

# mvac7 blog

01 agosto 2017

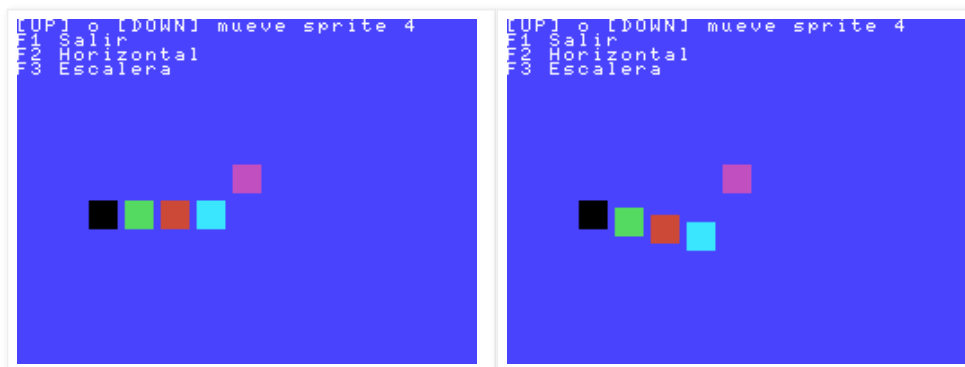
## Sprites MSX · La regla del quinto Sprite

Una de las limitaciones que tienen los vídeo procesadores de los ordenadores y consolas de 8 y 16bits que disponen de Sprites, es que dependiendo de la potencia de estos, permitirán más cantidad de planos y se podrán representar un número limitado de estos horizontalmente.

En el caso del [TMS9918A](#), solo permite mostrar 4, por lo que se ha bautizado por la *Regla del Quinto Sprite* y tendremos que tenerla en cuenta cuando diseñemos un juego o aplicación donde utilicemos más de cuatro planos de sprite.

Qué Sprites se muestran y cuáles no, dependerá del número de plano en el que se encuentren, ya que tienen preferencia los planos superiores.

Hay que tener en cuenta de que el VDP dibuja la pantalla línea a línea por lo que esta limitación actúa a nivel de línea. Es decir, solo aquellas partes de los sprites que se encuentren en conflicto con esta limitación, no se dibujarán, así que dependiendo de la posición de estos se podrá ver Sprites parcialmente dibujados.



*Demostración del efecto de la Regla del Quinto Sprite (Ejemplo 1).*

### Notas:

Esta norma tiene en cuenta todo el área del sprite (8x8 o 16x16), independientemente de si el patrón tiene puntos dibujados o se muestran con el color 0 (transparente).

Los VDPs que contengan compatibilidad con el TMS9918A, dispondrán de la misma limitación en los modos iguales (el V9938, el V9958 o el VDP de la Sega Master System).

En los nuevos modos gráficos de los vídeo procesadores V9938 y V9958, disponen de la misma limitación pero con la mejora de que permiten hasta 8 sprites por línea, mientras que el V9990 se podrá mostrar hasta 16 sprites horizontales.

### Ejemplo 1

A continuación tenéis un ejemplo en MSX Basic donde podéis probar cómo afecta la ley del quinto sprite a dos visualizaciones diferentes: horizontal (desaparece la totalidad del quinto sprite), o en escalera (afecta a una parte del quinto sprite).

```
10 REM Ejemplo Regla del Quinto Sprite
20 SCREEN 1,2:KEY OFF
30 WIDTH 32
```

```

40 PAT=BASE(9):REM tabla de patrones de sprite
50 Y=80
60 FOR BC=PAT TO PAT+31:VPOKE BC,255:NEXT
70 PRINT "[UP] o [DOWN] mueve sprite 4"
80 PRINT "F1 Salir"
90 PRINT "F2 Horizontal"
100 PRINT "F3 Escalera"
110 ON KEY GOSUB 400,200,300
120 KEY(1) ON:KEY(2) ON:KEY(3) ON
130 GOSUB 200
140 PUT SPRITE 4,(120,Y),13,0
150 IF STICK(0)=1 THEN Y=Y-1
160 IF STICK(0)=5 THEN Y=Y+1
170 GOTO 140
200 PUT SPRITE 0,(40,100),1,0
210 PUT SPRITE 1,(60,100),3,0
220 PUT SPRITE 2,(80,100),6,0
230 PUT SPRITE 3,(100,100),7,0
240 RETURN
300 PUT SPRITE 0,(40,100),1,0
310 PUT SPRITE 1,(60,104),3,0
320 PUT SPRITE 2,(80,108),6,0
330 PUT SPRITE 3,(100,112),7,0
340 RETURN
400 END

