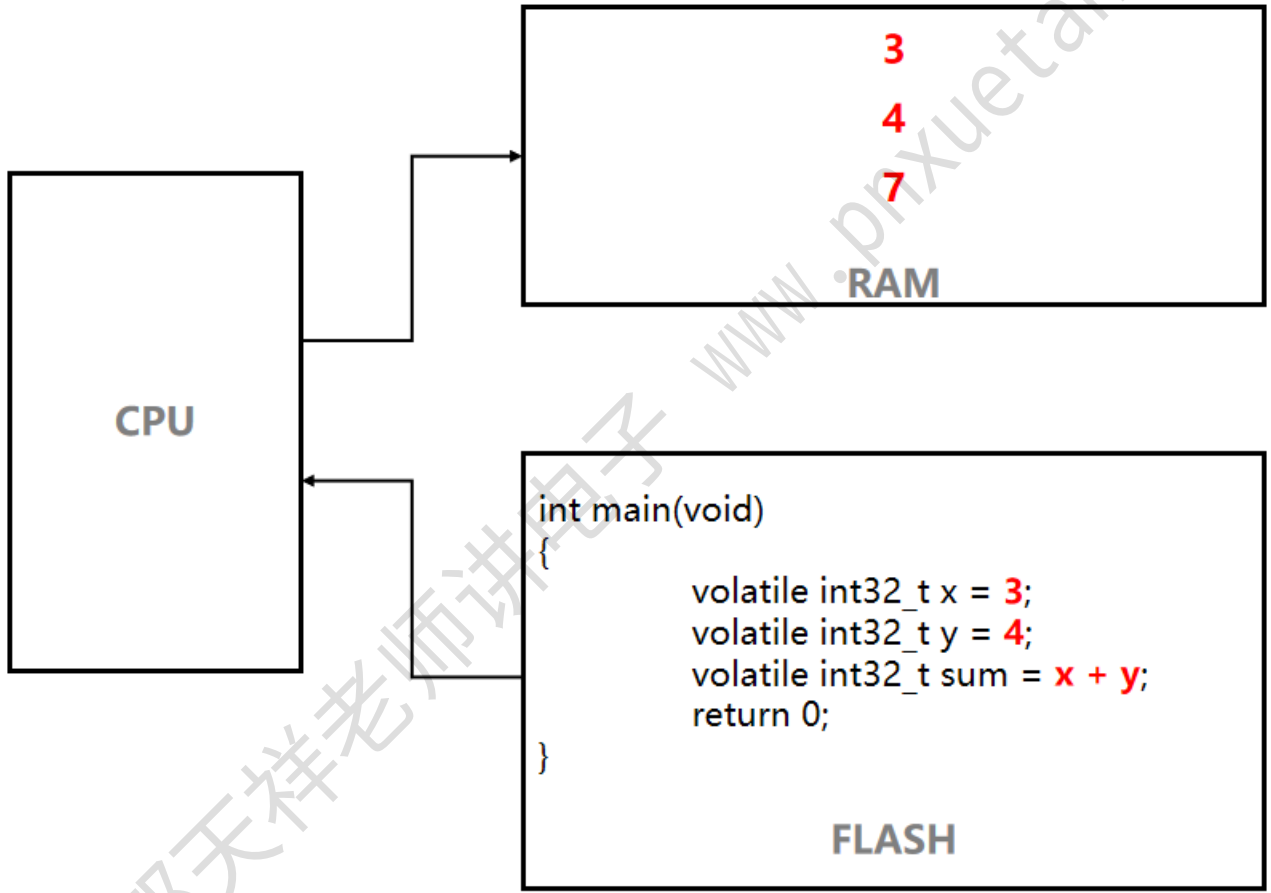
嵌入式C语言之-

局部变量赋值及栈的工作原理

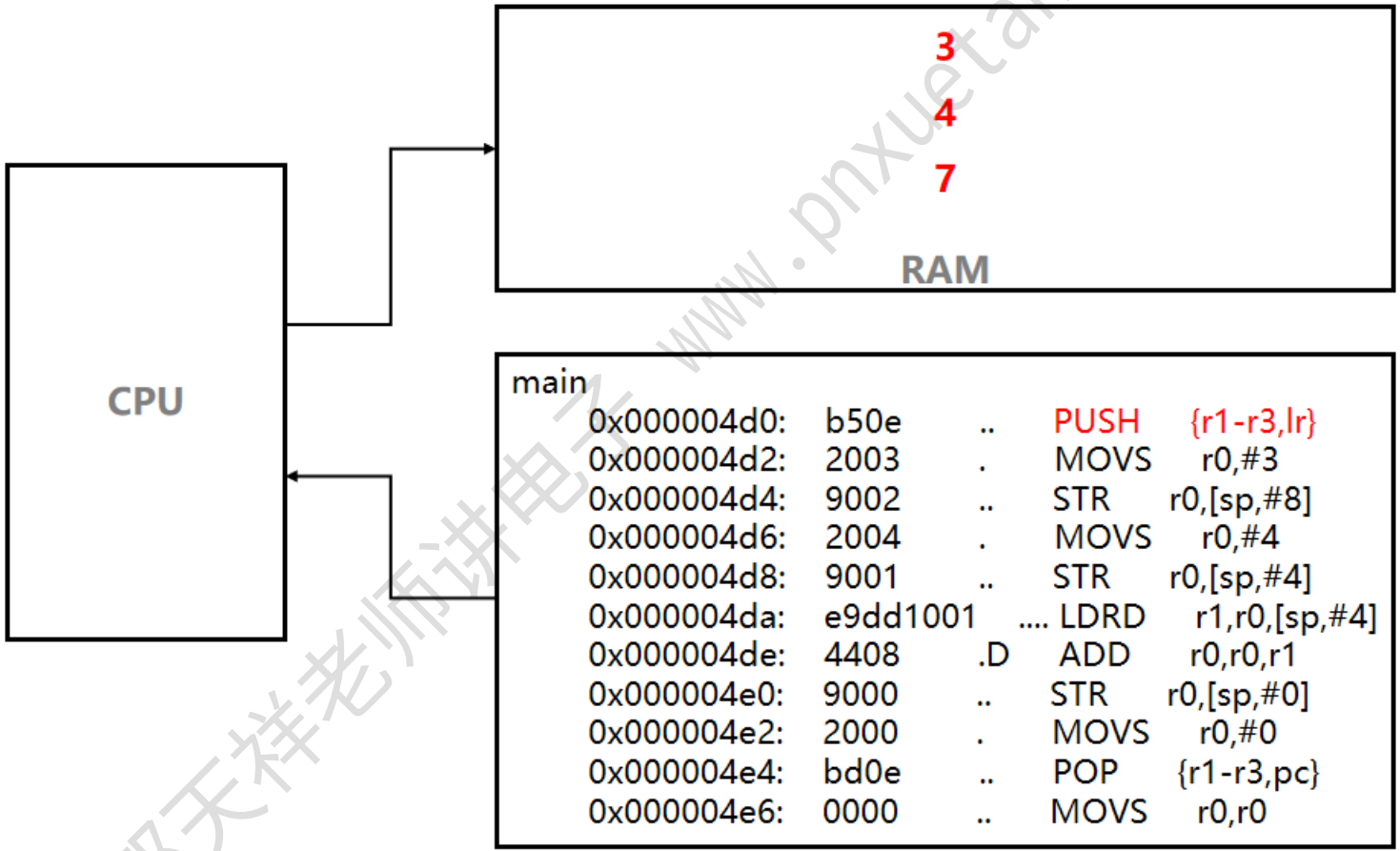
讲师：叶大鹏

助力你成为优秀的电子工程师！

局部变量的数值是如何存储到RAM中的?

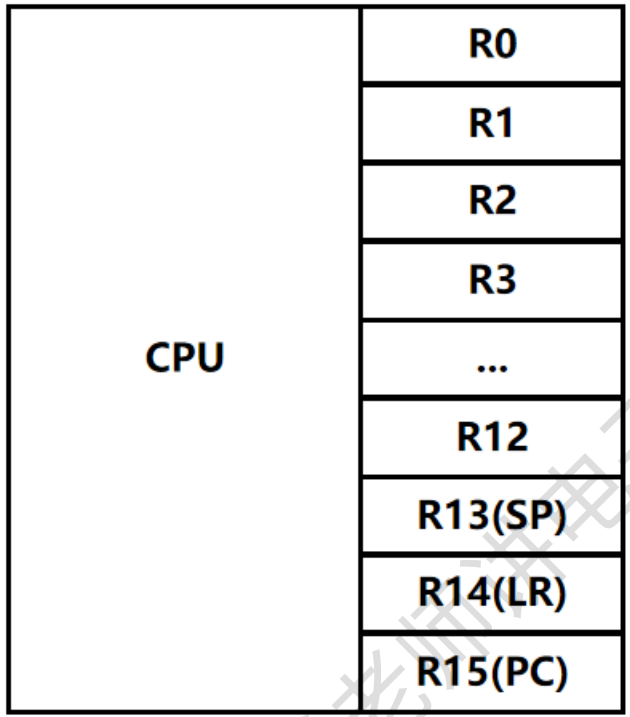


局部变量的数值是如何存储到RAM中的?



