

# Tutorial 9: Instruction Encoding and Decoding

## Problem 1: Pseudo Instruction

Consider the ARM assembly code in Figure 1 Answer the following questions:

What is the real ARM instruction corresponding to the pseudo instruction at label L2?

What is the real ARM instruction corresponding to the pseudo instruction at label L3?

What is the real ARM instruction corresponding to the pseudo instruction at label L4?

What is the content of the register r0 in instruction (str r0, [r2, #0])?

```
L1:    ldr    r0, data+4
L2:    ldr    r1, data+8
        ldr    r0, [r0, #0]
        ldr    r1, [r1, #0]
        add   r0, r0, r1
L3:    adr    r2, data
L4:    mov    r3, #0x00ffffff
        add   r0, r0, r3
        str   r0, [r2, #0]
ldmfd  sp!, {pc}

data:  .word   0x20000
        .word   0x300
        .word   0x400
```

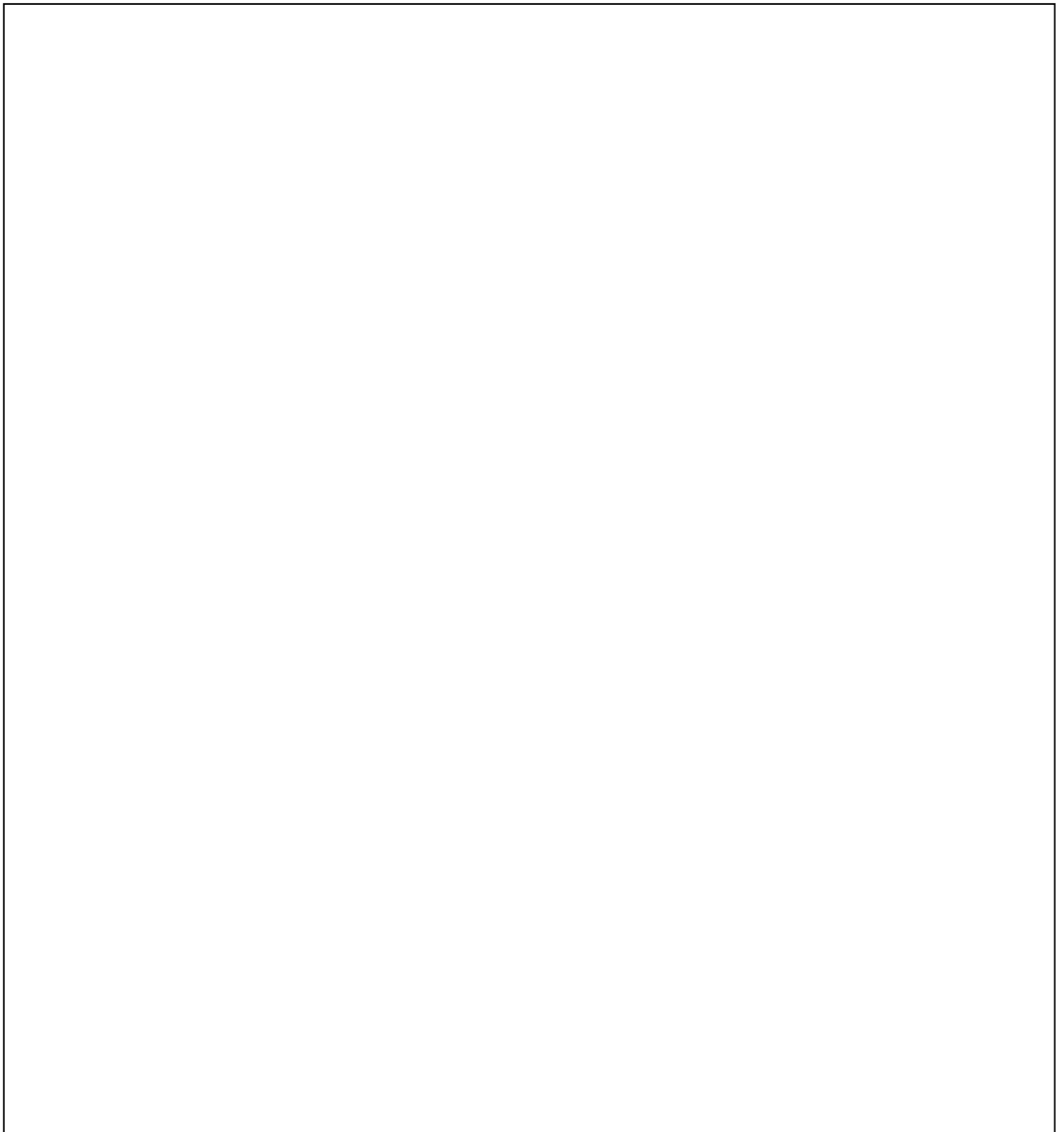
Figure 1: Sample ARM Assembly Code

## Problem 2: Instruction decoding

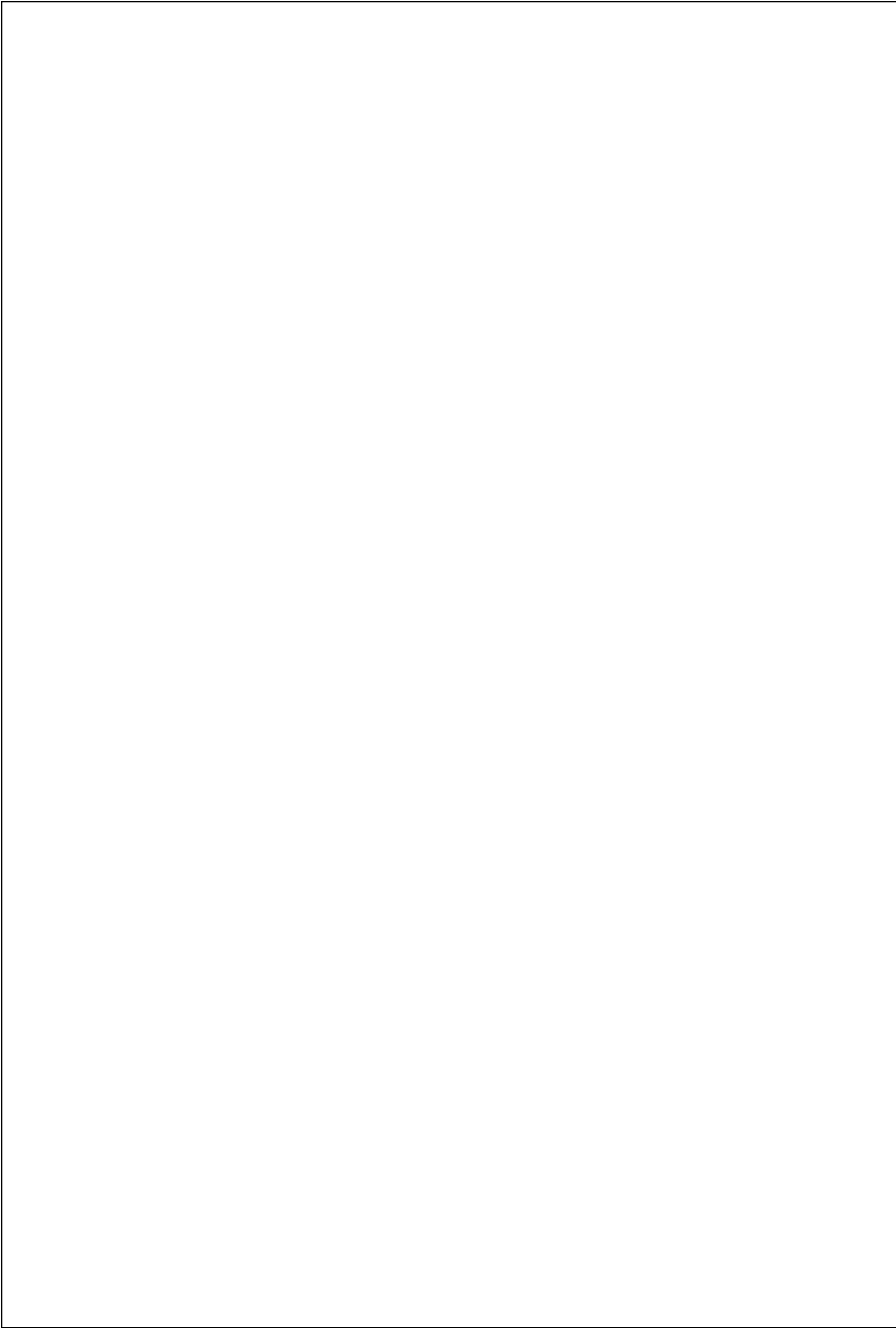
Convert the the ARM machine code in Figure 2 to a corresponding C Program

0:	e5902000
4:	e3a00000
8:	e1a03000
c:	e1530002
10:	a1a0f00e
14:	e0800003
18:	e2833001
1c:	e1530002
20:	baffffffb
24:	e1a0f00e

Figure 2: The Sample ARM Machine Code







### Problem 3: Literal Pool

Consider the C function in Figure 11. Produce the symbolic and true ARM assembly and machine versions of this code.

```
int sum(int s)
{
    int n = 0x00ff00ff,
    int m = 0xee00ee00;
    return (s + m + n);
}
```

Figure 11: C-code for the Demonstration of the Literal Pool