```

Descargar Ejemplo 1: [SPRITE5.BAS](#)

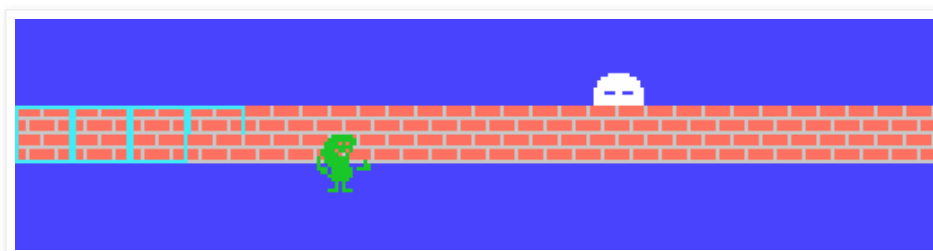
## Técnica para hacer desaparecer un sprite verticalmente con un gráfico de la pantalla

Aprovechando el efecto de la limitación de sprites horizontales, podemos utilizarla para que nuestras figuras se oculten en una área determinada de la pantalla, generando el efecto de que están pasando por debajo de un gráfico. Este truco solo funciona con bloques horizontales múltiplos de 8 o 16 líneas, según el tipo de sprites que tengamos configurado en la pantalla.

El efecto es muy sencillo. Añadiremos cuatro sprites con color 0 en la misma coordenada vertical con planos superiores al resto de figuras (por ejemplo los planos del 0 al 3).

Si queremos que este truco no afecte a un grupo de sprites para que estos se muestren por encima del gráfico, tendremos que colocarlos en planos superiores a los utilizados para generar este efecto.

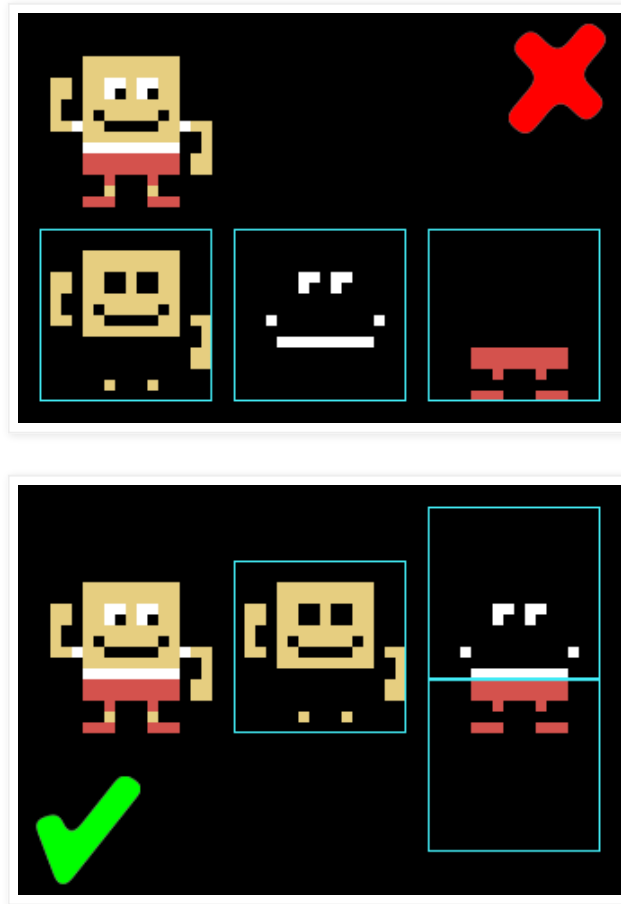
Un caso práctico sería su uso en un marcador horizontal, en la parte superior o inferior de la pantalla.



En la imagen podemos ver en el plano 0 una figura de color verde que se muestra por encima del gráfico de ladrillos. En los planos del 1 al 4, se ha utilizado un patrón de un recuadro mostrado en azul para distinguirlos (a la práctica utilizaremos color 0). El sprite blanco se encuentra en un plano inferior por lo que desaparece la parte que colisiona con los primeros planos realizando el efecto de ocultación por detrás de los ladrillos.

## Como utilizar 3 o más colores para un personaje y que afecte como 2 sprites horizontales

A la hora de dibujar una figura se suele utilizar una matriz de 16x16 y se suele pintar con los diferentes colores que necesitamos. A la hora de programar tendremos que usar un plano por color, por lo que nos podemos encontrar muy rápido con la regla del quinto sprite. Hay una forma de economizar el uso de planos horizontalmente que consiste en dibujar nuestros sprites de forma que colocados verticalmente solo se solapen 2 horizontalmente. De esta forma podemos concatenar verticalmente tantos sprites como necesitemos para completar la altura de nuestro personaje.