PN学堂

PUSH {r1-r3, lr}



- ARM 32单片机内核有16个寄存器，其中R0-R12这13个为通用目的寄存器，剩下的R13(SP)寄存器用来保存栈地址，R14(LR)寄存器用来保存函数的返回地址，R15(PC)寄存器用来保存程序当前指令的地址，这些寄存器都是32位的；
- 简单的说，就是用来保存程序运行和数据运算过程中的数据和地址。

PUSH {r1-r3, lr}

- **这条指令的作用，目的是为main函数的局部变量分配栈空间。**

◆ **PUSH 和 POP**

- 寄存器入栈指令,实现R0~R7寄存器和可选的 LR 寄存器入栈;
- 寄存器出栈指令,实现R0~R7寄存器和可选的 PC寄存器出栈;
- 堆栈地址由 SP 寄存设置,入栈时地址递减，出栈时地址递增。
- 指令格式如下:

PUSH {reglist [,LR]}

POP {reglist [,PC]}

其中 reglist 入栈/出栈寄存器列表,即 R0~R7

LR 入栈时的可选寄存器

PC 出栈时的可选寄存器

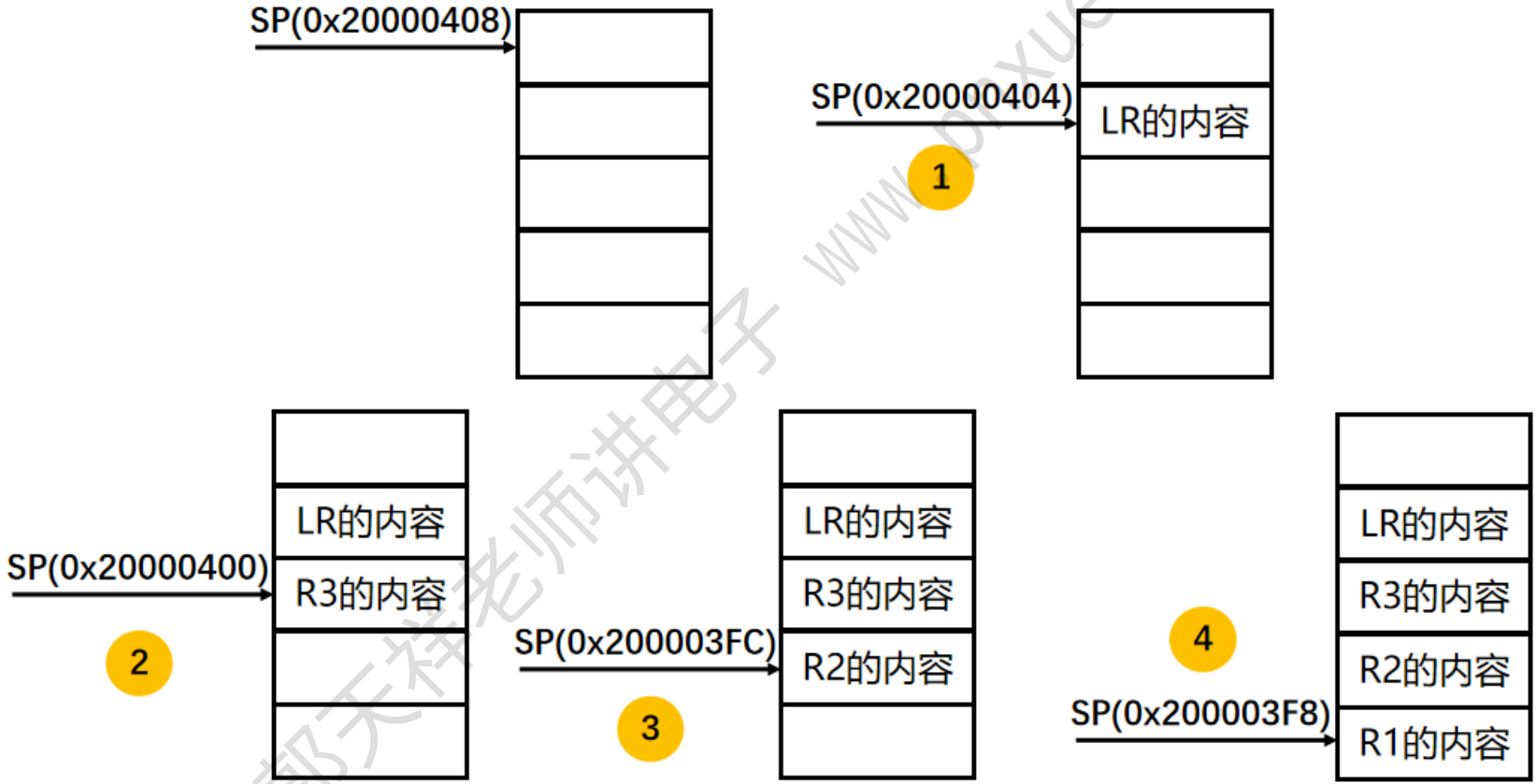
寄存器入栈及出栈指令举例如下:

