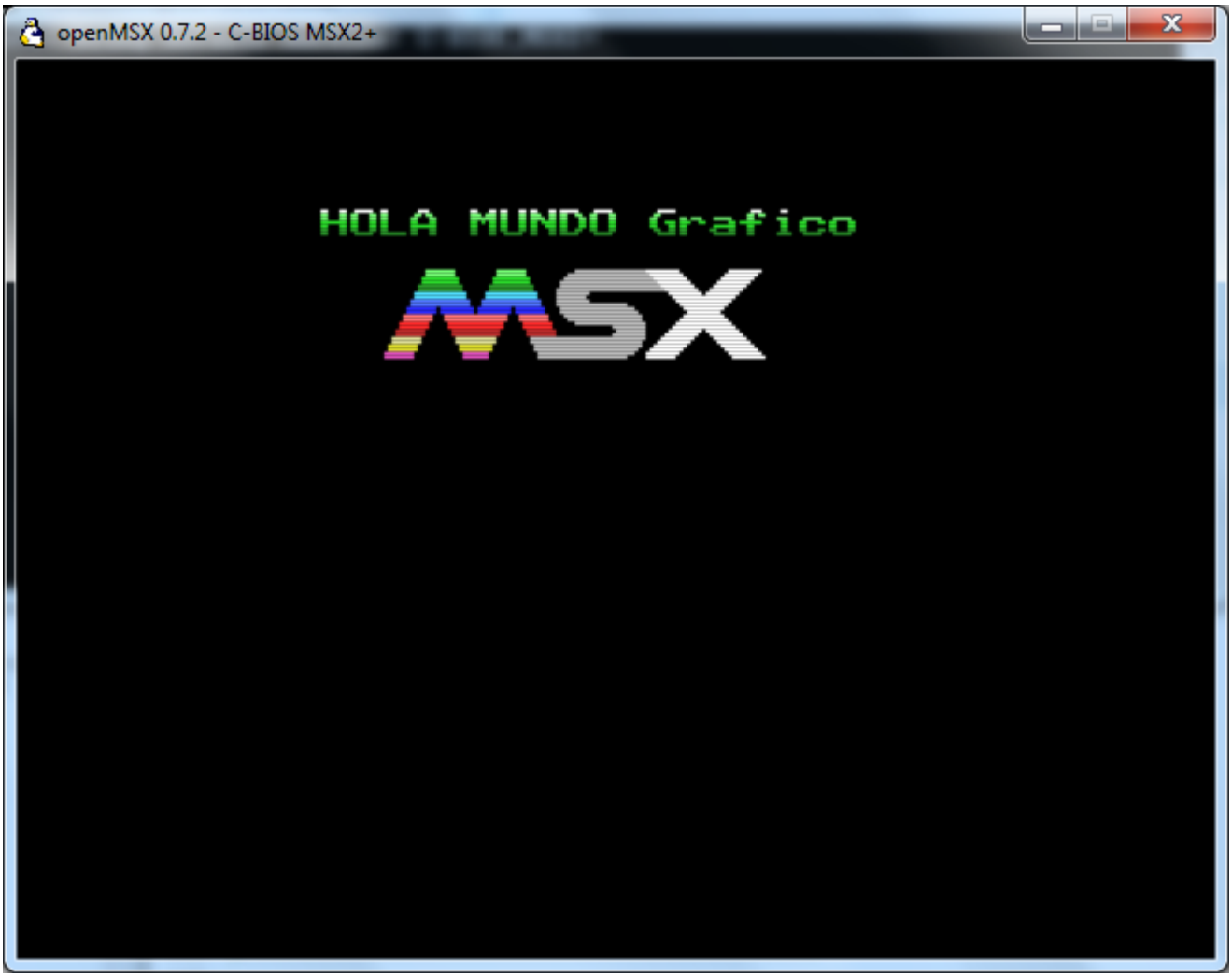
TUTORIAL DE INICIACION A LA PROGRAMACION EN LENGUAJE ENSAMBLADOR PARA MSX

4ª PARTE – EL MUNDO DE LOS GRAFICOS 2

En esta parte del tutorial vamos a retomar el código del Hola Mundo Grafico de la tercera entrega y añadirle un colorido especial a nuestro set de caracteres, además incorporaremos un grafico o logotipo y veremos cómo trabajarlo desde el código, así como mostrarlo en la pantalla, volveremos a repasar procesos dados en la tercera entrega, hasta conseguir la imagen que vemos debajo de este texto.

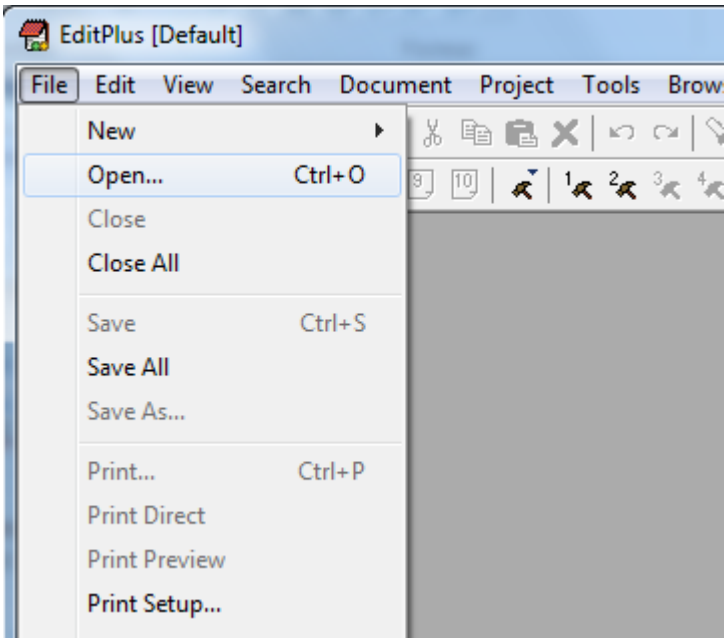


Este será el resultado final de nuestro nuevo tutorial, pero antes de llegar aquí veremos una opción intermedia. A esta versión intermedia la llamaré “HolaMundoGrafico2.asm” que tenéis incluida en los ficheros suministrados para este tutorial aquí. Tutorial4.rar
http://www.dimensionzgames.com/wp-content/uploads/downloads/2012/02/Tutorial4.rar

