STK 庫存系統
游標殘影 & 游標消失
修復技術報告

GTWVT 雙緩衝渲染引擎 x Windows 系統游標整合

日期：2026-04-19 ~ 2026-04-20

檔案：E:\ConvertFW\HBSTK\compat\usrfont.prg

技術摘要

問題：ScanAndDrawUda 雙緩衝令 GTWVT XOR 游標被截入 buffer，形成殘影；
同時游標高度、全形/半形管理、SC_NONE 處理均出現各種失誤。

解法：HideCaret -> 截圖 -> 畫 UDA -> BitBlt -> 自建 CreateCaret + ShowCaret。

成果：殘影消除、半形/全形游標正確、INSERT 高度可設定、SC_NONE 不破壞游標。

1. 背景與技術環境

1.1 系統架構

本系統為 Clipper 5.2 庫存管理系統，目標轉換為 Harbour 32-bit GTWVT Console 模式。
CJK 字型透過 usrfont.prg 的 C 擴充以 GDI 覆蓋渲染（UDA: User Defined Attribute）。

元件	說明
Harbour 32-bit	Clipper 相容語言，GT 驅動 = GTWVT
GTWVT	WM_PAINT 呼叫 CallWindowProcW 畫 ASCII；管理系統游標（XOR caret）
ScanAndDrawUda	WM_PAINT 後半段：掃描 UDA/PUA 格，用 GDI 疊畫 CJK 字形
usrfont.prg	#pragma BEGINDUMP C 代碼，含雙緩衝、游標邏輯
BCC32 C89	編譯器：所有變數必須在 block 頂端宣告，未用變數=編譯錯誤
stk.ini	設定檔：含 CARET_NORMAL_H / CARET_INSERT_H

1.2 雙緩衝設計（2026-04-19 前）

雙緩衝目的：消除 UDA 字形一格一格出現的閃爍。

- HideCaret 尚未加入 -> GTWVT XOR caret 仍在螢幕上
- BitBlt(screen -> buffer) 把 XOR caret 一起截入 buffer
- BitBlt(buffer -> screen) 把 caret 位置「烙印」回螢幕
- GTWVT 再 XOR 一次 -> net effect = 殘影（白色矩形）

1.3 問題清單（工作開始時）

問題	現象	原因
殘影	游標移動後留白色矩形	XOR caret 被截入 double-buffer
游標消失	WINFORM 輸入時無游標	SC_NONE 狀態 DestroyCaret 破壞 GTWVT caret
全形游標不見	中文格位置游標缺失	GTWVT 固定 1xCellW，未依全形建 2xCellW
INSERT 高度錯誤	INSERT 游標跑到視窗頂端	計算公式未分 SC_INSERT/SC_SPECIAL1

2. 游標殘影（Ghost Artifact）

2.1 根本原因

GTWVT 使用 Windows XOR caret（CreateCaret/SetCaretPos/ShowCaret）。
XOR caret 的特性：繪製時把螢幕像素 XOR 反相，故 caret 長在螢幕 DC 上。
雙緩衝截圖時 caret 已在螢幕 -> 被一起截入 buffer -> BitBlt 回螢幕時烙印。

```
1. CallWindowProcW  -> GTWVT 畫 ASCII（caret 已 XOR 在螢幕）
2. BitBlt(hdc -> s_hdcDblBuf)   <- caret 被截入 buffer！
3. DrawXxx(s_hdcDblBuf, ...)    -> 畫 UDA 字形到 buffer
4. BitBlt(s_hdcDblBuf -> hdc)  -> 含 caret 殘影的 buffer 貼回螢幕
5. GTWVT blink timer XOR 一次  -> 游標閃爍，舊位置殘留白塊
```