PUSH {R0-R7,LR} ;将寄存器 R0~R7 全部入栈,LR 也入栈

POP {R0-R7,PC} ;将堆栈中的数据弹出到寄存器 R0~R7 及 PC 中

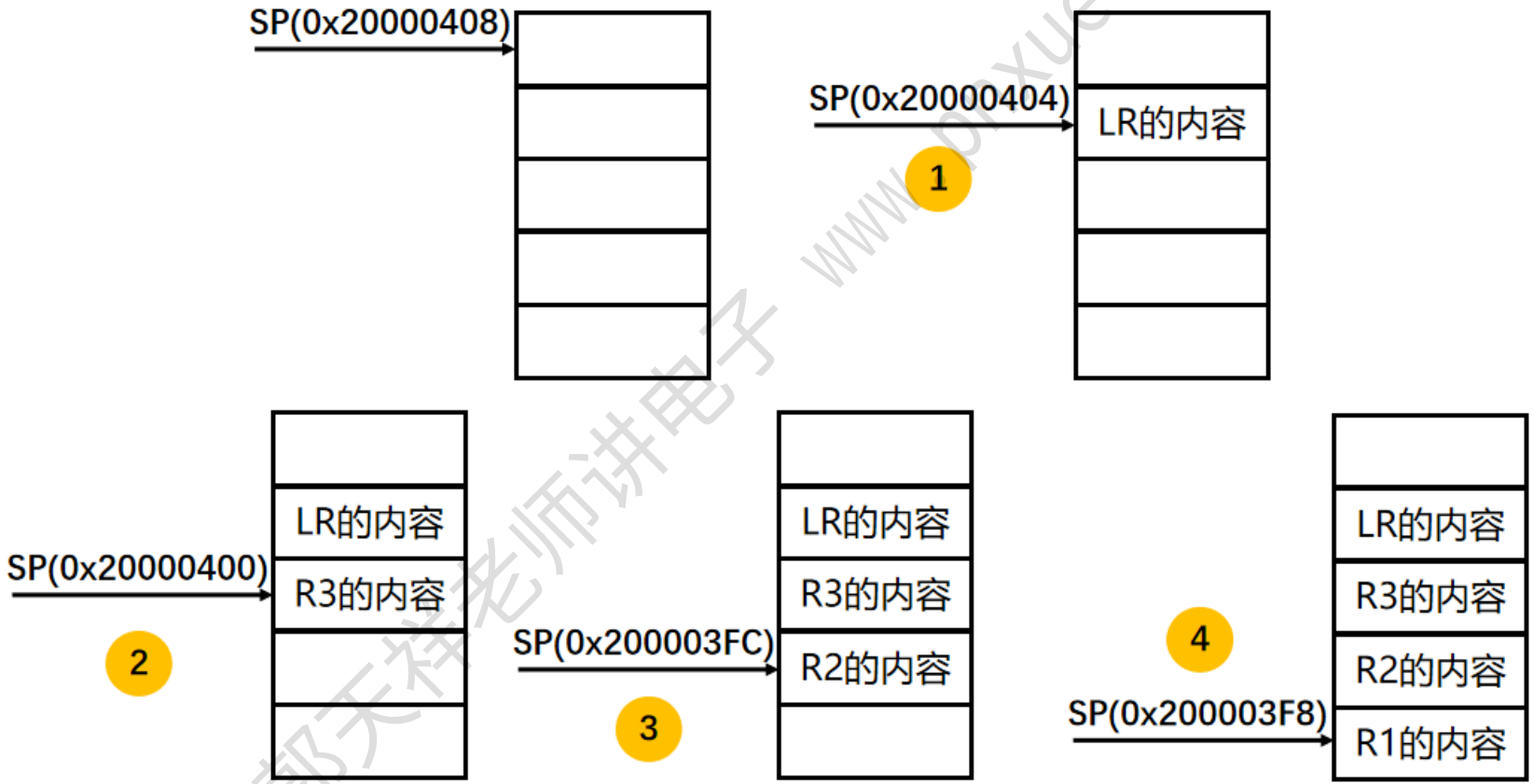
PUSH {r1-r3, lr}

- 如果程序在进入main函数时，sp保存的内存地址为0x20000408

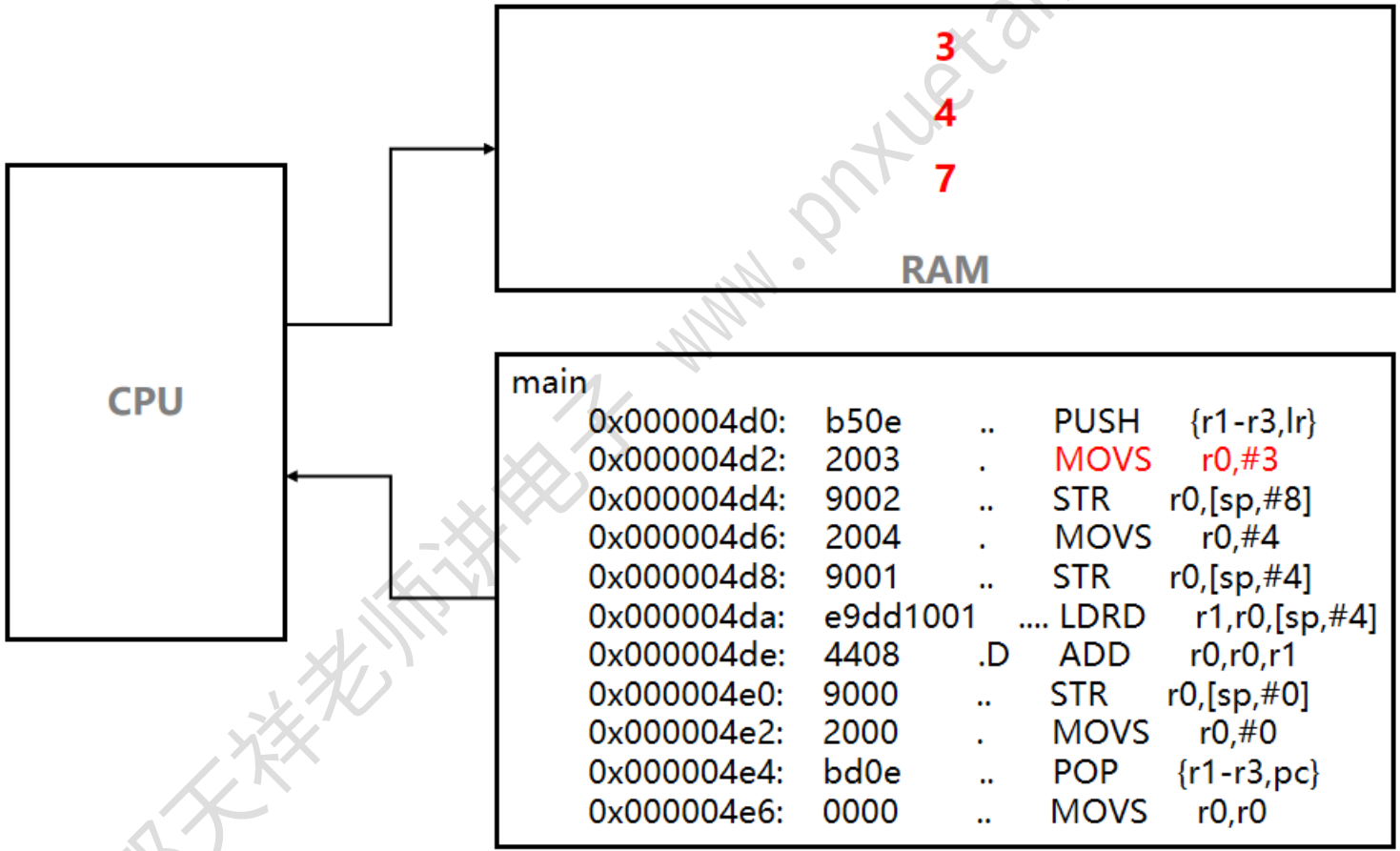


PUSH {r1-r3, lr}

- 实际上，这条指令的目的并不是为了保存R1-R3和LR的内容，只是为了给main函数在栈空间占4个坑。

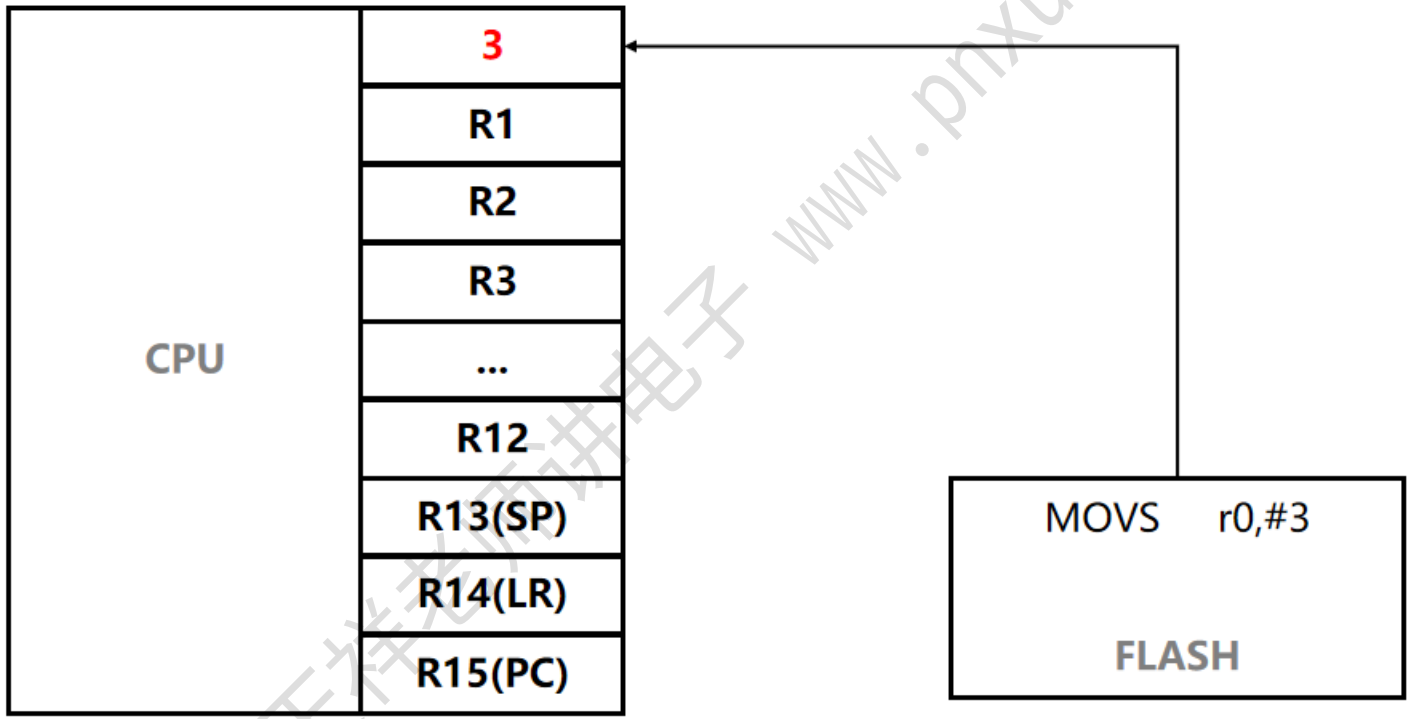


局部变量的数值是如何存储到RAM中的?



MOVS r0,#3

- 这条指令的作用是将立即数3保存在R0寄存器中。



MOVS r0,#3

◆ **MOV**

- 数据传送指令，将8位立即数或寄存器(operant2)传送到目标寄存器Rd,可用于移位运算等操作。
- 指令格式如下:

MOV{cond}{S} Rd,operand2

- MOV 指令举例如下:

