

Full Double Buffer Plan & Impact Analysis

Date: 2026-04-30 Project: HBSTK Status: Planning

1. Current: Two Buffers, Four Screen Ops

Buffer	Owner	Purpose
s_hdcPaintBuf	gtwvt.c PaintText	ASCII/CJK text rendering
s_hdcDb1Buf	usrfont_scan.c ScanAndDrawUda	UDA/ASCFONT/PUA overlay

- * 1. BitBlt(s_hdcPaintBuf <- screen) - PaintText capture
- * 2. BitBlt(screen <- s_hdcPaintBuf) - PaintText write ASCII
- * 3. BitBlt(s_hdcDb1Buf <- screen) - ScanAndDrawUda capture
- * 4. BitBlt(screen <- s_hdcDb1Buf) - ScanAndDrawUda write UDA

Visible intermediate state between #2 and #4 causes flicker.

2. Target: One Shared Buffer, One Screen Write

- * g_hSharedBuf: global shared off-screen memory DC
- * g_hPaintDC: signal to PaintText (non-NULL = use shared buffer)
- * PaintText -> g_hSharedBuf (draw ASCII, no screen write)
- * ScanAndDrawUda -> g_hSharedBuf (draw UDA, no capture/writeback)
- * WndProc: ONE BitBlt(screen <- g_hSharedBuf) at the end

3. Implementation Steps (Atomic)

Step 1: Global Variables

File: usrfont_c.c | Risk: LOW

Add g_hSharedBuf, g_hSharedBmp, g_nSharedW/H, g_hPaintDC globals. Pure addition, no existing code affected.

Step 2: WndProc Lifecycle

File: usrfont_wndproc.c | Risk: HIGH

EnsureSharedBuf + set g_hPaintDC + HideCaret before CallWindowProcW + final BitBlt after ScanAndDrawUda + caret rebuild + clear g_hPaintDC. Most complex step.

Step 3: PaintText Redirect

File: gtwvt.c | Risk: MEDIUM

if g_hPaintDC: capture to shared buf, hdc=g_hPaintDC, skip final BitBlt. if NULL: original path.

Step 4: ScanAndDrawUda Redirect

File: usrfont_scan.c | Risk: MEDIUM

if g_hPaintDC: hdcDraw=g_hPaintDC, skip
GetDC/capture/writeback/ReleaseDC. if NULL: original path.

Step 5: Caret to WndProc

File: wndproc+scan | Risk: MEDIUM

ScanAndDrawUda computes caret params only. WndProc does HideCaret -> final BitBlt -> CreateCaret -> ShowCaret.

Step 6: Resize/Rebuild

File: usrfont_wndproc.c | Risk: MEDIUM

Compare GetClientRect vs stored W/H. Rebuild on mismatch. First-time init from screen capture.

4. Risk Analysis

Step	Risk	Key Concern
1. Globals	LOW	Pure addition
2. WndProc	HIGH	Capture timing, re-entrancy, GDI lifecycle, must ensure final BitBlt always executes
3. PaintText	MEDIUM	Skip final BitBlt correctly, font SelectObject on shared DC, margin fill
4. ScanAndDrawUda	MEDIUM	CreateCompatibleDC from memory DC, skip capture/writeback
5. Caret	MEDIUM	HideCaret counter, timing between EndPaint and final BitBlt
6. Resize	MEDIUM	Same patterns as existing code

5. Previous "Flash Then Disappear" Root Causes

#	Root Cause	Solution
1	WndProc calls BeginPaint -> PaintText calls again -> empty dirty rect	WndProc uses GetUpdateRect only, never BeginPaint
2	PaintText writes screen -> WndProc overwrites with stale buffer	PaintText skips final BitBlt when g_hPaintDC set
3	Capture at wrong timing (after BeginPaint, before TextOut)	Capture stays inside PaintText (correct timing)
4	ScanAndDrawUda early return (no UDA) -> final BitBlt skipped	WndProc ALWAYS does final BitBlt
5	EndPaint re-shows caret -> captured into buffer as ghost	WndProc HideCaret before CallWindowProcW

6. Safety Net (g_hPaintDC == NULL Fallback)

- * s_pFontData == NULL (startup) -> hook disabled, original path
- * g_hPaintDC == NULL (outside WM_PAINT) -> PaintText/ScanAndDrawUda original path
- * Optional runtime toggle (g_bUseSharedBuf) for A/B testing
- * All original buffers (s_hdcPaintBuf, s_hdcDblBuf) preserved, not deleted

7. Edge Cases

Scenario	Risk	Notes
Program startup (no font)	None	s_pFontData==NULL, hook disabled
Minimize/restore	Low	GetClientRect 0x0 -> early return; restore rebuilds buffer
Alt+Tab	Low	focus events reset caret; WM_PAINT repaints
Maximize/restore	Low	WM_SIZE -> font recalc -> WM_PAINT -> buffer rebuild
Window overlap	Low	Windows invalidates exposed area only
PrintScreen	Improved	Single BitBlt reduces partial-state capture chance
MessageBox dialog	Low	Modal loop, repaint on dismiss
DispBegin/DispEnd	None	Controls InvalidateRect timing, not WM_PAINT internals
WM_PAINT re-entry	Medium	s_bInsidePaint prevents, re-entrant PaintText uses same g_hPaintDC

8. All Static DC/Bitmap Variables

Variable	File	Purpose
s_hdcPaintBuf / s_hbmPaintBuf	gtwvt.c	PaintText ASCII buffer (kept, unused when g_hPaintDC active)
s_hdcDb1Buf / s_hBmpDb1Buf	usrfont_scan.c	ScanAndDrawUda full-screen buffer (kept, unused when g_hPaintDC active)
s_hdcMemScan / s_hBmpScan	usrfont_scan.c	Per-glyph bitmap workspace (24x24 DIBSection, still used)
s_hdcAlpha / s_hbmAlpha	usrfont_cache.c	Alpha blending DIBSection (still used, draws to hdcDraw param)
s_hdcDynBuf / s_hbmDynBuf	usrfont_render.c	96x96 supersampling buffer (still used, StretchBlt to hdcDraw)
g_hSharedBuf / g_hSharedBmp	usrfont_c.c (NEW)	Shared off-screen buffer (full screen, sole source for screen write)

9. Modified Files

File	Steps	Changes
usrfont_c.c	1	New globals: g_hSharedBuf, g_hSharedBmp, g_nSharedW/H, g_hPaintDC
usrfont_wndproc.c	2, 5	WM_PAINT: buffer mgmt + g_hPaintDC + final BitBlt + caret
gtwvt.c	3	PaintText: if g_hPaintDC -> capture to buffer, skip final BitBlt
usrfont_scan.c	4, 5	ScanAndDrawUda: if g_hPaintDC -> direct draw, skip capture/writeback/caret
usrfont_int.h	1	extern declarations (if needed cross-file)