2.2 最終解法：HideCaret 夾住截圖

```
/* 截圖前隱藏系統游標，防止 XOR caret 被截入 buffer */
HideCaret( hWnd );

/* 截取乾淨螢幕到 off-screen buffer */
if( s_hdcDblBuf )
  BitBlt( s_hdcDblBuf, 0,0, rc.right, rc.bottom, hdc, 0,0, SRCCOPY );

/* 畫所有 UDA 字形到 buffer */
hdcDraw = s_hdcDblBuf ? s_hdcDblBuf : hdc;
... /* DrawXxx 迴圈 */

/* 一次 BitBlt：buffer -> 螢幕（無殘影）*/
if( s_hdcDblBuf )
  BitBlt( hdc, 0,0, rc.right, rc.bottom, s_hdcDblBuf, 0,0, SRCCOPY );

/* 重建正確尺寸游標並 ShowCaret */
if( s_nCaretStyle > 0 ) {
  DestroyCaret();
  CreateCaret( hWnd, NULL, s_nCaretWidth * nCellW, _ch );
  SetCaretPos( s_nCaretCol * nCellW, (s_nCaretRow+1)*nCellH - _ch );
  ShowCaret( hWnd );
} else {
  ShowCaret( hWnd );  /* SC_NONE: 讓 GTWVT caret 自然顯示 */
}
```

[OK] 殘影完全消除，游標正常顯示。

3. 游標消失問題（5 輪迭代）

3.1 第一輪：WM_MY_UPDATE_CARET 抑制方案（失敗）

初期想法：在 WM_MY_UPDATE_CARET（WM_USER+0x0101）返回 0，改由我們自己建立 caret。

[坑] 抑制 WM_MY_UPDATE_CARET -> GTWVT 從不建立 caret -> blink timer 從不啟動 -> WM_TIMER 從不觸發 -> 游標永遠不見。

解法：完全移除 WM_MY_UPDATE_CARET 抑制，讓 GTWVT 正常管理 blink timer。

3.2 第二輪：else 分支 DestroyCaret（失敗）

原意：if(s_nCaretStyle > 0) ... else { DestroyCaret(); } 在 SC_NONE 時清掉游標。

[坑] WINFORM 進入時 SetCursor(SC_NONE)，ScanAndDrawUda 每次 WM_PAINT 都進 else -> DestroyCaret() 把 GTWVT 的 caret 一直刪掉 -> 游標永久消失。

解法：else 改為 ShowCaret(hWnd)——不主動刪游標，讓 GTWVT 的維持存在。

3.3 第三輪：BCC32 W8004 未用變數（編譯錯誤）

移除 GDI 游標繪製區塊後，switch(curStyle) 計算 curH 但後面已無用途。

BCC32 W8004「assigned a value that is never used」在 -w 模式下視為 error。

[坑] BCC32 C89：未用變數 = 編譯錯誤，必須同時移除 int 宣告和 switch 計算。

解法：刪除 int curH; 宣告與整個 switch 區塊，改在 ShowCaret 段計算 _ch。

3.4 第四輪：診斷 FillRect 永不執行

為確認游標位置，加入綠色 FillRect 診斷，但寫在 curStyle != SC_NONE 條件內。

WINFORM 狀態 curStyle=0（SC_NONE）-> 永遠不執行 -> 誤以為位置計算有誤。

[坑] 診斷代碼必須無條件執行，否則可能因條件永不成立而收不到任何輸出。

3.5 第五輪：Python heredoc 的 \\n 問題

用 Python 寫入 C fprintf 格式字串時，\\n 在 heredoc 內被解為換行字元

-> C 原始碼含未終止字串 -> BCC32 編譯失敗「Unterminated string constant」。

```
# 錯誤：\n 被解為換行
c_code = b'fprintf(f, "val=%d\n", x);'

# 正確：改用 fputc(10, f) 避免 \n 問題
c_code = b'fprintf(f, "val=%d"); fputc(10, f);'
```