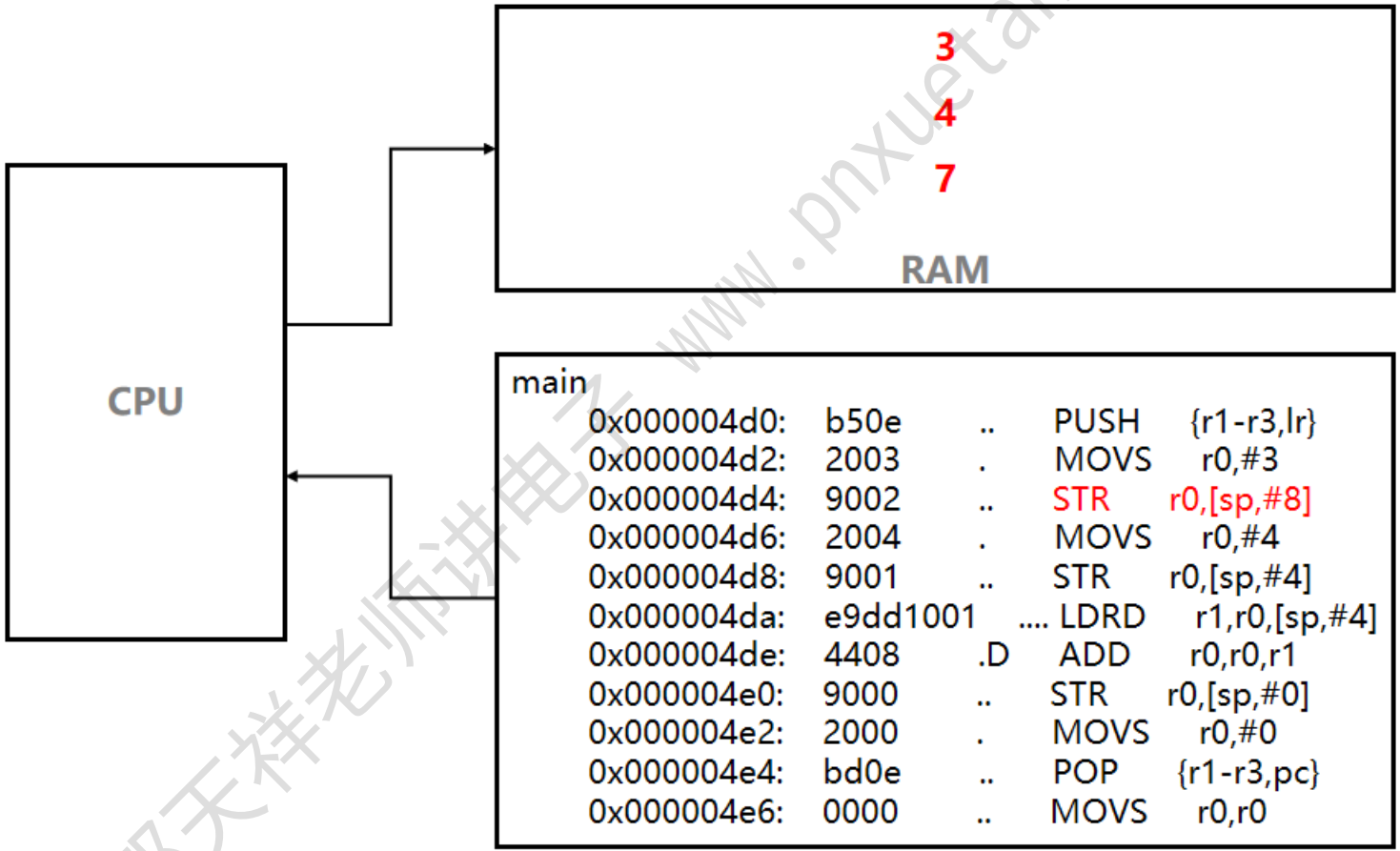
MOV R1,#0x10 ;R1=0x10

MOV R0,R1 ;R0=R1

MOVS R3,R1,LSL #2 ;R3=R1 < < 2,并影响标志位

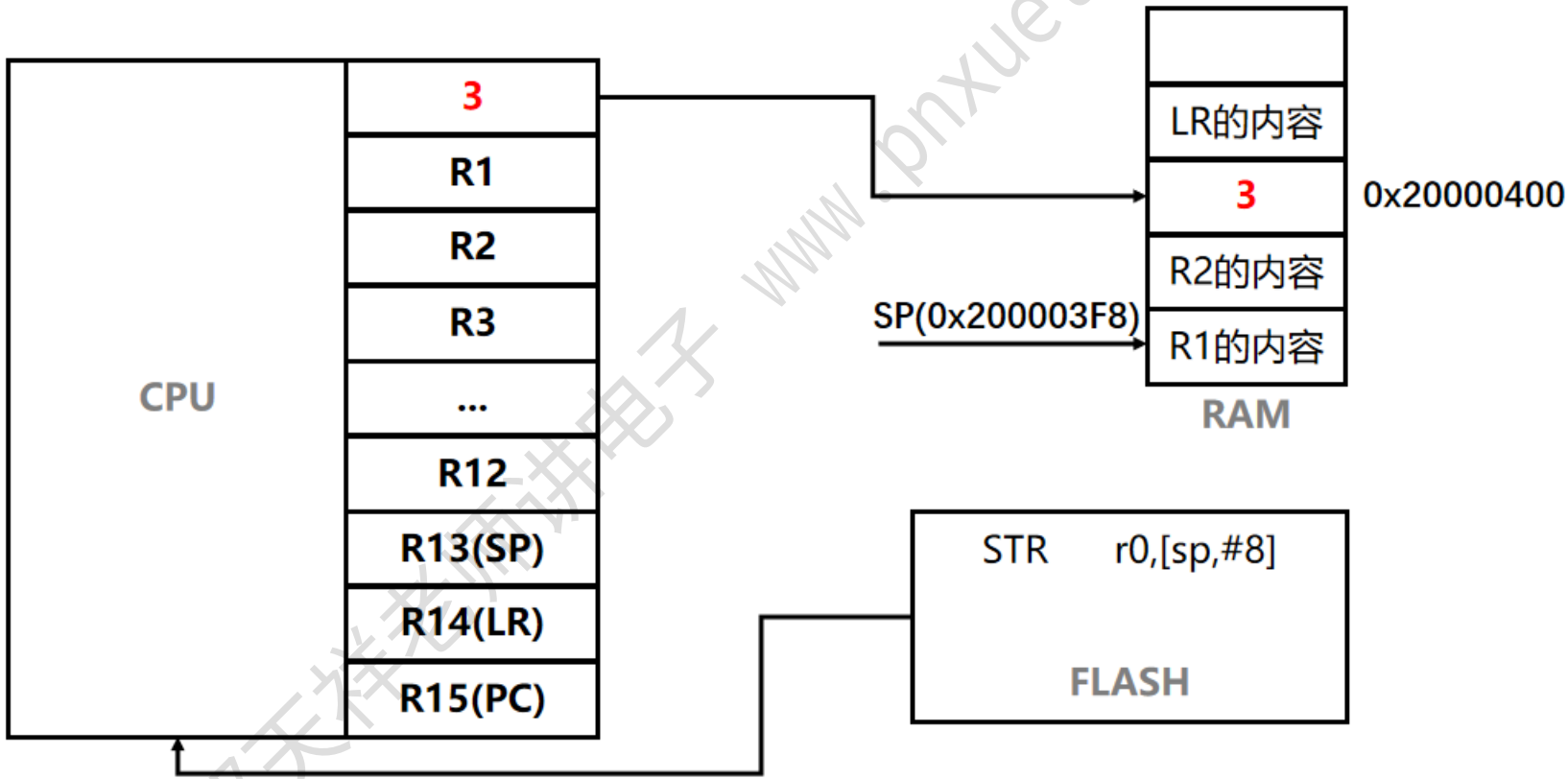
MOV PC,LR ;PC=LR ,子程序返回

局部变量的数值是如何存储到RAM中的?



STR r0,[sp,#8]

- 这条指令的作用是将立即数3保存在地址为0x20000400的栈中。



STR r0,[sp,#8]

◆ **LDR和STR**

- LDR 指令用于从内存中读取数据放入寄存器中;
- STR指令用于将寄存器中的数据保存到内存;
- 指令格式如下:

LDR{cond}{T} Rd,<地址>;加载指定地址上的4字节数据,放入Rd中

STR{cond}{T} Rd,<地址>;存储4字节数据到指定地址的存储单元,要存储的数据在Rd中

LDR{cond}B{T} Rd,<地址>;加载1字节数据,放入Rd中,即 Rd 最低字节有效,高 24 位清零

STR{cond}B{T} Rd,<地址>;存储1字节数据,要存储的数据在Rd,最低字节有效

STR r0,[sp,#8]

- LDR/STR 指令寻址是非常灵活的,由两部分组成,一部分为一个基址寄存器,可以为任一个通用寄存器,另一部分为一个地址偏移量.地址偏移量有以下 3 种格式:

(1) 立即数.立即数可以是一个无符号数值,这个数据可以加到基址寄存器,也可以从基址寄存器中减去这个数值.指令举例如下:

LDR R1,[R0,#0x12] ;将 R0+0x12 地址处的数据读出,保存到 R1 中(R0 的值不变)

LDR R1,[R0,#-0x12];将 R0-0x12 地址处的数据读出,保存到 R1 中(R0 的值不变)

LDR R1,[R0] ;将 R0 地址处的数据读出,保存到 R1 中(零偏移)

(2)寄存器.寄存器中的数值可以加到基址寄存器,也可以从基址寄存器中减去这个数值.指令举例值.指令举例如下:

LDR R1,[R0,R2] ;将 R0+R2 地址的数据计读出,保存到 R1 中(R0 的值不变)

