Move          8(FP),R0        Get first parameter.
              Move          12(FP),R1       Get secondparameter.
              :
              Move          PARAM3,–(SP)    Placea parameteron stack.
2160          Call          SUB2
2164          Move          (SP)+,R2        Pop SUB2 result into R2.
              :
              Move          R3,8(FP)        Placeanseweron stack.
              MoveMultiple  (SP)+,R0–R3     Restoreregisters.
              Move          (SP)+,FP        Restoreframe pointer register.
              Return                        Return to Main program.
```

Nested Subroutines Sub-2

```
3000   SUB2   Move          FP,–(SP)      Save frame pointer register.
              Move          SP,FP         Load the frame pointer.
              MoveMultiple  R0–R1,–(SP)   Save registers R0 and R1.
              Move          8(FP),R0      Get the parameter.
              .
              .
              .
              Move          R1,8(FP)      Place SUB2 result on stack.
              MoveMultiple  (SP)+,R0–R1   Restore registers R0 and R1.
              Move          (SP)+,FP      Restore frame pointer register.
              Return                      Return to Subroutine 1.
```

Nested Subroutines Stack Frames

Assignment: indicate where the SP is pointing to after the execution of each instruction