4. 全形 / 半形游標管理

4.1 問題

GTWVT 一律建立 1xnCellW 寬度的游標。中文全形字占 2 格，游標應為 2xnCellW。
修改前：全形位置游標只有半格寬，看起來像消失。

4.2 寬度判斷邏輯

```
/* 計算游標所需寬度 ( 格數 ) */
int neededW = bUdaPua ? 2 : ( bFullWidth ? 2 : 1 );
s_nCaretWidth = neededW;   /* 1=半形, 2=全形 */
s_nCaretRow   = curRow;
s_nCaretCol   = curCol;
s_nCaretStyle = curStyle;
```

情況	CreateCaret 寬度	高度 _ch
半形英數	1 x nCellW	由 curStyle/stk.ini 決定
全形中文 / UDA	2 x nCellW	同上
SC_NONE (隱藏)	不呼叫 CreateCaret	直接 ShowCaret(hWnd)

4.3 游標高度與 stk.ini 整合

```
/* stk.ini */
CARET_NORMAL_H = 6   ; SC_NORMAL：底部 6px 橫條
CARET_INSERT_H = 12  ; SC_INSERT / SC_SPECIAL1：12px 半高塊

/* 高度計算 */
switch( s_nCaretStyle ) {
 case 2:  /* SC_INSERT   */
 case 3:  /* SC_SPECIAL1 (AWICursor INSERT mode) */
   _ch = (s_nCursorInsertH > 0) ? s_nCursorInsertH : (nCellH+1)/2;
   break;
 case 4:  /* SC_SPECIAL2 */
   _ch = nCellH * 3 / 4;
   break;
 default: /* SC_NORMAL */
   _ch = (s_nCursorNormalH > 0) ? s_nCursorNormalH : 2;
   break;
}
```

5. AWICursor 與 WINFORM 游標狀態機

5.1 游標常數

常數	值	說明
SC_NONE	0	無游標（隱藏）
SC_NORMAL	1	底部細橫條（NORMAL MODE）
SC_INSERT	2	半高塊（INSERT MODE）
SC_SPECIAL1	3	全高塊（AWICursor INSERT 模式用）
SC_SPECIAL2	4	3/4 高塊

5.2 WINFORM 游標生命週期

```
WINFORM.PRG:
 L19:  LOCAL nLastCursor := SetCursor( SC_NONE ) // 進入先隱藏
 L670: nLastCursor := AWICursor()              // 顯示正確游標
 L678: ReadModal(LocalGetList)                 // 輸入迴圈
 L682: SetCursor( nLastCursor )               // 離開後還原 SC_NONE

AWICursor() (TLIB.PRG L398):
 RETURN SetCursor( IIF( READINSERT(), SC_SPECIAL1, MEMVAR->nStkCursor ) )
 // INSERT 模式 -> SC_SPECIAL1(3)，NORMAL 模式 -> nStkCursor(=1)

GETSYS.PRG:
 L125: AWICursor()  // oGet:SetFocus() 後確保游標可見
 L454: nLastCursor := AWICursor()  // GETINKEYBLOCK eval 路徑
```

5.3 ScanAndDrawUda 游標流程（簡化）

```
WM_PAINT
 |
 +- CallWindowProcW   -> GTWVT 畫 ASCII + 管理 XOR caret
 |
 +- ScanAndDrawUda
    |
    +- HideCaret(hWnd)          <- 防止 caret 被截入 buffer
    +- BitBlt(screen -> buffer)   <- 截取乾淨畫面
    +- hdcDraw = buffer
    |
    +- [掃描每個格]
    |   +- 若游標格: 記錄 s_nCaretRow/Col/Width/Style
    |   +- DrawXxx(hdcDraw, ...)  -> 畫 UDA/PUA/ASCFONT
    |
    +- BitBlt(buffer -> screen)   <- 一次更新全畫面
    |
    +- 游標重建
       +- s_nCaretStyle > 0:
```

```
    |   DestroyCaret()
    |   CreateCaret(hWnd, NULL, s_nCaretWidth*nCellW, _ch)
    |   SetCaretPos(col*cellW, (row+1)*cellH - _ch)
    |   ShowCaret(hWnd)
    +- s_nCaretStyle <= 0 (SC_NONE):
        ShowCaret(hWnd)     <- 讓 GTWVT caret 自然顯示
```