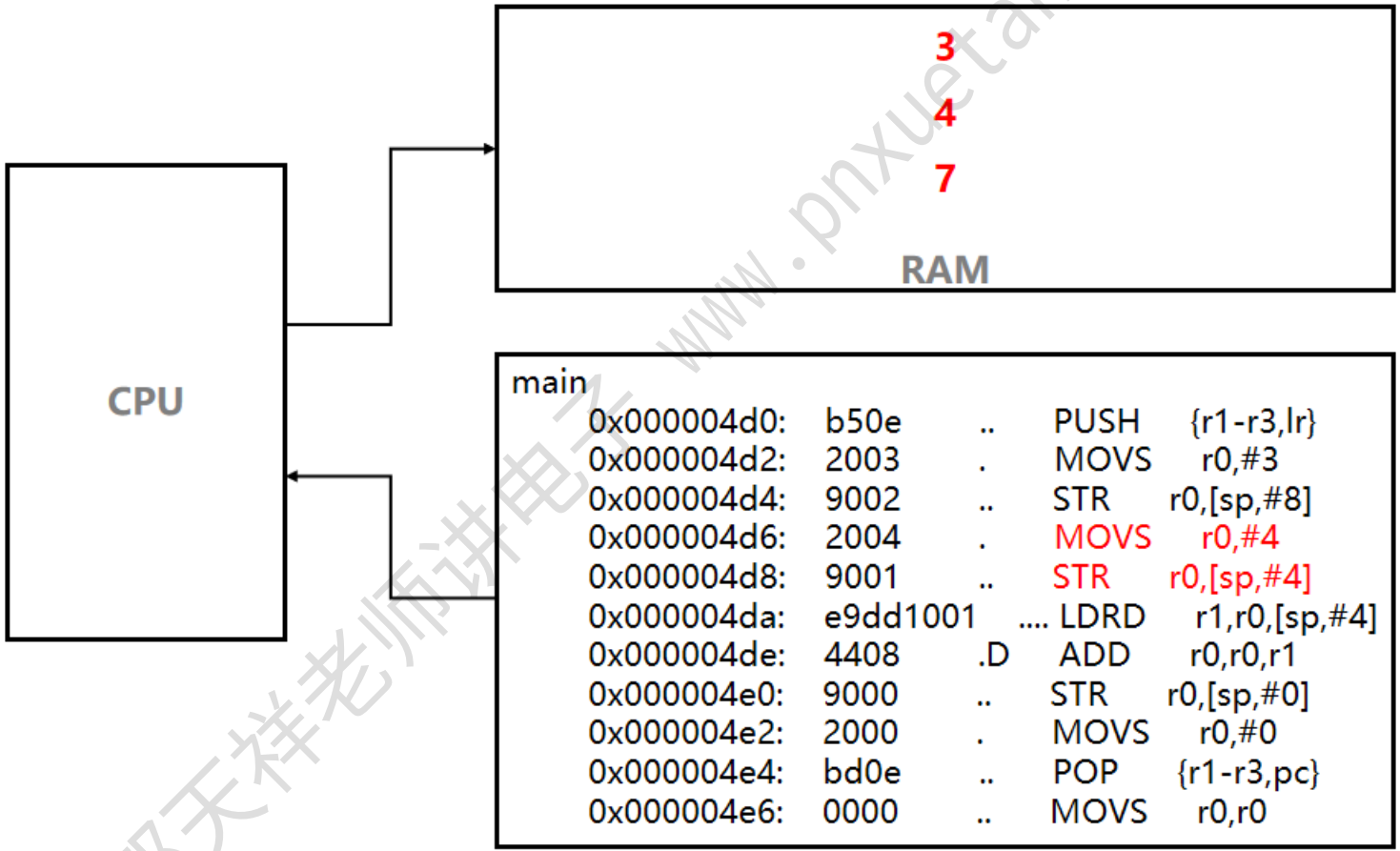
LDR R1,[R0,-R2] ;将 R0-R2 地址处的数据计读出,保存到 R1 中(R0 的值不变)

(3)寄存器及移位常数.寄存器移位后的值可以加到基址寄存器,也可以从基址寄存器中减去这个数值.指令举例如下:

LDR R1,[R0,R2,LSL #2] ;将 R0+R2*4 地址处的数据读出，保存到 R1 中（R0,R2 的值不变）

LDR R1,[R0,-R2,LSL #2];将R0-R2*4地址处的数据计读出,保存到R1中(R0,R2的值不变)

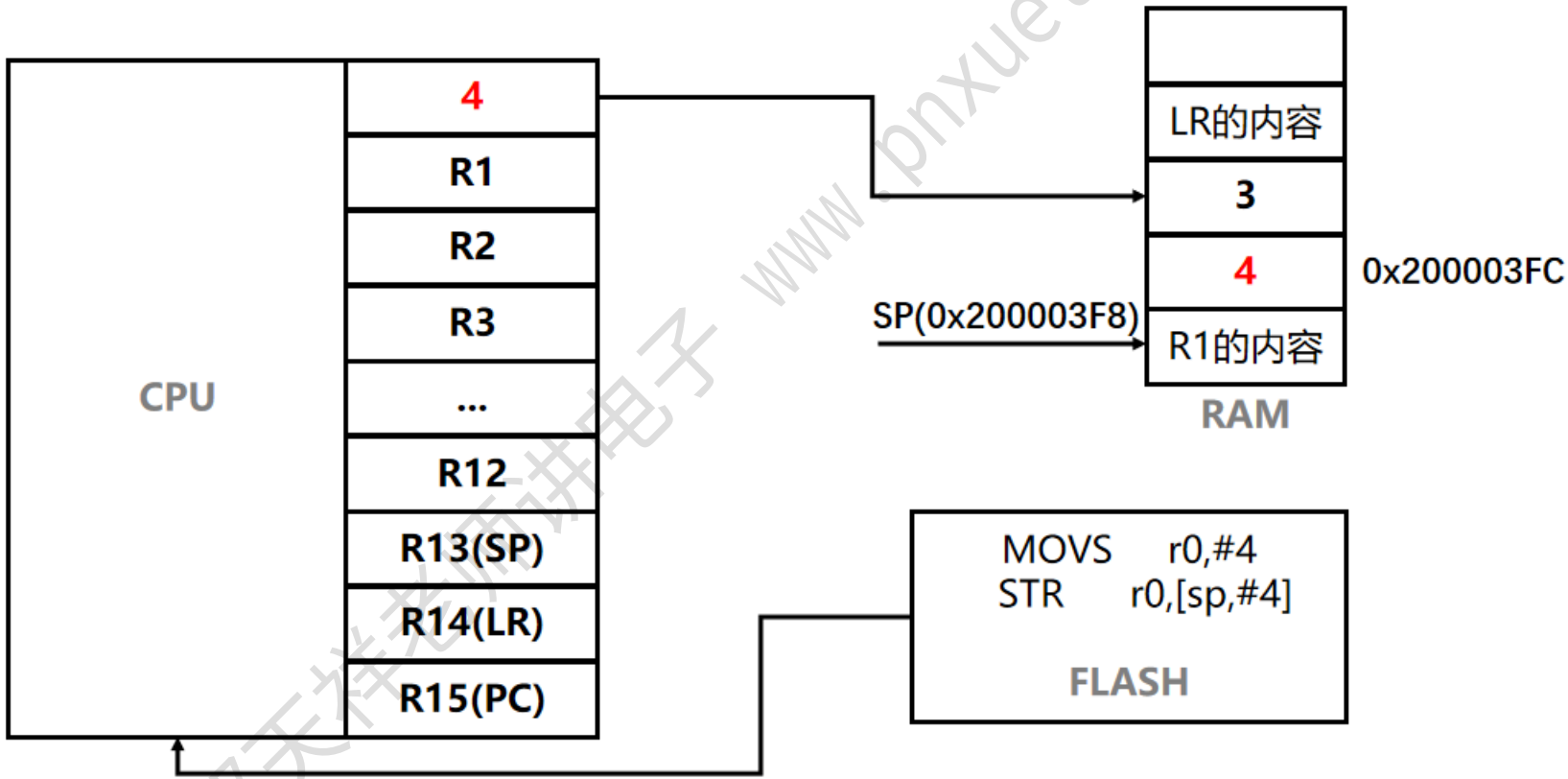
局部变量的数值是如何存储到RAM中的?



