```
        .global _start

_start:

        .data

a:      .byte   1
b:      .word   32
c:      .byte   3
d:      .hword  45
e:      .hword  72
f:      .byte   15
g:      .word   128
```

Disassembly of section .data:

00020054 <a>:
  20054: 00002001 andeq r2, r0, r1

```
>>> print("{:b}".format(0x20055))
1000000000001010101
```

00020055 <b>: @ unaligned data. Notice non-word aligned addresses

  20055: 00000020 andeq r0, r0, r0, lsr #32

```
>>> print("{:b}".format(0x20059))
1000000000001011001
```

00020059 <c>:
  20059: 48002d03 stmdami r0, {r0, r1, r8, sl, fp, sp}

0002005a <d>:

```
>>> print("{:b}".format(0x2005a))
1000000000001011010
```

  2005a: 0048002d subeq r0, r8, sp, lsr #32

0002005c <e>:
  2005c: 800f0048 andhi r0, pc, r8, asr #32

```
>>> print("{:b}".format(0x2005e))
1000000000001011110
```

0002005e <f>:

  2005e: 0000800f andeq r8, r0, pc

```
>>> print("{:b}".format(0x2005f))
1000000000001011111
```

0002005f <g>:

  2005f: 00000080 andeq r0, r0, r0, lsl #1

```
        .global _start

_start:

        .data

a:      .byte   1
        .align  2               @ use .align: clear the two low order bits or offset
b:      .word   32
c:      .byte   3
d:      .hword  45
e:      .hword  72
f:      .byte   15
g:      .word   128
```

Disassembly of section .data:

00020054 <a>:
  20054: 00000001 andeq r0, r0, r1

00020058 <b>: This offset is now word aligned

```
>>> print("{:b}".format(0x20058))
10000000001011000
```

  20058: 00000020 andeq r0, r0, r0, lsr #32

0002005c <c>:

```
>>> print("{:b}".format(0x2005c))
10000000001011100
```

  2005c: 48002d03 stmdami r0, {r0, r1, r8, sl, fp, sp}

0002005d <d>: UNALIGNED!

```
>>> print("{:b}".format(0x2005d))
10000000001011101
```

  2005d: 0048002d subeq r0, r8, sp, lsr #32

0002005f <e>:

```
>>> print("{:b}".format(0x2005f))
10000000001011111
```

  2005f: 800f0048 andhi r0, pc, r8, asr #32

00020061 <f>:
  20061: 0000800f andeq r8, r0, pc

00020062 <g>:

```
        .global _start

_start:
```

```
        .data

a:      .byte   1
        .balign 2       @ use .balign: align address based on number of bytes
b:      .word   32
c:      .byte   3
d:      .hword  45
e:      .hword  72
f:      .byte   15
g:      .word   128

00020054 <a>:
  20054:        00200001        eoreq   r0, r0, r1

00020056 <b>:  HALF WORD ALIGNED
  20056:        00000020        andeq   r0, r0, r0, lsr #32

0002005a <c>:
  2005a:        48002d03        stmdami         r0, {r0, r1, r8, sl, fp, sp}

0002005b <d>:
  2005b:        0048002d        subeq   r0, r8, sp, lsr #32

0002005d <e>:
  2005d:        800f0048        andhi   r0, pc, r8, asr #32

0002005f <f>:
  2005f:        0000800f        andeq   r8, r0, pc

00020060 <g>:
```

```
>>> print("{:b}".format(0x20056))
100000000001010110
```

```
        .global _start

_start:

        .data

a:      .byte   1
        .balign 4       @ use .balign: align address based on number of bytes
b:      .word   32
c:      .byte   3
d:      .hword  45
e:      .hword  72
f:      .byte   15
g:      .word   128
```

Disassembly of section .data:

00020054 <a>:
  20054: 00000001 andeq r0, r0, r1

00020058 <b>:
  20058: 00000020 andeq r0, r0, r0, lsr #32

0002005c <c>:
  2005c: 48002d03 stmdami r0, {r0, r1, r8, sl, fp, sp}

0002005d <d>:
  2005d: 0048002d subeq r0, r8, sp, lsr #32

0002005f <e>:
  2005f: 800f0048 andhi r0, pc, r8, asr #32

00020061 <f>:
  20061: 0000800f andeq r8, r0, pc

00020062 <g>:
  20062: 00000080 andeq r0, r0, r0, lsl #1