6. 踩坑總表

坑	現象	根本原因	解法
WM_MY_UPDATE_CARET 抑制	游標完全消失 blink 不動	GTWVT blink timer 從不啟動	移除抑制 讓 GTWVT 正常建立 caret
else 分支 DestroyCaret	WINFORM 輸入 無游標	SC_NONE 時每次 WM_PAINT 都刪 caret	else 改 ShowCaret 不主動刪游標
診斷 FillRect 寫在條件內	診斷無效 誤判位置	curStyle!=SC_NONE 但 WINFORM 永遠 SC_NONE	診斷代碼移到 條件外無條件執行
BCC32 W8004 int curH 未用	編譯失敗	移除 GDI 游標區塊後 curH switch 無用途	同時刪 int curH 和整個 switch 計算
Python \\n 在 C 字串	BCC32 Unterminated string constant	heredoc 內 \\n 被解為換行	改用 fputc(10, f) 避免 \n
GTWVT XOR caret 截入 buffer	游標移動後 留白色殘影	BitBlt 前 caret 已 XOR 在螢幕	BitBlt 前先 HideCaret(hWnd)
全形格游標 寬 1xCellW	中文格游標 看起來消失	GTWVT 固定 1xCellW	偵測 bFullWidth/bUdaPua CreateCaret 2xCellW
INSERT 游標 跑到頂端	INSERT 模式游標 位置不正確	SetCaretPos Y 偏移 未減 _ch	Y = (row+1)*cellH - _ch 讓游標貼底部

7. 最終架構與靜態變數

7.1 新增靜態變數

變數	型別	用途
s_hdcDblBuf	HDC	off-screen buffer DC
s_hBmpDblBuf	HBITMAP	off-screen bitmap
s_nDblBufW/H	int	buffer 目前尺寸（尺寸變化時重建）
s_nCaretRow	int	游標所在列（-1=無）
s_nCaretCol	int	游標所在欄
s_nCaretWidth	int	游標寬度格數（1=半形 / 2=全形）
s_nCaretStyle	int	游標樣式（0=SC_NONE, 1=SC_NORMAL, 2=SC_INSERT…）
s_nCursorNormalH	int	SC_NORMAL 高度像素（stk.ini CARET_NORMAL_H）
s_nCursorInsertH	int	SC_INSERT 高度像素（stk.ini CARET_INSERT_H）

7.2 移除的機制

- WM_MY_UPDATE_CARET 攔截（return 0）— 破壞 blink timer
- s_bCursorDrawn / s_nCaretH 靜態變數 — 已被新架構取代
- WM_TIMER 內呼叫 ScanAndDrawUda — 避免與 WM_PAINT 競爭
- ScanAndDrawUda 內 GDI FillRect 診斷塊 — 驗證完成後清除

7.3 保留的機制

- GTWVT 原有 XOR caret 建立與 blink timer — 我們只 Hide/Show，不主動 Destroy
- 雙緩衝 s_hdcDblBuf — 消除 UDA 字形逐格閃爍
- HideCaret -> 截圖 -> 畫 UDA -> BitBlt -> CreateCaret+ShowCaret — 核心流程