Quiero comentar que el grafico del Logo del MSX lo he creado y dibujado en GIMP y después en la segunda parte veremos cómo importar nuestros gráficos al programa nMSXtiles.
El colorido que le he impreso al logo del MSX es una referencia a la paleta original del MSX empleando los 16 colores originales de nuestro querido estándar.

Todo lo que aprendáis en esta entrega os servirá para crearos vuestros menús o pantallas de inicio de tus ROM´s, o si vuestro objetivo es crear un videojuego las pantallas de introducción.

Vamos con el Código del HolaMundoGrafico2.asm



Ejecuta el EditPlus 3 desde el menú inicio

Pulsa en los menús en…
File – Open

Abre el fichero HolaMundoGrafico1.asm de la tercera entrega del tutorial.

Cuando lo tengas abierto antes de modificar este fichero realiza lo siguiente

Pulsa en los menús en File – Save As… y lo guardas en el directorio que quieras y con el nombre que quieras. Yo le pongo HolaMundoGrafico2.asm

Este es el código del HolaMundoGrafico1 que ahora hemos grabado como HolaMundoGrafico2.asm

```
;-------------------------------------------------------
; Nombre de nuestro programa
; Hola Mundo Grafico - 24/09/2011
; Versión 1
;-------------------------------------------------------
```

Lo primero es cambiar en el código el nombre de nuestro proyecto, modificáis las líneas de la fecha de creación y el número de versión.

```
;-------------------------------------------------------
; Nombre de nuestro programa
; Hola Mundo Grafico - 28/09/2011
; Versión 2
;-------------------------------------------------------
```

Lo segundo que vamos modificar ya que queremos dar un color especial a todo nuestro set de letras es este apartado que te pongo debajo de este texto y que te explique en la tercera entrega.

```
; Colocar el color de las letras en VRAM en la CLRTBL
        ld      hl,CLRTBL+(32*8)        ; Empezar en el CHR 32 de la CLRTBL
        ld      bc,(32*24)              ; numero de caracteres
        ld      a,0Fh                   ; Valor a rellenar
        call    FILVRM                  ; BIOS -Fill block of VRAM with data byte
```

La rutina FILVRM está situada en la BIOS y tiene unos parámetros de entrada, que son los que te describo a continuación:
El registro HL tiene que apuntar a la dirección de memoria de la VRAM donde queremos empezar a colocar los bytes
En el ejemplo que nos ocupa queremos empezar a colocar bytes en el carácter 32 de la Colour Table – CLRTBL

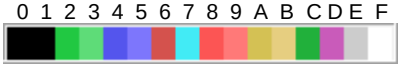
```
        ld      hl,CLRTBL+(32*8)        ; Empezar en el CHR 32 de la CLRTBL
```

En el registro BC le decimos cuantos son los bytes que hay que rellenar.
El set de caracteres ocupa 32 caracteres de ancho por 3 caracteres de alto, pero recuerda que cada CHR tiene de alto 8 bytes, entonces hay que multiplicar 3 x 8=24 bytes de altura, por eso en el registro BC pongo 32 x 24

```
        ld      bc,(32*24)              ; numero de caracteres
```

En el registro A le decimos el byte que tiene que usar para rellenar toda la zona seleccionada.
En el ejemplo que nos ocupa el Color de la tinta es blanco F y el fondo negro 0, o lo que es lo mismo 0Fh en hexa. 16 Colores de fondo y 16 colores de tinta, recuerda que los colores empiezan con el 0 y terminan en el 15 si mira la paleta del MSX veras que el blanco es el 15 y el negro el 0 - 15 en hexadecimal es F y el negro 0, resultado 0Fh si quieres cambiar colores ya sabes cómo tienes que hacerlo.

```
        ld      a,0Fh                   ; Valor a rellenar
```

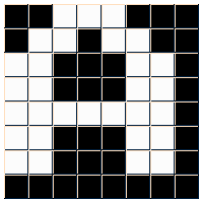


Paleta MSX

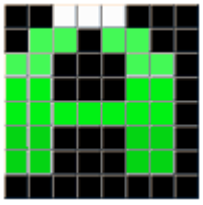
Esta es la llamada a la rutina que empezara escribiendo el byte 0Fh en VRAM empezando en CLRTBL+256 y no parara hasta que se rellenen 32x24=768 Bytes que son las 3 filas de 32 CHR que ocupa el set de caracteres en la Color Table - CLRTBL

```
        call    FILVRM                  ; BIOS -Fill block of VRAM with data byte
```

Haciendo esto rellenábamos los 96 CHRs cada uno con sus 8 bytes en la Colour Table - CLRTBL para dar color blanco fondo negro a todas las letras de nuestro set de caracteres.



Byte 0 - 0Fh
Byte 1 - 0Fh
Byte 2 - 0Fh
Byte 3 - 0Fh
Byte 4 - 0Fh
Byte 5 - 0Fh
Byte 6 - 0Fh
Byte 7 - 0Fh



Byte 0 - 0Fh
Byte 1 - 03h
Byte 2 - 03h
Byte 3 - 02h
Byte 4 - 02h
Byte 5 - 0Ch
Byte 6 - 0Ch
Byte 7 - 0Ch

Este es el nuevo color que vamos a darle a nuestro set de caracteres.

