Lab Exercise #10 -- Assembler Processing

A. Complete the exercises below.

1. The second page of this worksheet lists the source lines for a ARM assembly language program. Fill in each blank in the column on the left of the page with the value of the location counter before that source line is processed during Pass One.

Note: the assembler maintains a separate location counter for each segment. Use the notation "T+xxxx" or "D+xxxx" (where "T" and "D" refer to the text and data segments, and "xxxx" refers to a hexadecimal offset) to give the value of the location counter.

2. Complete the symbol table shown below, based on the processing which you performed in part (1). For each symbol, indicate its value (a specific constant or a segment plus a hexadecimal offset), whether its value is absolute or relocatable, and whether it is a local or global symbol.

<table>
<thead>
<tr>
<th>symbol</th>
<th>value (segment+offset)</th>
<th>abs/rel</th>
<th>local/global</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

3. For each of the indicated source lines, fill in the object code (machine instruction or data value) which would generated during Pass Two and placed in the object code file. Give your answers using hexadecimal notation.

   | bge      | endloop          | @       |              |
   | bl       | unpack           | @       |              |
   | b        | loop             | @       |              |

B. Assemble the source code file and check your work using the following command:

   <prompt> /user/cse320/bin/assem /user/cse320/Labs/lab10.source.s

If any of your responses are incorrect, re-work that section of the worksheet.
.text
.balign 4

unpack:

    ldr    r4, =masks
    ldr    r1, [r4, #0]
    and   r1, r0, r1
    ldr    r2, [r4, #4]
    and   r2, r0, r2
    ldr    r3, [r4, #8]
    and   r3, r0, r3
    bx    lr

.data
.balign 4

masks:  .word   0x80000000
        .word   0x7f800000
        .word   0x007fffff

list:   .single 0r-64.75
        .single 0r+1.625
        .single 0r-1.3e-6
        .single 0r+1.5e+30
        .single 0r-2.25e-28

SIZE = 5

.global main
.text
.balign 4

main:   push    {lr}
        ldr    r5, =list
        mov    r6, #0
        loop:  cmp    r6, #SIZE
                bge    endloop
        lsl    r7, r6, #2
        ldr    r8, [r5, r7]
        ldr    r0, =fmt1
        mov    r1, r8
        bl     printf
        mov    r0, r8
        bl     unpack
        ldr    r0, =fmt2
        bl     printf
        add    r6, r6, #1
        b      loop
    endloop:
        pop    {lr}
        bx     lr

fmt1:   .asciz  "Number: %08x  
fmt2:   .asciz  "Fields: %08x %08x %08x
        .balign 4