```
                ; Tell the assembler to allocate space for the variables starting
                ; at address 002ah. Each variable is initialized to 0.
                        org     002ah
002a    00      widget_cnt:
                        db      0
002b    00      got_widgets:
                        db      0

                ; Tell the assembler that code generation should start at address
                ; 0000h.
                        org     0000h
                ; Load widget_cnt & test for 0. The ori instruction sets the
                ; condition codes. Then jump to the proper instruction sequence.
0000    3a              lda     widget_cnt
0001    2a
0002    00
0003    f6              ori     00h
0004    00
0005    ca              jz      no_widgets
0006    10
0007    00
0008    fa              jm      no_widgets
0009    10
000a    00
                ; Set got_widgets to the proper value. We load A with the correct
                ; value, then store A into got_widgets.
000b    3e      widgets:
                        mvi     A,01h
000c    01
000d    c3              jmp     set_value
000e    12
000f    00
0010    3e      no_widgets:
                        mvi     A,00h
0011    00
0012    32      set_value:
                        sta     got_widgets
0013    2b
0014    00
```