

mvac7 blog

27 junio 2014

Compresión RLE WB para MSX

Hace un tiempo, estuve investigando la compresión RLE ([Run-Length Encoding](#)), para utilizarlo en gráficos de MSX. Si no conoces el RLE, se puede resumir, diciendo que es un sistema muy simple, que se basa en reducir secuencias de valores repetidos.

Si hablamos del RLE más básico, en el modo gráfico de los MSX de primera generación (SC2), funciona muy bien con las tablas de colores, ya que es donde suele repetirse muchos bloques de datos. En muchos casos se puede conseguir más de un 70% de ganancia, dependiendo de la técnica creativa del dibujo y si la relación patrón/color(*1) esta correctamente ordenada.

Sin embargo, a la hora de comprimir los datos de los patrones, este sistema no suele ser efectivo, ya que los valores sufren más de variaciones que los colores. Esto provoca que por cada valor se gasten dos Bytes, por lo que en muchos casos obtendríamos el efecto contrario, un bloque de datos de mayor tamaño que el original.

Por otro lado, una de las ventajas que tiene, es que es muy rápido y a la hora de programar, es más simple que usar una función de volcado a VRAM (como la función de la BIOS LDIRVM), ya que no necesitaremos averiguar e indicar el tamaño del bloque de datos.

Pero volviendo al tema de este post, pensando en un sistema basado en RLE que no penalizará en los valores no repetidos, encontré en la WEB [SMS Power](#), un artículo donde se describen diferentes sistemas utilizados en varios juegos de la Sega Master System. De entre todos, le vi posibilidades al del [Wonder Boy](#), así que lo adapté para MSX, modificando algunas cosas ya que está adaptado a la forma de trabajar el modo gráfico de la Master System.

Su funcionamiento es el siguiente:

- Se utiliza un valor como control. He elegido el valor \$80 ya que el 0 es un valor muy común y como veremos más adelante, cuando un valor coincida con el número de control se utilizarán 2 bytes. Con este pequeño cambio se obtiene mayor compresión. En una prueba de un tileset de 115 tiles, obtuve una ganancia de más de 70 Bytes.
- Si el siguiente valor al control es 0, se utiliza para cuando el valor coincide con el de control y este no se repite.
- Si el siguiente valor al control es igual a 255, indicará que es el final.
- Si el siguiente valor al control no es ni 0 ni 255, entonces indicará el número de repeticiones (entre 1 y 254), y habrá un tercer dato que contendrá el valor que se repite. Como 1 repetición no se utiliza, en el decodificador se incrementa uno para abarcar de 2 a 255 repeticiones.
- Si el primer valor no es igual al número asignado como control, será un dato que no se repite y se escribe directamente a la salida.

Forma resumida:

```
$80 nn dd      ;repite nn veces (de $1 a $FE) el valor dd
$80 $0         ;para cuando el valor a escribir es igual al      dígito
de control ($80) sin repetición
$80 $FF        ;final del bloque de datos
dd (!=$80)     ;valor sin repetición
```

En definitiva, los datos que no se repitan sólo ocuparán un valor excepto cuando sea igual al valor de control que se usarán dos, pero en los casos que si se repitan, se gastarán 3 Bytes (dígito de control, numero de repeticiones y valor), uno más que el RLE básico.

Este sistema funciona muy bien con los datos de patrones, pero con los colores hay casos donde es más eficiente el RLE básico.

Adjunto la rutina descompresora a RAM y a VRAM, junto con un ejemplo:

- [unRLEWB v1.1 Assembler \(asMSX\)](#)
- [unRLEWB v1.1 C \(SDCC\)](#)

En breve publicaré la utilidad MSXtiles devtool, donde se puede sacar a código y de forma independiente (patrones, colores y mapa), los diferentes tipos de datos que componen una pantalla de MSX (SC2) en RLE básico o el basado en WB.

Notas:

*1 - Si creamos una imagen utilizando un editor gráfico, al convertirlo a datos en formato MSX, puede que dependiendo del algoritmo de conversión, los valores de las líneas queden desordenados, lo cual penalizaría el ratio de compresión. Ejemplo:

- Forma desordenada: \$FA,\$AF,\$AF,\$FA,\$AF,\$1F,\$F1
- Forma ordenada: \$FA,\$FA,\$FA,\$FA,\$FA,\$1F,\$1F



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

Enlaces a esta entrada

[Crear un enlace](#)

Entrada más reciente • • • • • • • • • • [Página principal](#) • • • • • • • • • • [Entrada antigua](#) |