Puedes probar combinaciones de colores. y darle tu mismo el colorido que quieras. pero deberás modificar la en el código los valores en TBL_MICLR y poner los valores que queráis utilizar.

Ahora vamos con lo que hay que cambiar o quitar dentro del código en ensamblador.

Esta es la parte que tenéis que quitar o borrar en el código, o bien modificarlo con el código de abajo.

```
; Colocar el color de las letras en VRAM en la CLRTBL
        ld      hl,CLRTBL+(32*8)        ; Empezar en el CHR 32 de la CLRTBL
        ld      bc,32*24                ; numero de caracteres
        ld      a,0Fh                   ; Valor a rellenar
        call    FILVRM                  ; BIOS -Fill block of VRAM with data byte
```

En el mismo lugar del código que has borrado tenéis que colocar este otro fragmento de código.

```
; Colocar el Multi-color de las letras en VRAM en la CLRTBL
        ld      hl,TBL_MICLR            ; Tabla con el CHR de Color
        ld      de,CLRTBL+(32*8)        ; Empezar en el CHR 32 de la CLRTBL
        ld      b,(32*3)                ; Numero de caracteres
        call    COPY_BLOCK              ; Rutina Encargada de realizar el proceso
```

Seguro que ya adiviináis que es lo que va a realizar esta parte del código.
Cargamos HL apuntando a la tabla de 8 bytes del Color que daremos a los CHRs
Cargamos DE apuntando a la dirección CLRTBL+(32*8) en VRAM para que empiece en el CHR nº32
Cargamos el registro B con el número de CHRs que hay que rellenar con la tabla del color.

Y con esos parámetros de entrada llamamos a una nueva rutina que vamos a crear en nuestro código llamada COPY_BLOCK que será la encargada de tomar los 8 bytes de color que componen la tabla, y que vamos a aplicar al set de caracteres en la Colour Table – CLRTBL en VRAM.

A estas alturas solo voy a utilizar los nombres cortos en las descripciones de las zonas en VRAM.

Justo debajo en el apartado del TXT_HOLA añadiremos la tabla del Multi-color que queremos dar a nuestro set de caracteres añadiendo este código.

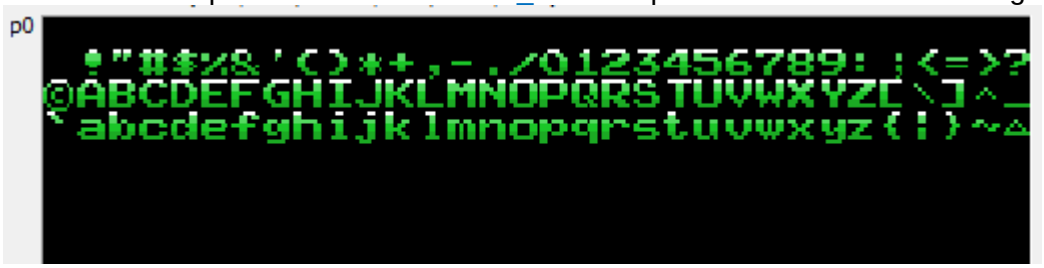
```
; Cadena de Texto a visualizar en la pantalla
TXT_HOLA:
        db      "HOLA MUNDO Grafico"

; Tabla con el Multi-color para los CHRs
TBL_MICLR:
        db      0Fh,03h,03h,02h,02h,0Ch,0Ch,0Ch     ; este es el color que  hemos visto arriba en los bytes de la letra A
```

Esta es la rutina COPY_BLOCK que tienes que añadir justo debajo de lo que acabas de añadir.

