C256 – Developer Introduction Notes

# Introduction to the C256 System

The C256 system uses a 65C816 micro-processor.

System clock is 14MHz.



# Git Repositories

Nu256 Emulator: <https://github.com/tomxp411/Nu256.git>

C256 Kernel: <https://github.com/Trinity-11/Kernel>

# Tools

To modify the Nu256 Emulator, you will need Visual Studio 2017 Community edition.

The emulator is written in C#.

# Memory Map

The CPU can access 24-bit worth of addresses.



|  |  |  |
| --- | --- | --- |
| $FF:0000 - $FF:FFFF | Bank $FF | **16 MB Address Space** |
| $FE:0000 - $FE:FFFF | Bank $FE |
|  |  |
|  |  |
|  |  |
|  |  |
| $00:0000 - $01:FFFF | Bank $01 |
| $00:0000 - $00:FFFF | Bank $00 |

The address space is mapped as follows:

|  |  |
| --- | --- |
| $F8:0000 - $FF:FFFF | 512 KB User Flash (if populated) |
| $F0:0000 - $F7:FFFF | 512 KB System Flash |
| $B0:0000 - $EF:FFFF | 4 MB Video RAM |
| $AF:0000 - $AF:FFFF | IO Space |
| $40:0000 - $AE:FFFF | <empty> |
| $20:0000 - $3F:FFFF | 2 MB RAM (optional) |
| $00:0000 - $1F:FFFF | 2 MB RAM |

On boot, Gavin copies the first 64KB of the content of System Flash (or User Flash, if present) to Bank $00. The entire 512KB are copied to address range $18:0000 to $1F:FFFF.

IO Space is mapped to Vicky: $AF:0000 to $AF:DFFF and Beatrix: $AF:E000 to $AF:FFFF.

## Gavin – Location $00:0000 to $00:FFFF



## Vicky – Location $AF:0000 to $AF:DFFF



### Screen Page 0 – Location $AF:A000

Screen Page 0 memory is used to store text characters for display.

One page of text is 128 columns by 64 rows. This adds up to 8 KB of memory of text. C256 does not display the entire buffer on the screen. Typically, we render 72 characters per row, with 56 rows.

This uses 576 x 448 of the available 640 x 480 resolution. The border size can be modified or turned off completely.

The display process reads Screen Page 0 and for each character, displays it’s character set bitmap.

### Screen Page 1 – Location $AF:C000

An additional page of 128 x 64 is used to store the colors. Each byte is split into foreground (4bits) and background (4 bits). The high nibble (bits 7..4) are the foreground and the low nibble (bits 3..0) are the background.

The colors used (the 4 bits) are used to lookup RBG values in two lookup tables (LUT).

The foreground (FG) LUT is located at $AF:1F40 for 64 bytes – only 16 x 3 = 48 bytes are used. The extra byte may be used for alpha (transparency) later on.

The background (BG) LUT is located at $AF:1F80 for 64 bytes – only 16 x 3 = 48 bytes are used. The extra byte may be used for alpha (transparency) later on.

The colors are assigned 8-bit blue, 8-bit green, 8-bit red, 8-bit alpha (not used) for each of those colors in Text Mode.

#### Example – Color Lookup

Consider the following Foreground and Background Lookup Tables

|  |  |
| --- | --- |
| Foreground Color Lookup Table, starting at address $AF:1F40 | Background Color Lookup Table, starting at address $AF:1F80 |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | Index | Blue | Green | Red | Alpha | Color | | 0 | $00 | $00 | $00 | $FF | Black | | 1 | $00 | $00 | $80 | $FF | Maroon | | 2 | $00 | $80 | $00 | $FF | Green | | 3 | $80 | $00 | $00 | $FF | Navy | | 4 | $00 | $80 | $80 | $FF | Olive | | 5 | $80 | $80 | $00 | $FF | Teal | | 6 | $80 | $00 | $80 | $FF | Purple | | 7 | $80 | $80 | $80 | $FF | Gray | | 8 | $00 | $45 | $FF | $FF | Orange | | 9 | $13 | $45 | $8B | $FF | Brown | | A | $00 | $00 | $20 | $FF | Dark Red | | B | $00 | $20 | $00 | $FF | Dark Green | | C | $20 | $00 | $00 | $FF | Indigo | | D | $20 | $20 | $20 | $FF | Dark Gray | | E | $40 | $40 | $40 | $FF | Slate Gray | | F | $FF | $FF | $FF | $FF | White | | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | Index | Blue | Green | Red | Alpha | Color | | 0 | $00 | $00 | $00 | $FF | Black | | 1 | $00 | $00 | $80 | $FF | Maroon | | 2 | $00 | $80 | $00 | $FF | Green | | 3 | $80 | $00 | $00 | $FF | Navy | | 4 | $00 | $20 | $20 | $FF | ?? | | 5 | $20 | $20 | $00 | $FF | ?? | | 6 | $20 | $00 | $20 | $FF | ?? | | 7 | $20 | $20 | $20 | $FF | ?? | | 8 | $1E | $69 | $D2 | $FF |  | | 9 | $13 | $45 | $8B | $FF | Brown | | A | $00 | $00 | $20 | $FF | Dark Red | | B | $00 | $20 | $00 | $FF | Dark Green | | C | $40 | $00 | $00 | $FF | Blue | | D | $10 | $10 | $10 | $FF | Midnight Gray | | E | $40 | $40 | $40 | $FF | Slate Gray | | F | $FF | $FF | $FF | $FF | White | |

If a character in Screen Page 1 is $ED (the default text color combination), then the foreground color index is E and the background color index is D. Looking up the index for E will make the foreground “Slate Gray” and the background “Midnight Gray”. The image below shows this color combination in text.



### Gamma Lookup Table

The Gamma lookup table is used to adjust the color between different display devices (such as DVI versus VGA). Each of the red, green and blue can be corrected. Each table consists of 256 values.

GAMMA\_B\_LUT\_PTR = $AF:4000

GAMMA\_G\_LUT\_PTR = $AF:4100

GAMMA\_R\_LUT\_PTR = $AF:4200

Gamma can be enabled or disabled.

Master Control Register

The Master Control Register (MCR) is used to enable/disable various video mode. The MCR is located at address $AF:0000.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Master Control Register ($AF:0000)** | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Disable Vid | Gamma | Sprite | Tilemap | Bitmap | Graph Mode | Text Overlay | Text Mode |

|  |  |  |
| --- | --- | --- |
| **MCR Bit** | **MCR Name** | **Description** |
| 0 | Mstr\_Ctrl\_Text\_Mode\_En | Enable the Text Mode. Default = 1 |
| 1 | Mstr\_Ctrl\_Text\_Overlay | Enable the Overlay of the text mode on top of Graphic Mode (the Background Color is ignored) |
| 2 | Mstr\_Ctrl\_Graph\_Mode\_En | Enable the Graphic Mode |
| 3 | Mstr\_Ctrl\_Bitmap\_En | Enable the Bitmap Module in Vicky |
| 4 | Mstr\_Ctrl\_TileMap\_En | Enable the Tile Module in Vicky |
| 5 | Mstr\_Ctrl\_Sprite\_En | Enable the Sprite Module in Vicky |
| 6 | Mstr\_Ctrl\_GAMMA\_En | Enable the GAMMA correction - The Analog and DVI have different color value, the GAMMA is great to correct the difference |
| 7 | Mstr\_Ctrl\_Disable\_Vid | This bit disables the Scanning of the Video Memory, so 100% of CPU Bandwidth is allowed |

## Beatrix

