

# GTIA RGB 80-Column Display

## Hardware Reference Manual

Gowin FPGA GTIA Replacement Project

April 2026 — Draft

## Contents

<b>1</b>	<b>Overview</b>	<b>3</b>
1.1	Operating Principle . . . . .	3
<b>2</b>	<b>Register Interface</b>	<b>3</b>
2.1	Mode Control — \$D01D (PMCTL) . . . . .	3
2.1.1	Mode Selection . . . . .	3
2.1.2	COLPF1 Chroma Override (Bit 4) . . . . .	4
2.2	Font Upload Registers — \$D000-\$D003 . . . . .	4
2.2.1	Write Sequence . . . . .	4
<b>3</b>	<b>Display Modes</b>	<b>4</b>
3.1	Mode 1 — Simple (No Attributes) . . . . .	5
3.1.1	Memory Layout Per Row . . . . .	5
3.2	Mode 2 — CGA Palette Attributes . . . . .	5
3.2.1	Memory Layout Per Row . . . . .	5
3.2.2	Attribute Byte Format . . . . .	5
3.2.3	CGA Palette . . . . .	6
3.3	Mode 3 — Atari Chroma + Luma . . . . .	6
3.3.1	Memory Layout Per Row . . . . .	6
3.3.2	Chroma Byte Format . . . . .	6
3.3.3	Luma Byte Format . . . . .	6
3.4	Mode 4 — Full RGB . . . . .	7
3.4.1	Memory Layout Per Row . . . . .	7
3.4.2	RGB Byte Format . . . . .	7
<b>4</b>	<b>Font Memory</b>	<b>7</b>
4.1	Address Space . . . . .	7

4.2	Uploading the Font . . . . .	8
4.2.1	Suggested ATASCII-Order Upload . . . . .	8
4.3	Custom Glyphs . . . . .	8
<b>5</b>	<b>ANTIC Note</b>	<b>8</b>
<b>6</b>	<b>Quick Reference</b>	<b>9</b>
6.1	Mode Summary . . . . .	9
6.2	Font Address Calculation . . . . .	9
<b>A</b>	<b>Program Listings</b>	<b>9</b>
A.1	Font Upload — ATASCII Order (MADS) . . . . .	9

# 1 Overview

The GTIA RGB module is a Gowin FPGA-based replacement for the GTIA chip in the Atari 800XL. When 80-column mode is inactive (bit 6 of `$D01D` cleared), the FPGA operates as a standard GTIA with RGB output. When 80-column mode is enabled, the FPGA intercepts ANTIC's mode F byte stream and renders an 80-column text display with per-character colour attributes.

Key capabilities:

- 80 columns × 24–28 rows of text (8×8 pixel character cells)
- 4 display modes with increasing colour depth
- 256-character font memory (2 KB), uploadable at runtime
- Per-character foreground and background colour attributes
- Compatible with standard ANTIC display lists (mode F based)

## 1.1 Operating Principle

ANTIC generates mode F scan lines (40 bytes each, 320 pixels). The FPGA captures these bytes and interprets them as 80-column character data rather than bitmap pixels. Each text row uses multiple mode F lines: two for character codes (80 bytes), and additional lines for colour attributes depending on the selected mode. Blank scan lines between text rows (`$10–$70`) provide the remaining scan lines needed to fill an 8-pixel-tall character cell.

The FPGA latches one complete row of character and attribute data during the first 8 scan lines, then renders the text row during the following 8 scan lines while simultaneously loading the next row's data into a second buffer (double-buffered design). This means the first mode F instruction (`$4F`) in the display list triggers the start of 80-column display. If a subsequent group of 8 scan lines contains no mode F data, the FPGA treats that row as inactive and displays the background colour.

# 2 Register Interface

## 2.1 Mode Control — `$D01D` (PMCTL)

The 80-column modes are controlled by bits in register `$D01D`, which replaces the standard GTIA GRCTL register when the FPGA is active.

Bit	Name	Function
7	—	Reserved
6	COL80	80-column enable (1 = on, 0 = standard GTIA)
5	ATTRHI	Attribute mode high bit
4	CHROMAPF1	Use COLPF1 chroma for all characters (Mode 1 only)
3	ATTRLO	Attribute mode low bit
2–0	—	Reserved / original GTIA function

### 2.1.1 Mode Selection

The display mode is determined by bits 6, 5, and 3. Bit 6 must be set to enable any 80-column mode.