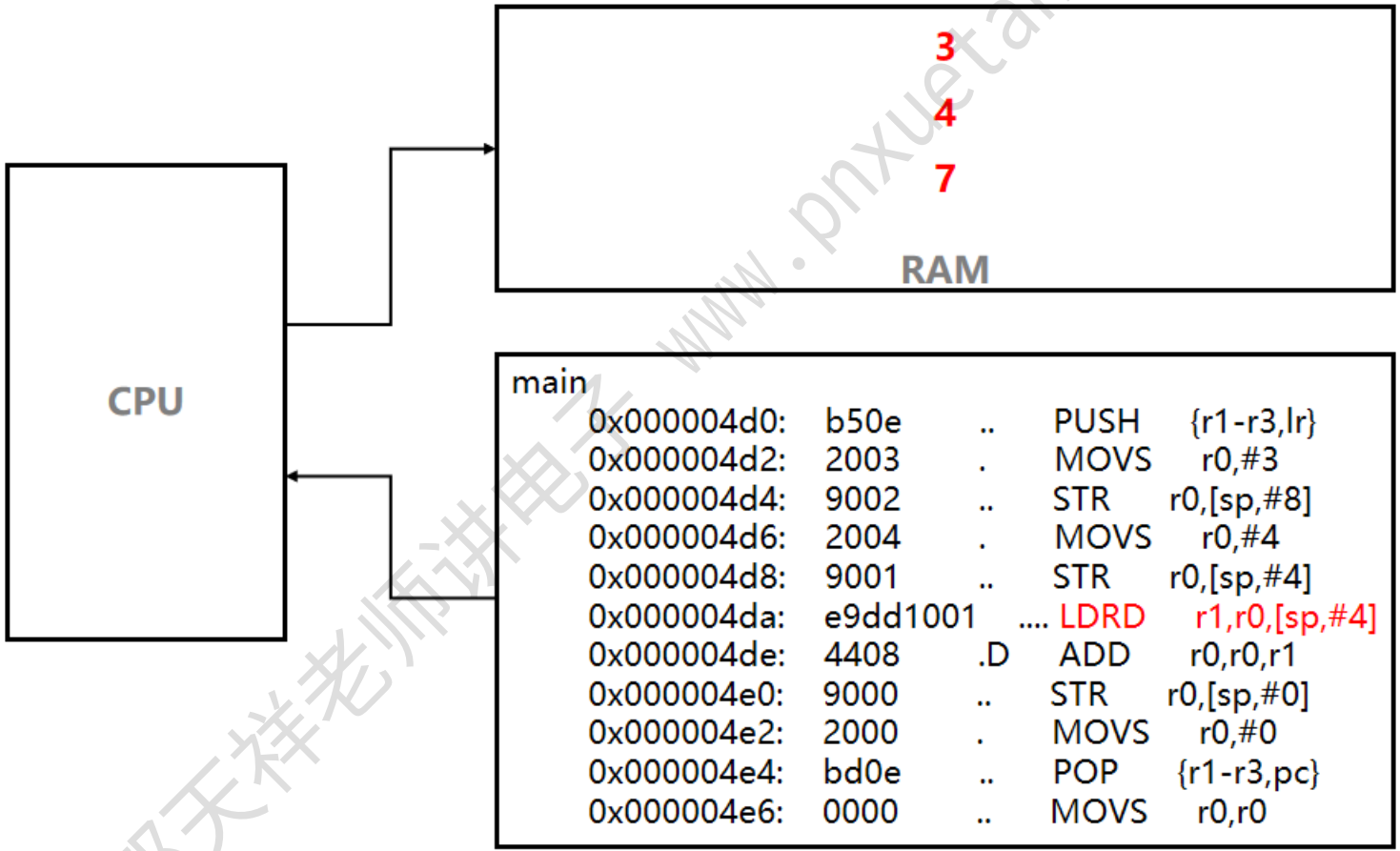
PN学堂

STR r0,[sp,#4]

- 这条指令的作用是将立即数4保存在地址为0x200003FC的栈中。



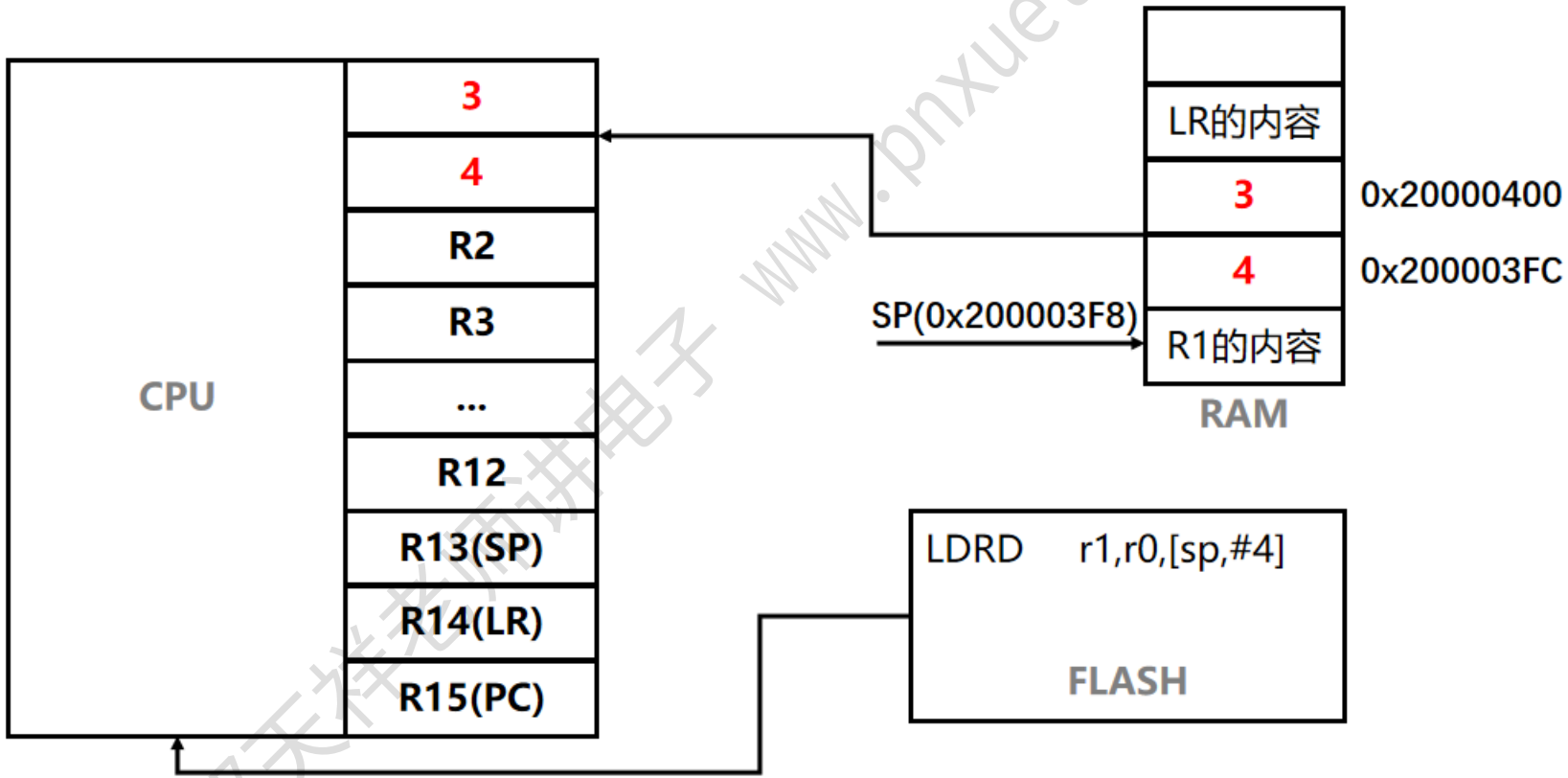
局部变量的数值是如何存储到RAM中的?



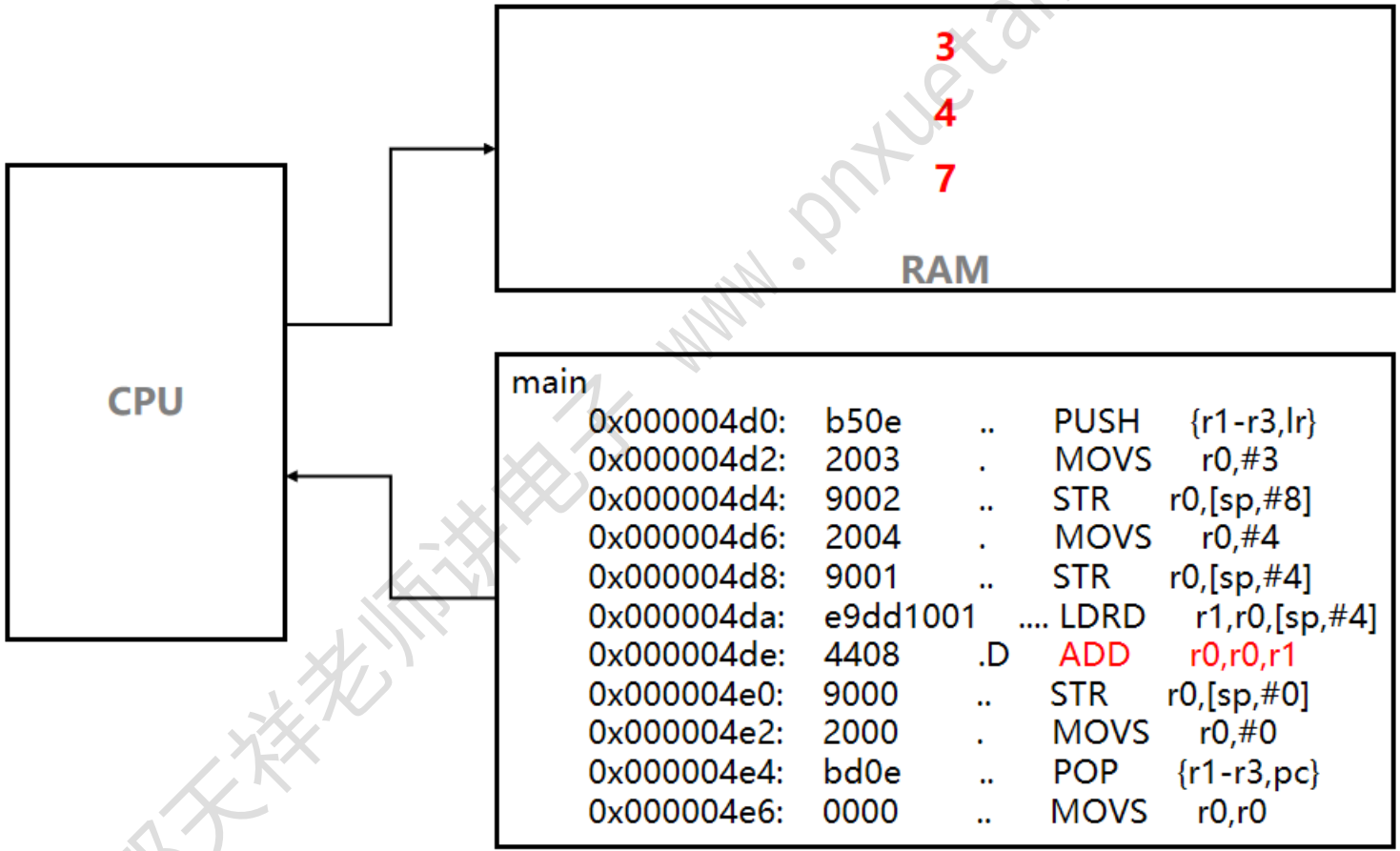
PN学堂

LDRD r1,r0,[sp,#4]