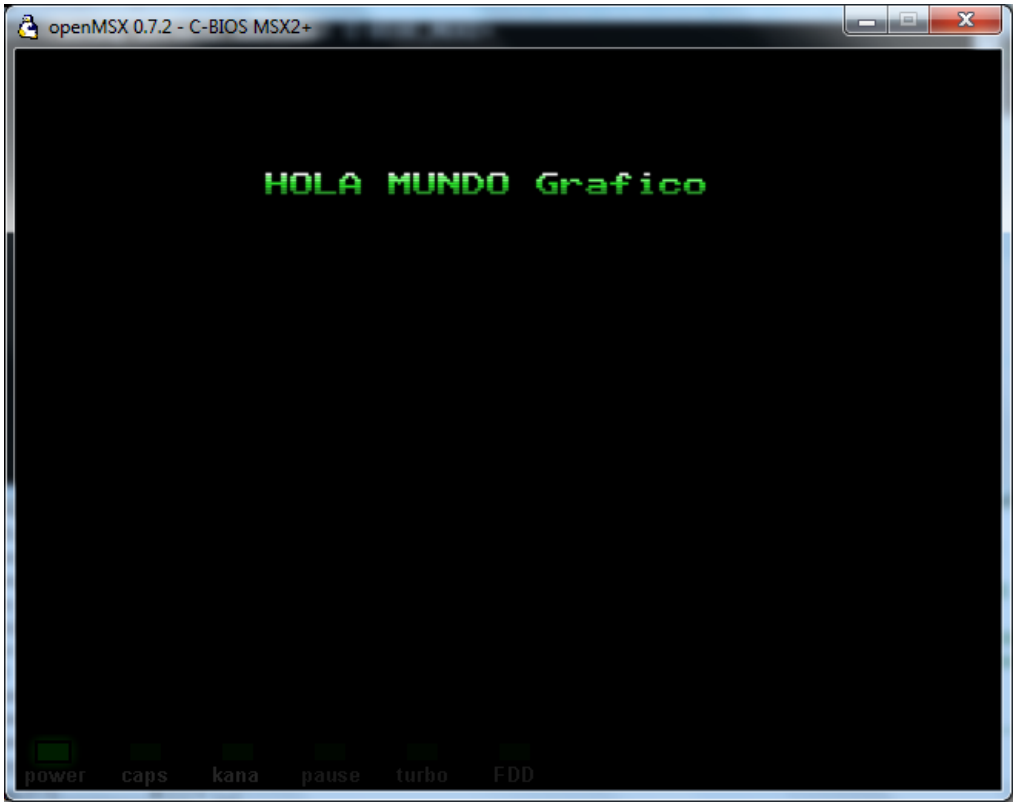
```
;-----------------------------------------------------------------------------
; Copia de manera secuencial bloques de 8 bytes a la VRAM
; Parámetros:  HL = Dirección de Origen en RAM-ROM
;              DE = Dirección de Destino en VRAM
;              B  = Numero de CHRs a copiar máximo 256 CHRs
;-----------------------------------------------------------------------------
COPY_BLOCK:
        push    bc              ; Guardamos cuantos CHRs rellenaremos
        push    hl              ; Guardamos la dirección de origen
        push    de              ; guardamos la dirección de destino
        ld      bc,8            ; copiamos los primeros 8 bytes
        call    LDIRVM          ; BIOS - Copy block to VRAM, from memory
        pop     hl              ; Recuperamos la dirección de destino
        ld      bc,8            ; 8 bytes que sumaremos a la dirección
        add     hl,bc           ; el destino en VRAM será 8 bytes mas
        ex      de,hl           ; pasamos el registro HL al registro DE
        pop     hl              ; recuperamos la dirección de origen
        pop     bc              ; recuperamos el numero de CHRs que quedan
        djnz    COPY_BLOCK      ; restamos 1 al número de CHRs y repetimos…
                                ; …todo el proceso hasta llegar a cero CHRs
        ret                     ; salimos cuando esté finalizado
;-----------------------------------------------------------------------------
```

Como puedes observar en la rutina que está muy comentadita, se llama a la rutina en BIOS LDIRVM encargada de llevar bytes de ROM-RAM a VRAM en este caso va copiando de 8 en 8 bytes la tabla del color a cada carácter en la CLRTBL finalizando el bucle cuando el numero de CHRs sea cero.

La CLRTBL después de llamar a COPY_BLOCK quedará como ves en la imagen. Banco 0



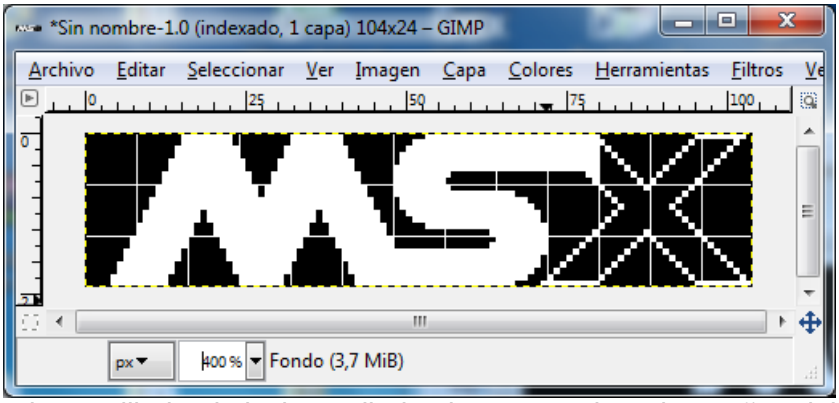
Ya puedes grabar el fichero compilar y ejecutar nuestro código HolaMundoGrafico2.asm



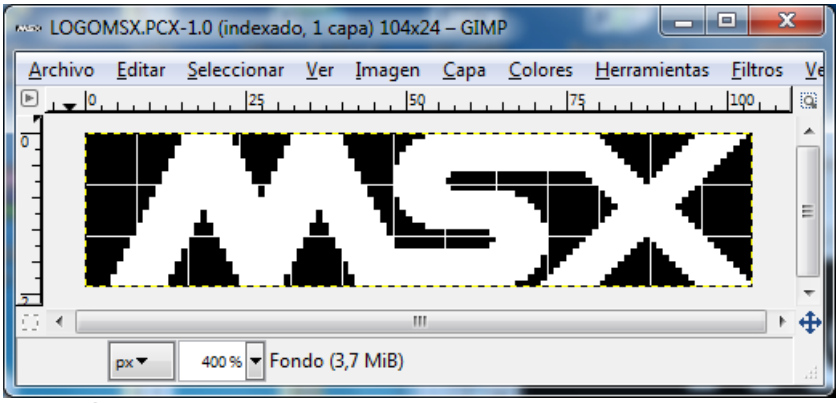
Y voila, a partir de ahora nuestros textos aparecen ya con el Multi-color. Elegido

Veis como era muy fácil de implementar el color.

Vamos ahora a explicar la parte que mostrara el logotipo o grafico junto al Hola Mundo.
Lo primero será crear el logotipo o grafico que vamos a insertar en la pantalla. Así que abre el GIMP y manos a la obra, crea un grafico como te enseñe en la 3ª entrega cuando realizamos el set de caracteres. Os pongo unos ScreenCaps del proceso de creación. El fichero se llama LOGOMSX2.pcx



Primero dibujo al pixel, ampliado al 1600% sobre el tamaño original, creando el cuerpo del dibujo.