Mode	Bit pattern	PMCTL value	Description
1	x10x0xxx	\$40	Simple — no attributes
2	x11x0xxx	\$60	CGA palette attributes
3	x10x1xxx	\$48	Atari chroma + luma
4	x11x1xxx	\$68	Full RGB

### 2.1.2 COLPF1 Chroma Override (Bit 4)

When bit 4 is set together with bit 6, the FPGA applies the chroma value from the Atari COLPF1 register (shadow at address 709/\$02C5) to all displayed characters. The background colour comes from COLPF2 (shadow 710/\$02C6). This is useful in Mode 1 for coloured text on a solid background (e.g. green-on-black terminal style) without needing per-character attributes.

## 2.2 Font Upload Registers — \$D000–\$D003

Font memory can only be written when 80-column mode is enabled (bit 6 of \$D01D is set).

Address	Name	Function
\$D000	FONT_ADDRH	Font address bits 10, 9, 8 (3 low bits used)
\$D001	FONT_ADDRL	Font address bits 7–0
\$D002	FONT_DATA	Data byte to write
\$D003	FONT_WR	Write trigger

### 2.2.1 Write Sequence

To write one byte to font memory:

1. Write the high address bits to \$D000 (bits 2–0 used).
2. Write the low address byte to \$D001.
3. Write the data byte to \$D002.
4. Write \$FF to \$D003 (commits the write).
5. Write \$00 to \$D003 (resets the trigger).

The font address space is 2048 bytes (11 bits), organised as 256 characters  $\times$  8 bytes per character. The address for character  $c$ , row  $r$  is:

$$addr = c \times 8 + r \quad (c = 0 \dots 255, r = 0 \dots 7)$$

The three bits in \$D000 select the “page” (group of 32 characters), while \$D001 selects the byte within that page.

## 3 Display Modes

All four modes share the same basic structure: ANTIC mode F lines deliver data to the FPGA, which interprets consecutive lines as character codes and (optionally) colour attributes. Each text row ends with a blank-line instruction to pad the total to 8 scan lines per row.

### 3.1 Mode 1 — Simple (No Attributes)

**PMCTL value:** \$40 (or \$50 with COLPF1 chroma override)  
**DL per row:** \$0F, \$0F, \$50 (+ LMS as needed)  
**Mode F lines:** 2 (80 bytes: characters)  
**Blank lines:** 6 (\$50)  
**Bytes per row:** 80

Mode 1 provides 80-column monochrome text. Each character cell displays the font glyph using a fixed foreground/background colour pair determined by the Atari colour registers (or COLPF1 override if bit 4 is set).

#### 3.1.1 Memory Layout Per Row

Mode F line	Bytes	Content
1	0–39	Character codes, columns 0–39
2	40–79	Character codes, columns 40–79

Character codes in screen memory select glyphs from the FPGA’s font memory. The mapping between codes and glyphs depends entirely on how the font was uploaded (see Section 4). There is no hardware-imposed character ordering — the programmer is free to arrange the font in any order that suits the application.

### 3.2 Mode 2 — CGA Palette Attributes

**PMCTL value:** \$60  
**DL per row:** \$0F×4, \$30 (+ LMS as needed)  
**Mode F lines:** 4 (160 bytes: 80 chars + 80 attributes)  
**Blank lines:** 4 (\$30)  
**Bytes per row:** 160

Mode 2 adds per-character colour attributes using a 16-colour CGA-style palette.