- 这条指令的作用是读取地址sp+4和sp+8中的数据，存储到r1和r0中。



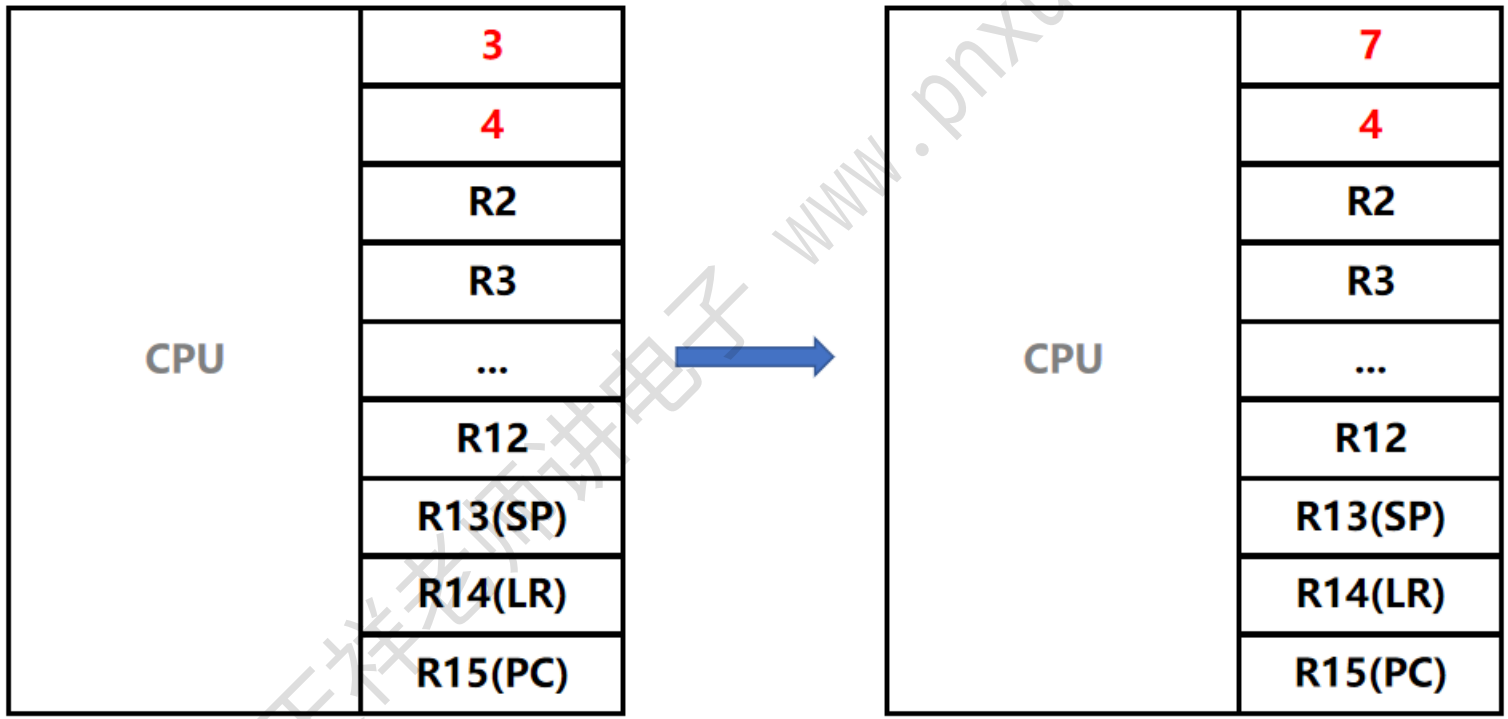
局部变量的数值是如何存储到RAM中的?



