Key to Practical 10
Space Invaders (Part 3)

Step 1

```
Main                    ; A0 points to the dot matrix of the invader.
                        lea     Bitmap_Invader,a0

                        ; A1 points to the video memory.
                        lea     VIDEO_START,a1

                        ; D7.W = Loop counter
                        ;      = Number of iterations - 1 (DBRA)
                        ; Number of iterations = Number of lines
                        move.w  #16-1,d7

\loop                   ; Draw an invader pixel line.
                        ; (22 pixels require 3 bytes.)
                        move.b  (a0)+,(a1)
                        move.b  (a0)+,1(a1)
                        move.b  (a0)+,2(a1)

                        ; Point to the video address of the next line.
                        adda.l  #BYTE_PER_LINE,a1

                        ; Branch to loop as long as there are lines to draw.
                        dbra    d7,\loop

                        illegal
```

Step 2

```
PixelToByte         ; Size in pixels + 7 -> D3.W
                    addq.w  #7,d3

                    ; D3.W/8 -> D3.W
                    lsr.w   #3,d3

                    ; Return from subroutine.
                    rts
```

```
CopyLine            ; Save registers on the stack.
                    movem.l d3/a1,-(a7)

                    ; Number of iterations = Width in bytes
                    ; Number of iterations - 1 -> D3.W (DBRA)
                    subq.w  #1,d3

\loop               ; Copy all the bytes of the line.
                    move.b  (a0)+,(a1)+
                    dbra    d3,\loop

                    ; Restore registers from the stack and return from subroutine.
                    movem.l (a7)+,d3/a1
                    rts
```

```
CopyBitmap          ; Save registers on the stack.
                    movem.l d3/d4/a0/a1,-(a7)

                    ; Width in bytes -> D3.W
                    move.w  WIDTH(a0),d3
                    jsr     PixelToByte

                    ; Number of iterations - 1 -> D4.W (DBRA)
                    ; Number of iterations = Height in pixels
                    move.w  HEIGHT(a0),d4
                    subq.w  #1,d4

                    ; Address of the dot matrix -> A0.L
                    lea     MATRIX(a0),a0

\loop               ; Copy a line of the matrix.
                    jsr     CopyLine

                    ; Point to the video address of the next line.
                    adda.l  #BYTE_PER_LINE,a1

                    ; Branch to loop as long as there are lines to draw.
                    dbra    d4,\loop

                    ; Restore registers from the stack and return from subroutine.
                    movem.l (a7)+,d3/d4/a0/a1
                    rts
```

Step 3

```
                    ; ==============================
                    ; Data
                    ; ==============================

InvaderA_Bitmap     dc.w    24,16
                    dc.b    %00000000,%11111111,%00000000
                    dc.b    %00000000,%11111111,%00000000
                    dc.b    %00111111,%11111111,%11111100
                    dc.b    %00111111,%11111111,%11111100
                    dc.b    %11111111,%11111111,%11111111
                    dc.b    %11111111,%11111111,%11111111
                    dc.b    %11111100,%00111100,%00111111
                    dc.b    %11111100,%00111100,%00111111
                    dc.b    %11111111,%11111111,%11111111
                    dc.b    %11111111,%11111111,%11111111
                    dc.b    %00000011,%11000011,%11000000
                    dc.b    %00000011,%11000011,%11000000
                    dc.b    %00001111,%00111100,%11110000
                    dc.b    %00001111,%00111100,%11110000
                    dc.b    %11110000,%00000000,%00001111
                    dc.b    %11110000,%00000000,%00001111

InvaderB_Bitmap     dc.w    22,16
                    ; ...
                    ; ...

InvaderC_Bitmap     dc.w    16,16
                    dc.b    %00000011,%11000000
                    dc.b    %00000011,%11000000
                    dc.b    %00001111,%11110000
                    dc.b    %00001111,%11110000
                    dc.b    %00111111,%11111100
                    dc.b    %00111111,%11111100
                    dc.b    %11110011,%11001111
                    dc.b    %11110011,%11001111
                    dc.b    %11111111,%11111111
                    dc.b    %11111111,%11111111
                    dc.b    %00110011,%11001100
                    dc.b    %00110011,%11001100
                    dc.b    %11000000,%00000011
                    dc.b    %11000000,%00000011
                    dc.b    %00110000,%00001100
                    dc.b    %00110000,%00001100

Ship_Bitmap         dc.w    24,14
                    dc.b    %00000000,%00011000,%00000000
                    dc.b    %00000000,%00011000,%00000000
                    dc.b    %00000000,%01111110,%00000000
                    dc.b    %00000000,%01111110,%00000000
                    dc.b    %00000000,%01111110,%00000000
                    dc.b    %00000000,%01111110,%00000000
                    dc.b    %00111111,%11111111,%11111100
                    dc.b    %00111111,%11111111,%11111100
                    dc.b    %11111111,%11111111,%11111111
                    dc.b    %11111111,%11111111,%11111111
                    dc.b    %11111111,%11111111,%11111111
                    dc.b    %11111111,%11111111,%11111111
                    dc.b    %11111111,%11111111,%11111111
                    dc.b    %11111111,%11111111,%11111111
```