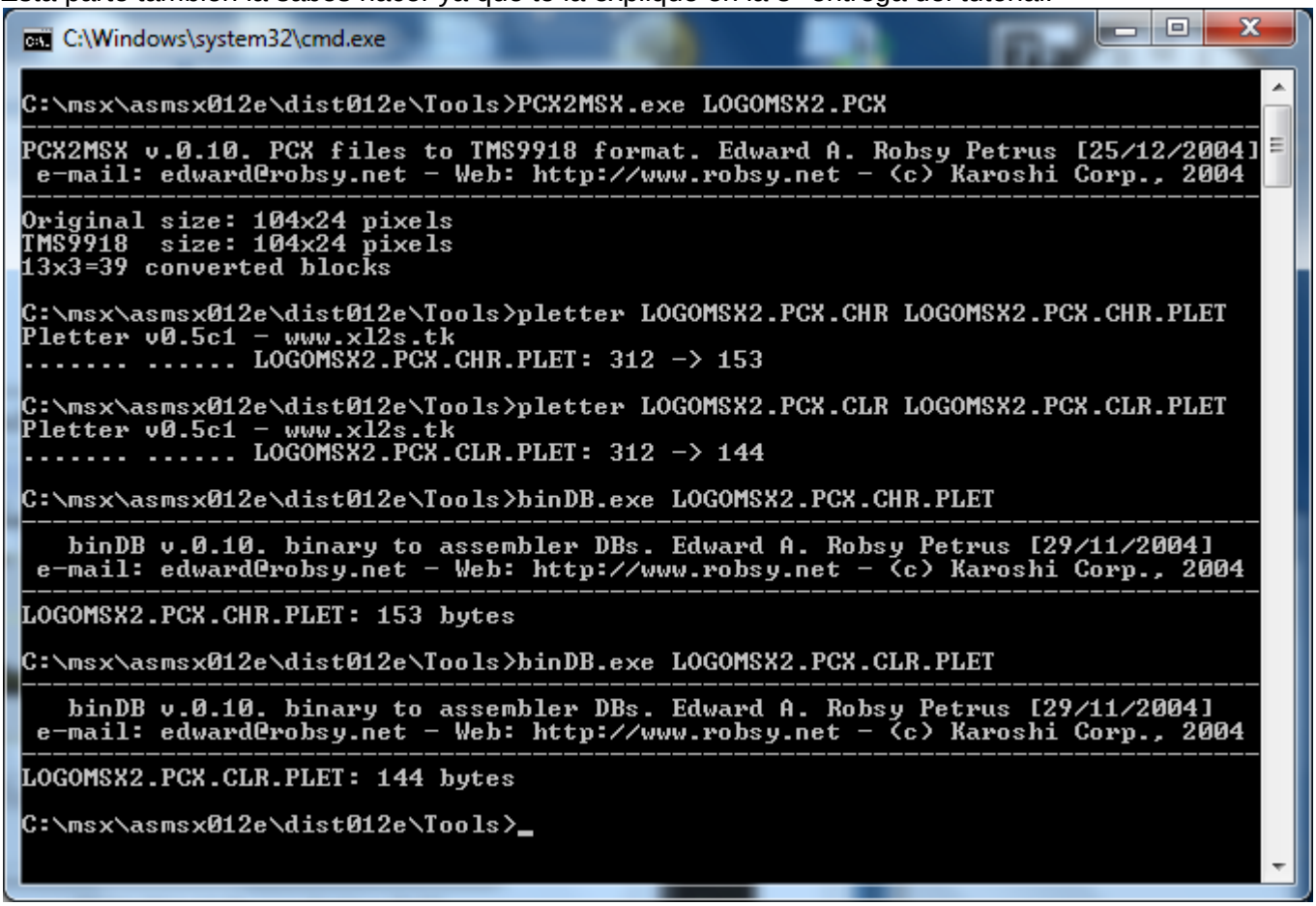
Después con la herramienta de relleno realizo un rellenado interior en color blanco.

Finalmente le doy el color al logotipo con la precaución de no utilizar más de dos colores por CHR con la ayuda del Grid 8x1 en Gimp. El PCX2MSX te avisará del Error si empleas más de 2 colores por byte.

Ojo con la paleta de colores ya que cada vez que creamos una nueva imagen GIMP toma la paleta por defecto, en el menú Colores – Mapa – Establecer el mapa de colores pulsas en Default y selecciona de la lista de paletas la mía VDP-MSX-TMS9918-(Paleta PEPE) pulsa el botón aceptar y listo. Esto te lo explico si no te quieres llevar sorpresas después, cuando veas el color en la pantalla del MSX.

Guardas el fichero de imagen en formato PCX con el nombre de LOGOMSX2.PCX en el directorio donde tienes las herramientas, pletter.exe, pcx2msx.exe y el bindb.exe.

Esta parte también la sabes hacer ya que te la explique en la 3ª entrega del tutorial.



Te muestro esta captura para que veas todo el proceso.

1 - Generamos los ficheros binarios separando los CHRs y los CLRs con el PCX2MSX.exe

2 - Comprimimos el fichero binario LOGOMSX2.PCX.CHR con pletter que son los caracteres.

3 - Comprimimos el fichero binario LOGOMSX2.PCX.CLR con pletter que son los colores de los CHRs.