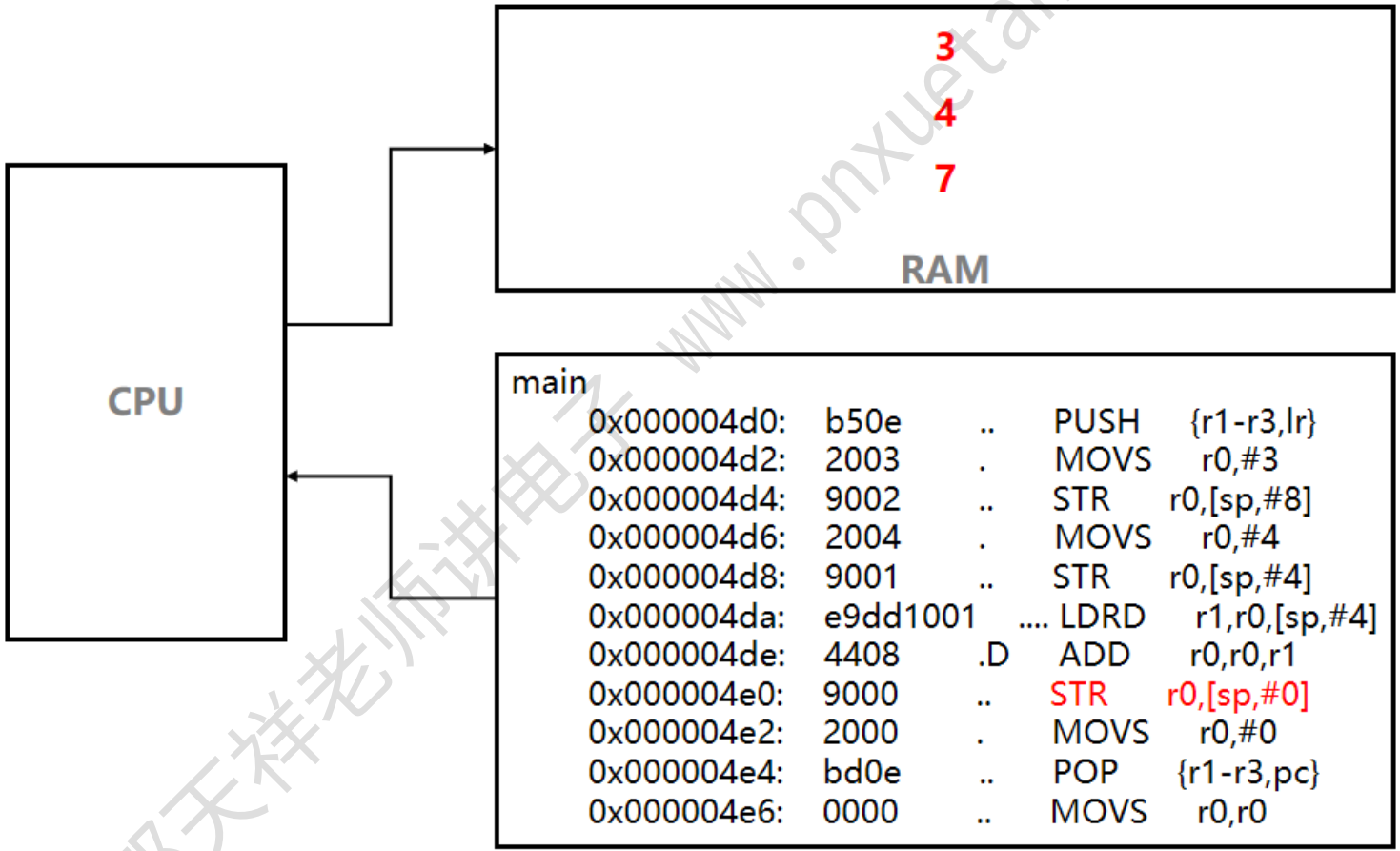
PN学堂

ADD r0,r0,r1

- 这条指令的作用是求r0(3) + r1(4)的和，将结果放到r0中。



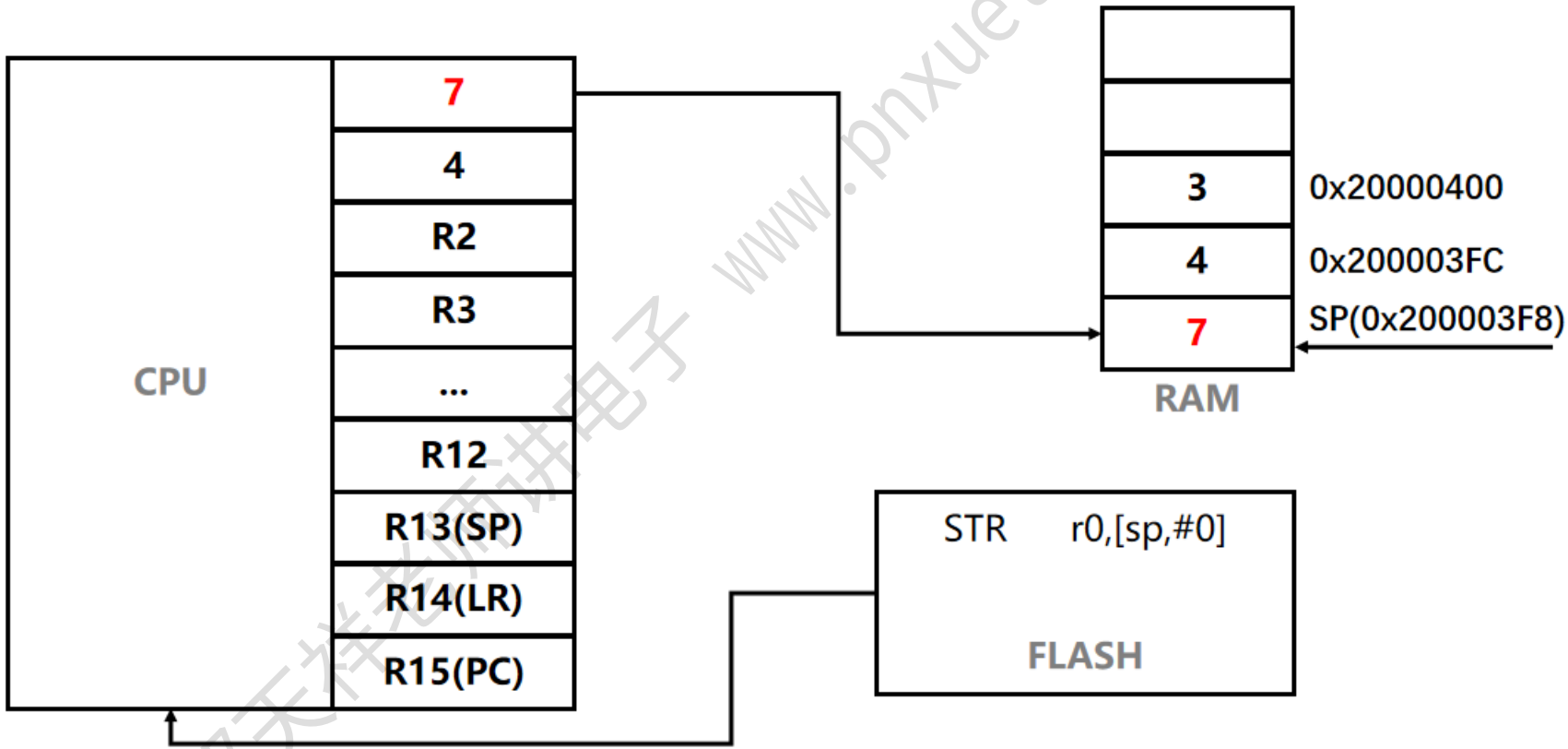
局部变量的数值是如何存储到RAM中的?



PN学堂

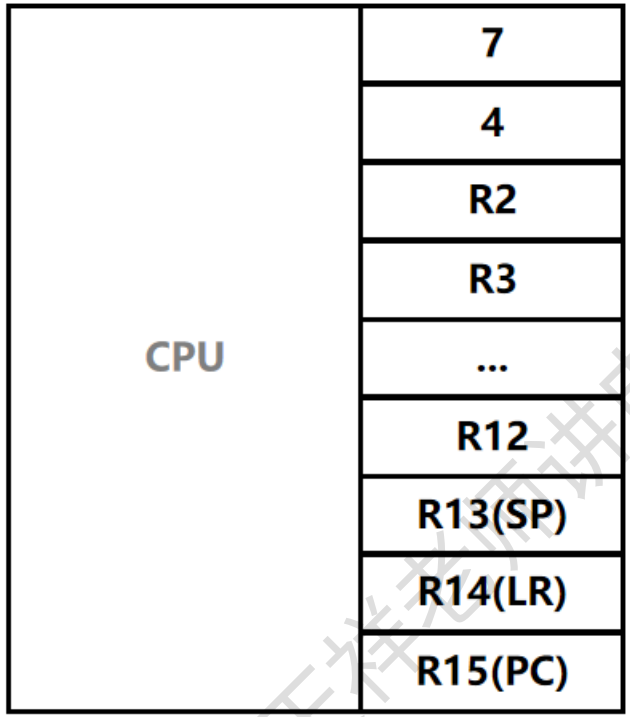
STR r0,[sp,#0]

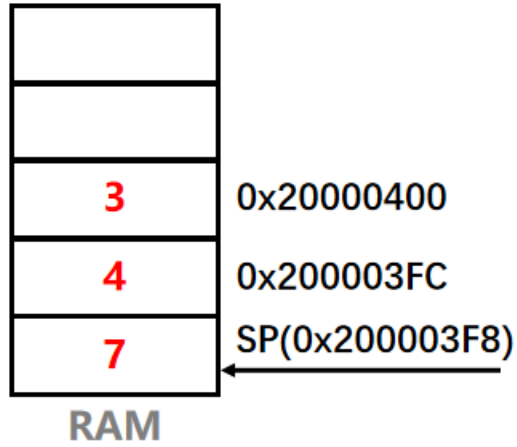
- 这条指令的作用是将r0(7)存储到地址为0x200003F8的栈中。



局部变量的数值是如何存储到RAM中的?

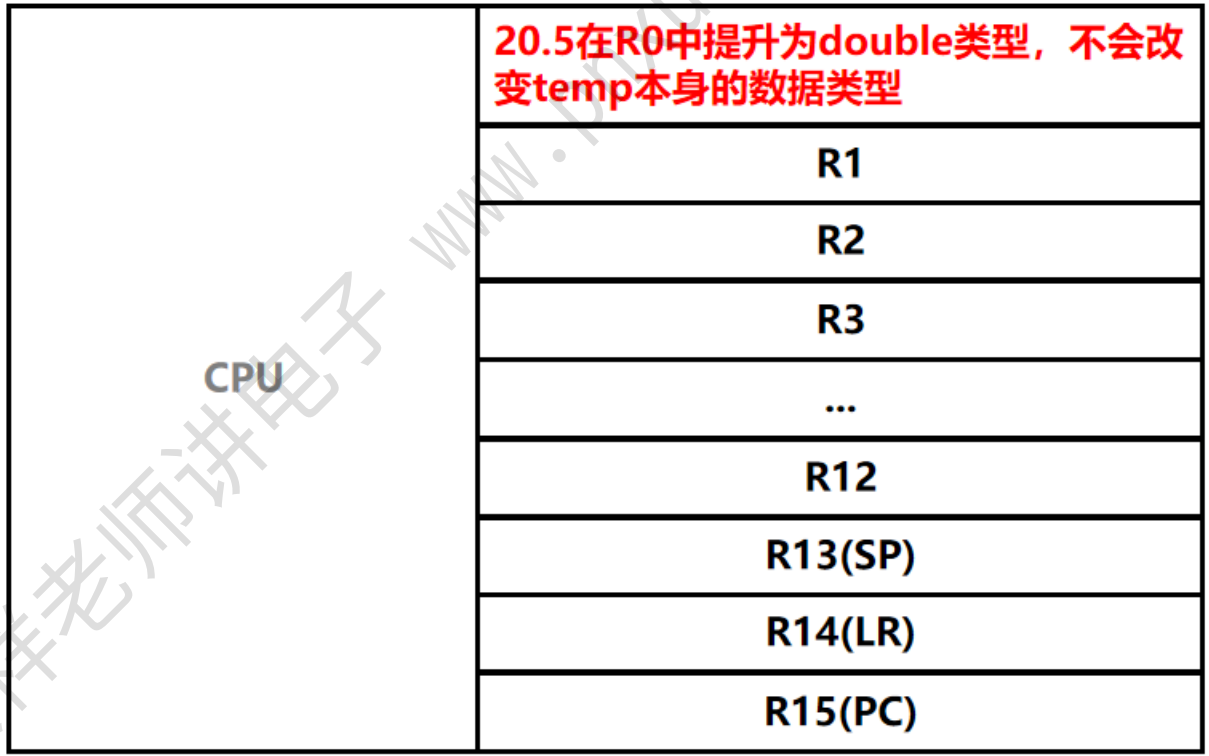
- 最终，在内存中就可以看到x、y、sum的数值了。





局部变量的数值是如何存储到RAM中的?

```
float temp = 20.5f;
temp *= 1.2;
```



THANK YOU!