#### 3.2.1 Memory Layout Per Row

Mode F line	Bytes	Content
1	0–39	Character codes, columns 0–39
2	40–79	Character codes, columns 40–79
3	80–119	Attributes, columns 0–39
4	120–159	Attributes, columns 40–79

#### 3.2.2 Attribute Byte Format

Bits	Meaning
7–4	Background colour (0–15, CGA palette)
3–0	Foreground colour (0–15, CGA palette)

### 3.2.3 CGA Palette

Index	Colour	Index	Colour
0	Black	8	Dark grey
1	Blue	9	Light blue
2	Green	10	Light green
3	Cyan	11	Light cyan
4	Red	12	Light red
5	Magenta	13	Light magenta
6	Brown	14	Yellow
7	Light grey	15	White

### 3.3 Mode 3 — Atari Chroma + Luma

**PMCTL value:** \$48  
**DL per row:** \$0F×6, \$10 (+ LMS as needed)  
**Mode F lines:** 6 (240 bytes: 80 chars + 80 chroma + 80 luma)  
**Blank lines:** 2 (\$10)  
**Bytes per row:** 240

Mode 3 provides native Atari colour control with separate chroma (hue) and luma (brightness) attributes per character, supporting the full 256-colour Atari palette.

#### 3.3.1 Memory Layout Per Row

Mode F line	Bytes	Content
1	0–39	Character codes, columns 0–39
2	40–79	Character codes, columns 40–79
3	80–119	Chroma, columns 0–39
4	120–159	Chroma, columns 40–79
5	160–199	Luma, columns 0–39
6	200–239	Luma, columns 40–79

#### 3.3.2 Chroma Byte Format

Bits	Meaning
7–4	Background chroma (hue 0–15)
3–0	Foreground chroma (hue 0–15)

Chroma value 0 produces grey (no colour); values 1–15 select hues matching the standard Atari GTIA colour wheel.

#### 3.3.3 Luma Byte Format

Bits	Meaning
7–4	Background luma (brightness 0–15)
3–0	Foreground luma (brightness 0–15)

Together, chroma and luma select from the full Atari 256-colour palette for both foreground and background of each character cell independently.

### 3.4 Mode 4 — Full RGB

**PMCTL value:** \$68  
**DL per row:** \$0F×8 (+ LMS as needed)  
**Mode F lines:** 8 (320 bytes)  
**Blank lines:** 0  
**Bytes per row:** 320

Mode 4 provides full per-character RGB colour control. All 8 scan lines of each character row are used for data — no blank lines are needed.

#### 3.4.1 Memory Layout Per Row

Mode F line	Bytes	Content
1	0–39	Character codes, columns 0–39
2	40–79	Character codes, columns 40–79
3	80–119	Red, columns 0–39
4	120–159	Red, columns 40–79
5	160–199	Green, columns 0–39
6	200–239	Green, columns 40–79
7	240–279	Blue, columns 0–39
8	280–319	Blue, columns 40–79

#### 3.4.2 RGB Byte Format

Each colour channel (Red, Green, Blue) uses the same nibble packing:

Bits	Meaning
7–4	Background intensity (0–15)
3–0	Foreground intensity (0–15)

With 4 bits per channel per colour (foreground and background), each character cell can display two colours chosen from a palette of  $16^3 = 4096$  colours.

## 4 Font Memory

### 4.1 Address Space

The FPGA contains 2048 bytes of font memory, organised as 256 characters of 8 bytes each. Each byte represents one pixel row of a character, MSB = leftmost pixel. A set bit selects the foreground colour; a clear bit selects the background colour.

The 256 characters are divided into 8 pages of 32 characters:

Page	FONT_ADDRH	Character codes
0	0	\$00–\$1F
1	1	\$20–\$3F
2	2	\$40–\$5F
3	3	\$60–\$7F
4	4	\$80–\$9F
5	5	\$A0–\$BF
6	6	\$C0–\$DF
7	7	\$E0–\$FF

## 4.2 Uploading the Font

The FPGA font memory has no predefined content — after power-on it contains the standard Atari character set in ANTIC internal order (as stored in ROM at \$E000), with normal characters followed by their inverse counterparts. There is no automatic font reset mechanism other than power cycling, so every program that uses 80-column mode should upload the desired font at startup.

The font can be arranged in any order. A convenient convention is to upload the Atari ROM font remapped to ATASCII order, so that screen memory can store ATASCII codes directly without conversion.