4 - Convertimos el Binario comprimido de los CHRs a formato ensamblador con el binDB.exe

5 - Convertimos el Binario comprimido de los CLRs a formato ensamblador con el binDB.exe

---

Ya tenemos creado nuestros ficheros .asm con los bytes comprimidos de nuestro logo.
El Primero con los caracteres LOGOMSX2.PCX.CHR.PLET.asm al que le llamaremos LOGO_CHR:

```
; LOGOMSX2.PCX.CHR.PLET - binary dump
; generated by binDB v.0.10
LOGO_CHR:
  db 01Eh,0FFh,000h,000h,0FEh,0FEh,0FCh,0FCh
  db 0F8h,0F8h,0F0h,01Ah,0F0h,001h,001h,0C1h
.....
```

El Segundo con los colores LOGOMSX2.PCX.CLR.PLET.asm al que le llamaremos LOGO_CLR:

```
; LOGOMSX2.PCX.CLR.PLET - binary dump
; generated by binDB v.0.10
LOGO_CLR:
  db 01Eh,000h,000h,000h,003h,003h,002h,002h
  db 00Ch,00Ch,007h,040h,007h,007h,022h,022h
.....
```

Estas etiquetas se las ponemos al principio de cada binario para después por código localizar estos datos, y que deberéis añadir al código del HolaMundoGrafico3.asm que vamos a crear.

Repetimos el proceso descrito al principio del tutorial abre el HolaMundoGrafico2.asm y guarda el código como HolaMundoGrafico3.asm y modifica la versión y la fecha, ya que vamos a modificar este código para añadir el logo del MSX.

Vamos con lo nuevo que hay que añadir al código:
Justo debajo de colocar la cadena de texto agregaremos los nuevos fragmentos de código.

```
; Colocar la cadena de texto en la pantalla
; LOCATE 6,2: PRINT "Hola Mundo Grafico"
        ld      hl,TXT_HOLA             ; Dirección donde tenemos  el texto
        ld      de,NAMTBL+6+(2*32)      ; LOCATE 6,2 : Destino la NAMBTL en VRAM
        ld      bc,18                   ; Numero de CHR que tiene el texto
        call    LDIRVM                  ; BIOS - Copy block to VRAM, from memory
```

Coloca estos dos grupos de código en este punto.

```
; Colocar el logo del msx
        ld      hl,LOGO_CHR             ; Dirección de los CHR del logo MSX
        ld      de,CHRTBL+(128*8)       ; los situaremos a partir del CHR n°128 en la CHRTBL
        call    DEPLET                  ; Descomprimir en VRAM

; Colocar el color del logo del msx
        ld      hl,LOGO_CLR             ; Dirección de los CLR del logo MSX
        ld      de,CLRTBL+(128*8)       ; los situaremos a partir del CHR n°128 en la CHRTBL
        call    DEPLET                  ; Descomprimir en VRAM
```

Las explicaciones de esta parte son muy similares a lo explicado en la 3ª entrega del tutorial, donde descomprimimos el set de caracteres.

Básicamente lo que realiza este apartado es descomprimir los bytes de los CHRs del logo colocándolos en la CHRTBL empezando en el primer CHR de la 5ª Fila o lo que es lo mismo en el CHR nº128

El segundo apartado descomprime los bytes de los colores de los CHRs y los colocamos en la CLRTBL empezando en el primer CHR de la 5ª Fila o lo que es lo mismo en el CHR nº128.

Os pongo una imagen con el resultado final la CHRTBL y la CLRTBL, el 1º tercio o banco quedaría así.

Este apartado te lo explico por si quieres modificar los tutoriales o colocar texto o gráficos en cualquier zona de la pantalla para vuestras futuras creaciones.

Recuerdas que en la 3ª entrega os comente que la VRAM está dividida en tres bancos de 256 CHRs cada uno, solo podemos colocar datos en el primer tercio de la pantalla, ya que solo hemos colocado CHRs en el primer banco, si queremos mostrar textos o gráficos en toda la pantalla debemos colocar o repetir los mismos caracteres en los 3 tercios o bancos de la CHRTBL y CLRTBL.

No es muy difícil y con lo que has aprendido tu mismo ya lo puedes hacer te doy una pista.
256 CHRs tiene un banco multiplicado por 8 bytes que tiene cada CHR nos da un total de 2048 Bytes.

```
; Colocar los gráficos de las letras en VRAM en la CHRTBL en el tercio 2
        ld      hl,SET2_PLET            ; set de CHR de las letras
        ld      de,CHRTBL+2048+(32*8)   ; Empezar en el CHR 32 del banco 1
        call    DEPLET                  ; Descomprimir en VRAM

; Colocar los gráficos de las letras en VRAM en la CHRTBL en el tercio 3
        ld      hl,SET2_PLET            ; set de CHR de las letras
        ld      de,CHRTBL+4096+(32*8)   ; Empezar en el CHR 32 del banco 2
        call    DEPLET                  ; Descomprimir en VRAM
```

Banco 0
CHRTBL
CLRTBL

TERCIO 1

Banco 1
CHRTBL+2048
CLRTBL+2048

TERCIO 2

Banco 2
CHRTBL+4096
CLRTBL+4096

TERCIO 3

Haciendo esto ya podrías poner texto o el logo del MSX en cualquier Fila y Columna de la pantalla. Seguro que tu mismo podrás crear el HolaMundoGrafico4.asm si quieres probar cosas.

Sigamos con el HolaMundoGrafico3 ya tenemos los CHRs y los CLRs en la VRAM ahora nos queda colocar en la NAMTBL los números de los CHRs que pertenecen al logo del MSX para que este se muestre en la pantalla, casi de la misma manera como lo hacemos con el texto.

