Section 1 / Defining structs

Given:

```
struct Foo {
    short a;
    char b;
    int c;
};
struct Foo Bar = { Oxaaaa, Oxbb, Oxccccccc };
Here is one way of defining and accessing the struct:
                                                                               // 1
         .global
                         main
                                                                               // 2
         .text
         .align
                        2
                                                                               // 3
                                                                               // 4
main:
                                                                               // 5
                        x30, [sp, 16]!
                                                                               // 6
        str
                                                                               // 7
        ldr
                        x0, =fmt
                                                                               // 8
        ldr
                        x1, =Bar
                                                                               // 9
        ldrh
                        w2, [x1]
                                                                               // 10
                        w3, [x1, 2]
                                                                               // 11
        ldrb
                        w4, [x1, 4]
                                                                               // 12
        ldr
        bl
                        {\tt printf}
                                                                               // 13
                                                                               // 14
        ldr
                        x30, [sp], 16
                                                                               // 15
                                                                               // 16
        mov
                        w0, wzr
                                                                               // 17
        ret
                                                                               // 18
         .data
                                                                               // 19
                                                                               // 20
                        "%p a: 0x%x b: 0x%x c: 0x%x\n"
fmt:
                                                                               // 21
         .asciz
```

It would be understandable if you don't see where the struct is being defined. That's because it isn't. Rather, the implied +0 on line 10 and the 2 and 4 on lines 11 and 12 are the hard coded offsets into the struct.

Here is a second way to define a struct.

```
// 1
.global
               main
                                                                    // 2
.text
               2
                                                                    // 3
.align
                                                                    // 4
.equ
               foo_a, 0
                                 # like #define
                                                                    // 5
               foo_b, 2
                                 # like #define
                                                                    // 6
.equ
                                 # like #define
                                                                    // 7
               foo_c, 4
.equ
```

```
// 8
                                                                              // 9
main:
                        x30, [sp, 16]!
        str
                                                                              // 10
                                                                              // 11
        ldr
                         x0, =fmt
                                                                              // 12
        ldr
                         x1, =Bar
                                                                              // 13
        ldrh
                         w2, [x1, foo_a]
                                                                              // 14
                                                                              // 15
                         w3, [x1, foo_b]
        ldrb
        ldr
                         w4, [x1, foo_c]
                                                                              // 16
        bl
                         printf
                                                                              // 17
                                                                              // 18
                         x30, [sp], 16
                                                                              // 19
        ldr
                                                                              // 20
        mov
                         w0, wzr
                                                                              // 21
        ret
                                                                              // 22
                                                                              // 23
         .data
                                                                              // 24
                        "%p a: 0x\%x b: 0x\%x c: 0x\%x\n"
                                                                              // 25
fmt:
        .asciz
```

This method uses .equ to make the offsets into symbolic constants. This is just like using #define in C and C++. That is, the above is equivalent to the following in C or C++:

```
#define foo_a 0
#define foo_b 2
#define foo_c 4
```

Finally, here is a third way of defining structs.

```
.global
                                                                              // 1
                         main
                                                                              // 2
         .text
                         2
                                                                              // 3
         .align
                                                                              // 4
main:
                                                                              // 5
                         x30, [sp, 16]!
         str
                                                                              // 6
                                                                              // 7
        ldr
                         x0, =fmt
                                                                              // 8
        ldr
                         x1, =Bar
                                                                              // 9
        ldrh
                         w2, [x1, Foo.a]
                                                                              // 10
        ldrb
                         w3, [x1, Foo.b]
                                                                              // 11
        ldr
                         w4, [x1, Foo.c]
                                                                              // 12
                                                                              // 13
        bl
                         printf
                                                                              // 14
                         x30, [sp], 16
                                                                              // 15
        ldr
                         w0, wzr
                                                                              // 16
        mov
                                                                              // 17
        ret
                                                                              // 18
                                                                              // 19
         .section
                        Foo
```

```
\ensuremath{//} a starts at 0 and goes for 2
                                                                               // 20
         .struct
                         Foo.a + 2
                                       // b starts at 2 and goes for 2
                                                                               // 21
Foo.a:
         .struct
                         Foo.b + 2
Foo.b:
                                       // c starts at 4
                                                                               // 22
         .struct
Foo.c:
                                                                               // 23
                                                                               // 24
         .data
                                                                               // 25
                                                                               // 26
                        "%p a: 0x%x b: 0x%x c: 0x%x\n"
                                                                               // 27
fmt:
         .asciz
```

This method has a *substantial* benefit over the previous methods. Imagine you need to insert a new field between Foo.a and Foo.b. Simply do so. If you're using this third method, which is based on relative offsets, the assembler will do the work of adjusting the following offsets for you.