En las imágenes podemos ver la forma incorrecta y correcta de crear los patrones y colocarlos en la pantalla. Hemos utilizado un sprite para el color principal (amarillo) y dos sprites colocados verticalmente para dar los colores de detalle de la parte superior y la parte inferior de la figura.

## Técnica de parpadeo de sprites (sprite flicker)

Para poder mostrar más de 4 sprites horizontalmente, se suele utilizar una técnica mediante software que consiste en cambiar la prioridad de los planos en cada interrupción de barrido de la pantalla ([VBLANK](#)). Esto produce un parpadeo en los sprites afectados por la regla del quinto sprite.



*Ilustración del funcionamiento de la técnica de Sprite Flicker.  
En un MSX real la frecuencia es mucho mayor (50 veces por segundo en PAL).*

Una rutina de Sprite Flicker se basa en disponer de una copia de la tabla de atributos de Sprite (OAM), en RAM, donde se escribirán los cambios de las figuras móviles de nuestro motor de juego. La rutina de parpadeo copiará los datos de los planos a un segundo buffer, en un orden diferente cada vez que se ejecute. Este segundo buffer es el que se deberá volcar a la VRAM en la interrupción de VBLANK.

Aunque en muchos casos pueda funcionar una rutina genérica, dependiendo del diseño del juego puede ser necesario programar una que afecte de una forma concreta un rango de sprites. Por lo general, se suelen reservar los primeros planos para la figura del personaje principal y dependiendo del caso, la de otros personajes. Se suele dejar para la rutina de Flicker, los planos utilizados con la intención de dar un color de detalle de los personajes y no importe que en un momento dado parpadee. Donde más interesa utilizarla es en sprites que estén en continuo movimiento y se puedan ver afectados en determinados momentos, como por ejemplo en los proyectiles o enemigos de un shooter vertical.

También tenemos que tener en cuenta que el Sprite Flicker permitirá mostrar más o menos sprites por línea, dependiendo de la cantidad de planos que fijemos y que sprites se vean afectados en determinados casos. Un caso puede ser como el del ejemplo, donde hay dos fijos que están coincidiendo en la misma línea, por lo que se pueden ver 6: 2 fijos y 4 parpadeando (da la sensación de 7, ya que la posición de los dos sprites con forma de seta, se ven afectados parcialmente). En esta misma configuración, en el caso de que no coincidan en una línea los planos que están fuera de la rutina de Sprite Flicker, se podrían ver hasta 8 sprites parpadeando.

Es aconsejable que a la hora de diseñar un juego, controlar la cantidad de sprites utilizados y cómo se pueden ver afectados por la regla del quinto sprite, ya que un exceso de parpadeo puede ser molesto y hacer que el juego pierda jugabilidad.

La forma de intercambiar los planos podrá resolverse de diferentes formas, dependiendo del caso. Habrá ocasiones donde sí tenemos muy controlado como se muestran los sprites, se pueda resolver intercambiando los atributos entre dos bloques, mientras que en otros, puede ser necesario una técnica por rotación o inversión, como la que he utilizado en el Ejemplo 2. Un ejemplo de inversión sería trabajar con los atributos de los planos 10 al 20 e invertirlos para que el 20 pase a ser el 10, el 21 el 11 y así hasta el último.

## Ejemplo 2

El ejemplo descargable está realizado en ensamblador para [Sjasm Z80 v0.42c de XL2S](#) Entertainment. En el se puede ver el efecto de ocultación horizontal y del funcionamiento de la técnica de parpadeo de sprites. Incluye una rutina de parpadeo por inversión que he programado y que podéis usar y modificar libremente, para vuestros desarrollos. Se puede ajustar el rango de sprites que han de funcionar dentro de la inversión de planos. Para ello podéis modificar los valores de las etiquetas: `SPRFLK_INIT` y `SPRFLK_END`.

- [Sources](#) (requiere [Sjasm Z80 v0.42c](#))
- [SPR5FLKi.ROM](#)

## Referencias

- Karoshi · [Regla 5º Sprite, otra vez...](#)
- MSX Resorce Center · [5th sprite horizontal](#)



Etiquetas: [desarrollo](#), [MSX](#)

## Enlaces a esta entrada

[Crear un enlace](#)

Entrada más reciente · · · · · [Página principal](#) · · · · · [Entrada antigua](#) ·