El logo del MSX está compuesto por 13 CHRs de ancho por 3 líneas de CHRs de alto y como hemos colocado el primer CHR del logo en la CHRTBL en el numero 128 haremos lo siguiente.

```
Locate 8,4; Print chr$(128);chr$(129) ;chr$(130) ;chr$(131) ;chr$(132) ;chr$(133) ;chr$(134) ;chr$(135)
;chr$(136) ;chr$(137) ;chr$(138) ;chr$(139) ;chr$(140)

; Colocar los 13 primeros CHRs del logo
        ld      hl,NAMLOGO              ; Primera fila del mapeado
        ld      de,NAMTBL+8+(4*32)      ; LOCATE 8,4
        ld      bc,13                   ; Numero de CHRs
        call    LDIRVM                  ; BIOS - Copy block to VRAM, from memory

Locate 8,5; Print chr$(141);chr$(142) ;chr$(143) ;chr$(144) ;chr$(145) ;chr$(146) ;chr$(147) ;chr$(148)
;chr$(149) ;chr$(150) ;chr$(151) ;chr$(152) ;chr$(153)

; Colocar los 13 segundos CHRs del logo
        ld      hl,NAMLOGO+13           ; Segunda fila del mapeado
        ld      de,NAMTBL+8+(5*32)      ; LOCATE 8,5
        ld      bc,13                   ; Numero de CHRs
        call    LDIRVM                  ; BIOS - Copy block to VRAM, from memory

Locate 8,6; Print chr$(154);chr$(155) ;chr$(156) ;chr$(157) ;chr$(158) ;chr$(159) ;chr$(160) ;chr$(161)
;chr$(162) ;chr$(163) ;chr$(164) ;chr$(165) ;chr$(166)

; Colocar los 13 terceros CHRs del logo
        ld      hl,NAMLOGO+13+13        ; Tercera fila del mapeado
        ld      de,NAMTBL+8+(6*32)      ; LOCATE 8,6
        ld      bc,13                   ; Numero de CHRs
        call    LDIRVM                  ; BIOS - Copy block to VRAM, from memory
```

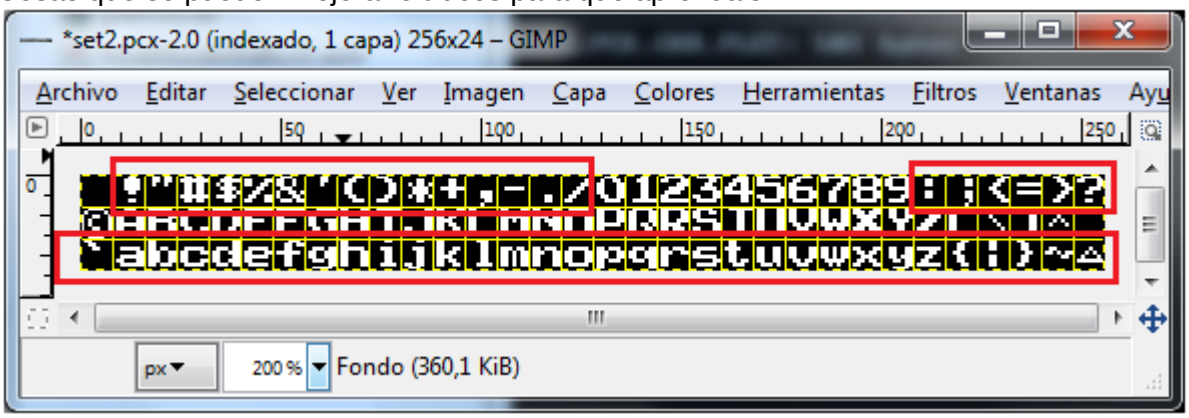
Por eso he creado una tabla llamada NAMLOGO que toma los valores de los bytes que colocaremos en la NAMTBL justo debajo de los bytes comprimidos y del color del Logo del MSX. (ver mas abajo)

Al final de los bytes comprimidos de set de caracteres es donde debes agregar los bytes comprimidos del Logo del MSX que previamente os he explicado.

```
;-----------------------------------------
; LOGOMSX2.PCX.CHR.PLET - binary dump
; generated by binDB v.0.10
;-----------------------------------------
LOGO_CHR:
 db 01Eh,0FFh,000h,000h,0FEh,0FEh,0FCh,0FCh
 db 0F8h,0F8h,0F0h,01Ah,0F0h,001h,001h,0C1h
 db 011h,07Ch,07Ch,038h,038h,0F4h,00Dh,000h
 db 07Fh,000h,07Fh,03Fh,03Fh,01Fh,01Fh,0F8h
 db 0E0h,0C0h,035h,080h,080h,0B5h,013h,080h
 db 01Eh,080h,0C0h,0E0h,0F0h,0F8h,0FCh,050h
 db 0FEh,012h,024h,00Fh,007h,003h,055h,001h
 db 04Eh,04Ch,02Ah,001h,029h,001h,003h,007h
 db 00Fh,01Fh,03Fh,0ACh,027h,0E0h,0B5h,014h
 db 000h,040h,020h,020h,070h,070h,010h,010h
 db 0D6h,042h,005h,008h,008h,01Ch,01Ch,00Fh
 db 03Ch,03Ah,03Dh,003h,085h,025h,09Fh,051h
 db 0F8h,04Ch,061h,057h,01Fh,054h,0FBh,09Bh
 db 0AEh,00Fh,01Dh,05Eh,0EEh,05Fh,06Eh,07Fh
 db 0FFh,055h,0C7h,019h,02Dh,047h,055h,04Bh
 db 00Bh,00Fh,013h,0F4h,01Dh,013h,03Eh,03Eh
 db 03Eh,07Fh,08Ch,0E6h,061h,0C0h,0E6h,051h
 db 0B4h,024h,037h,01Eh,0F3h,0BEh,05Bh,0BFh
 db 0CFh,0DFh,01Fh,0DEh,0FFh,0FFh,0FFh,0FFh
 db 0F8h

;          .INCBIN             "LOGOMSX2.PCX.CHR.PLET"
;-----------------------------------------

;-----------------------------------------
; LOGOMSX2.PCX.CLR.PLET - binary dump
; generated by binDB v.0.10
;-----------------------------------------
LOGO_CLR:
 db 01Eh,000h,000h,000h,003h,003h,002h,002h
 db 00Ch,00Ch,007h,040h,007h,007h,022h,022h
 db 0CCh,0CCh,077h,05Bh,077h,011h,01Eh,00Fh
 db 033h,033h,0CEh,00Fh,00Eh,000h,07Ch,0EEh
 db 000h,00Eh,06Dh,000h,007h,01Ch,01Dh,0FFh
 db 0EFh,000h,00Fh,01Eh,00Fh,000h,00Fh,08Fh
 db 000h,0FFh,01Fh,007h,000h,000h,000h,005h
 db 005h,004h,004h,009h,009h,088h,004h,088h
 db 055h,055h,044h,044h,007h,008h,051h,008h
 db 00Fh,007h,099h,099h,0BDh,00Fh,0D5h,01Fh
 db 007h,05Bh,0D3h,069h,05Fh,00Eh,0EFh,068h
 db 03Ch,078h,046h,0F0h,058h,0FFh,0FFh,0FBh
 db 067h,0E0h,05Fh,006h,006h,00Bh,00Bh,003h
 db 00Ah,00Ah,00Dh,00Dh,066h,066h,06Dh,007h
 db 0A8h,00Fh,017h,00Fh,0BBh,0BBh,036h,0AAh
 db 0AAh,00Fh,0EBh,027h,00Fh,067h,061h,000h
 db 0DBh,0C1h,09Ch,0D4h,000h,00Fh,0F6h,05Fh
 db 0BFh,0DFh,03Fh,0D7h,0FFh,0FFh,0FFh,0F0h

;          .INCBIN             "LOGOMSX2.PCX.CLR.PLET"
;-----------------------------------------

;-----------------------------------------------
; tabla con los CHRs del logoMSX
NAMLOGO:
   db  128,129,130,131,132,133,134,135,136,137,138,139,140
   db  141,142,143,144,145,146,147,148,149,150,151,152,153
   db  154,155,156,157,158,159,160,161,162,163,164,165,166
;-----------------------------------------


;-----------------------------------------
; FINAL DE NUESTRO CODIGO EN ENSAMBLADOR.
;-----------------------------------------
```