8. 工作時序與關鍵教訓

8.1 工作時序

時間	工作項目	結果
04-19	雙緩衝初版（fix_dblbuf.py）	UDA 閃爍消除，但出現殘影
04-20 上午	嘗試 WM_MY_UPDATE_CARET 抑制	游標完全消失，失敗
04-20 上午	加 cursor_debug.log 診斷	確認 curStyle=0 常態，blink 未啟動
04-20 下午1	改 HideCaret 方案，移除抑制	殘影消除，但 else DestroyCaret 破壞游標
04-20 下午2	修 else -> ShowCaret	半形游標正常，全形仍不見
04-20 下午3	加 s_nCaretWidth 全形判斷	全形游標出現，INSERT 高度錯誤
04-20 下午4	整合 stk.ini CARET_NORMAL/INSERT_H	INSERT 游標高度正確
04-20 下午5	清除診斷代碼，移除 curH 未用變數	編譯通過，功能全部正常
04-20 晚	確認「全部正常」，清除多餘備份	完成

8.2 關鍵教訓

- Windows HideCaret/ShowCaret 是配對 API，必須在同一 WM_PAINT 週期內平衡。
- GTWVT 的 XOR caret 是普通系統 caret，HideCaret 可暫時壓制而不破壞。
- DestroyCaret 是破壞性操作，只有確定要重建不同尺寸時才用。
- BCC32 C89：所有變數必須在 block 頂端宣告，未用變數是 error 而非 warning。
- SC_NONE 在 WINFORM 期間幾乎是常態，判斷條件必須考慮「游標隱藏」這個正常狀態。
- 診斷 log 要無條件觸發，若加在條件內可能因條件永不成立而收不到任何輸出。
- Python 修改 BIG5 PRG 時，C 字串內的 \n 必須用 fputc(10) 替代，否則 heredoc 斷行。
- GTWVT blink timer 與 caret 建立是一體的，抑制 WM_MY_UPDATE_CARET = 殺掉 blink。

8.3 可移植的設計模式

本次建立的「HideCaret -> 截圖 -> 畫疊加 -> BitBlt -> CreateCaret+ShowCaret」模式，
適用於任何在 WM_PAINT 中需要疊加自訂字形同時保有系統游標的情境。
重點：系統 caret 必須在截圖前隱藏，在貼圖後重建，並讓 GTWVT 的 blink timer 繼續運行。

9. 相關檔案一覽

檔案	說明
compat\usrfont.prg	主程式：雙緩衝 + ASCFONT + 游標管理（#pragma BEGINDUMP C 代碼）
source\WINFORM.PRG	輸入表單：SetCursor(SC_NONE)/AWICursor()/SetCursor(SC_NONE) 生命週期
source\GETSYS.PRG	GET 系統：AWICursor() 呼叫點（L125, L454）
source\TLIB.PRG	AWICursor() 定義（L398）：READINSERT() -> SC_SPECIAL1 or nStkCursor
source\STK.PRG	PUBLIC nStkCursor 宣告（L27）+ stk.ini StkCursor 讀取（L193-195）
build\stk.ini	CARET_NORMAL_H / CARET_INSERT_H 設定
build\fix_dblbuf.py	雙緩衝修改腳本（歷史，已套用）
build\fix_winform_paintlock.py	WINFORM paint-lock 修改腳本（歷史，已套用）
build\log\cursor_debug.log	診斷用游標 log（臨時診斷用）
usrfont.prg.bak_dblbuf_ok_20260420_1349	功能確認後的正常版基準備份

10. 結語

本次修復歷經多輪迭代，核心困難在於：GTWVT 的游標機制（XOR caret + blink timer）
是一個整體，不能半抑制；雙緩衝的截圖動作必須在游標隱藏後進行；
全形/半形的動態判斷必須與 GTWVT 的 bFullWidth 旗標同步。

最終方案以最少干預 GTWVT 內部機制為原則：
僅在截圖前加 HideCaret、在貼圖後加 CreateCaret+ShowCaret，
即達成「無殘影、無消失、全形/半形正確、INSERT 高度可設定」的完整目標。

本報告由 Claude Code (claude-sonnet-4-6) 自動產生
2026-04-20 E:\ConvertFW\HBSTK\doc\