### 4.2.1 Suggested ATASCII-Order Upload

The Atari ROM font at \$E000–\$E3FF is stored in internal screen code order. To remap it to ATASCII order, upload four 256-byte blocks with the following source→destination mapping:

ROM source	→	FPGA page (ATASCII codes)
\$E200 (internal \$40–\$5F)	→	Page 0 (\$00–\$1F)
\$E000 (internal \$00–\$1F)	→	Page 1 (\$20–\$3F)
\$E100 (internal \$20–\$3F)	→	Page 2 (\$40–\$5F)
\$E300 (internal \$60–\$7F)	→	Page 3 (\$60–\$7F)

For the inverse character set (ATASCII \$80–\$FF), repeat the same source blocks to pages 4–7, EOR’ing each data byte with \$FF.

An example font upload routine (MADS assembler) is provided in [Appendix A.1](#).

## 4.3 Custom Glyphs

Custom characters (e.g. halftone dither patterns, box drawing variants) can be uploaded to any character code by writing 8 bytes to the appropriate font addresses. The upload must happen *after* enabling 80-column mode (bit 6 of \$D01D), as the font write registers are inactive when the FPGA is in standard GTIA mode.

## 5 ANTIC Note

The 80-column modes are, from ANTIC’s perspective, graphics mode F. In modes 3 and 4 the screen data for a single frame exceeds 4KB, so the standard ANTIC 4KB address-wrap

limitation applies — just as it would for a native 320×192 bitmap. Use LMS instructions to reset the address counter as needed, the same way you would in any high-resolution ANTIC graphics mode.

## 6 Quick Reference

### 6.1 Mode Summary

Mode	Colour model	PMCTL	MF	Blank	B/row	DL pattern (per row)
1	None (fixed)	\$40	2	6 (\$50)	80	\$0F×2, \$50
2	CGA 16-colour	\$60	4	4 (\$30)	160	\$0F×4, \$30
3	Atari chroma+luma	\$48	6	2 (\$10)	240	\$0F×6, \$10
4	RGB 4096-colour	\$68	8	0	320	\$0F×8

*LMS bits and addresses are added to the \$0F instructions as needed by the application (see Section 5).*

### 6.2 Font Address Calculation

For character code  $c$  (0–255), pixel row  $r$  (0–7):

$$\begin{aligned} \text{FONT\_ADDRH} &= c \div 32 \quad (\text{equivalently: } c \gg 5) \\ \text{FONT\_ADDRL} &= (c \bmod 32) \times 8 + r \quad (\text{equivalently: } (c \& \$1F) \ll 3 + r) \end{aligned}$$

Or more simply, the linear byte address is  $c \times 8 + r$ , split as:

$$\begin{aligned} \text{FONT\_ADDRH} &= (c \times 8 + r) \gg 8 \\ \text{FONT\_ADDRL} &= (c \times 8 + r) \& \$FF \end{aligned}$$

## A Program Listings

### A.1 Font Upload — ATASCII Order (MADS)

The following routine uploads the full Atari ROM character set (128 normal + 128 inverse = 256 characters) to the FPGA font memory, remapped to ATASCII order. It must be called *after* enabling 80-column mode (bit 6 of \$D01D).

ZPTR is any available 2-byte zero-page location (e.g. \$CB/\$CC).

```

FONT_ADDRH = $D000
FONT_ADDRL = $D001
FONT_DATA  = $D002
FONT_WR    = $D003

LOAD_FONT:
    ldx #0
LF_BLOCK:
    lda LF_SRC_HI,x
    sta ZPTR+1          ; source page high byte
    lda #0
    sta ZPTR            ; source page low byte = 0
    txa
    sta FONT_ADDRH     ; destination page = block index
    lda LF_EOR,x
    sta LF_EOR_OP+1    ; self-modify: $00 or $FF
    ldy #0
LF_BYTE:
    sty FONT_ADDRL
    lda (ZPTR),y
LF_EOR_OP:
    eor #0              ; $00 = normal, $FF = inverse
    sta FONT_DATA
    lda #$FF
    sta FONT_WR
    lda #0
    sta FONT_WR
    iny
    bne LF_BYTE
    inx
    cpx #8
    bne LF_BLOCK
    rts

; Source pages (ROM $E000) remapped to ATASCII destination order.
; Pages 0-3: normal characters. Pages 4-7: inverse (EOR $FF).
LF_SRC_HI: .byte $E2,$E0,$E1,$E3,$E2,$E0,$E1,$E3
LF_EOR:    .byte 0, 0, 0, 0,$FF,$FF,$FF,$FF

```