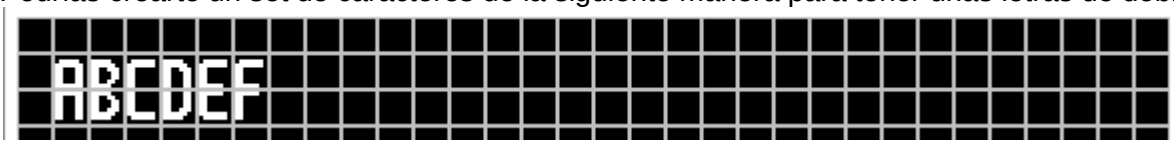
Ahora ya puedes grabar, compilar y ejecutar el Código del HolaMundo3.asm el resultado final ya lo conoces es la primera imagen de este manual.

Cosas que se pueden mejorar o trucos para que aprendáis.



Las letras en minúsculas no se suelen utilizar, puedes utilizar esos 32 CHRs para colocar más gráficos, o incluso para colocar el LOGO del MSX en esos CHRs al igual que con los símbolos otros 21 CHRs salvo que quieras usarlos. El resto de CHRs debe ir en su sitio. (Por la Norma del ASCII)
Yo añado en uno de los CHRs no usados en el ultimo, nuestra querida letra Ñ para usarla.

Podrías crearte un set de caracteres de la siguiente manera para tener unas letras de doble tamaño.



En las Mayúsculas creas la parte de arriba de las letras y en las Minúsculas creas la parte de abajo.
A la hora de crear el menú o colocar los CHRs en la NAMTBL en el código harías lo siguiente.

```
; Cadena de Texto a visualizar en la pantalla
TXT_MA:
        db      "ABCDEF"
TXT_MI:
        db      "abcdef"

; Colocar la cadena de texto en la pantalla
; LOCATE 6,2: PRINT "ABCDEF"
        ld      hl,TXT_MA               ; Dirección donde tenemos  el texto
        ld      de,NAMTBL+6+(2*32)      ; LOCATE 6,2 : Destino la NAMBTL en VRAM
        ld      bc,6                    ; Numero de CHRs que tiene el texto
        call    LDIRVM                  ; BIOS – Copy block to VRAM, from memory

; LOCATE 6,3: PRINT "abcdef"
        ld      hl,TXT_MI               ; Dirección donde tenemos  el texto
        ld      de,NAMTBL+6+(3*32)      ; LOCATE 6,3 : Destino la NAMBTL en VRAM
        ld      bc,6                    ; Numero de CHRs que tiene el texto
        call    LDIRVM                  ; BIOS – Copy block to VRAM, from memory
```

El resultado en pantalla sería el de la imagen que te pongo aquí debajo con Multi-color o sin el.

Os dejo a vosotros que experimentéis con esto… a ver si veo vuestros resultados en los Foros.

Espero que haya sido de vuestro total agrado y nos vemos en el próximo tutorial para finalizar con el mundo de los gráficos. Donde veremos otra forma de crear menús con texto y gráficos, importar imágenes al nMSXtiles y trabajar con este.

José Vila Cuadrillero

"ES DETESTABLE ESA AVARICIA ESPIRITUAL QUE TIENEN, LOS QUE SABIENDO ALGO, NO PROCURAN LA TRANSMISION DE ESOS CONOCIMIENTOS."
Miguel de Unamuno
Escritor y Filósofo.
(Bilbao 1864 - Salamanca 1936)