

# Controllo di display TFT con Actel FPGA

*Il controllo dei display a colori di tipo TFT, dal 320x240 al 640x480, è un argomento molto sentito grazie ai prezzi sempre più competitivi per questo tipo di dispositivi*

**N**ei sistemi ove esista un'interfaccia utente operatore, da sempre è presente l'esigenza di fornire una migliore soluzione grafica, ovviamente vincolata al costo massimo sopportabile dal sistema finale. Negli ultimi due anni i display a colori di tipo TFT sono talmente diminuiti di costo, da permettere ormai l'uso di LCD da 320x240 pixel (è il caso dei display da 3,5 pollici) fino ai 640x480 (5,6 pollici e 7 pollici). Il progetto che presentiamo è stato sviluppato per una applicazione 640x480, ma nell'articolo verranno evidenziati i punti di incontro e le problematiche rispetto allo sviluppo di applicazioni anche per LCD da 320x240 pixel da 16 bit colori e superiori. L'applicazione è stata provata in laboratorio utilizzando la scheda Colibrì, prodotta dalla Acme Systems di Roma. Questa piccola scheda di sviluppo, essendo fornita di banchi RAM esterni e I/O, ha permesso il test dell'applicazione video, collegando direttamente l'LCD ai suoi connettori di uscita. La scheda Colibrì mon-

ta una FPGA Actel da 250 Kgate, che per la specifica prova è risultata occupata soltanto al 9%. Per applicazioni di produzione, un chip da 60 Kgate è più che sufficiente per supportare il controllo del video nelle sue funzioni base. Grazie al fatto che dispone di PLL e di Dual Port Ram, oltre a un prezzo basso, da poter essere considerato da chi deve fare i conti con il costo finale del sistema, tale componente ben si adatta a questa applicazione. L'intero progetto e la relativa documentazione è disponibile su richiesta ai riferimenti riportati a fine articolo, mentre l'ambiente di sviluppo completo Libero 8.6 è disponibile gratuitamente sul sito del produttore [www.actel.com](http://www.actel.com).

## La generazione dei segnali di controllo video

La **figura 1** riporta lo schema a blocchi del controller di base. In generale si desidera gestire display a colori di tipo TFT (*Thin Film Transistor*) grazie alla loro qualità video, decisamente migliore rispetto alle

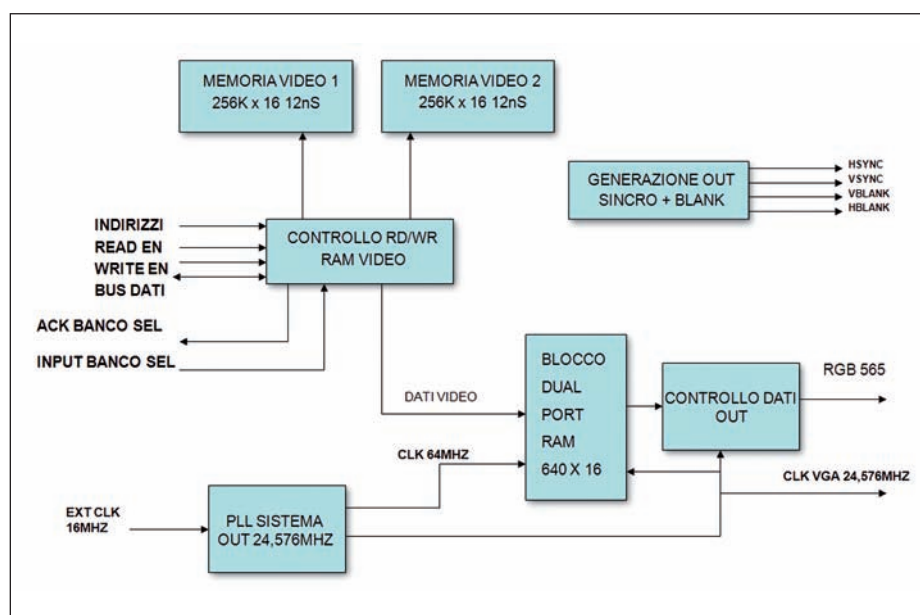


Figura 1: schema a blocchi del controller di base.

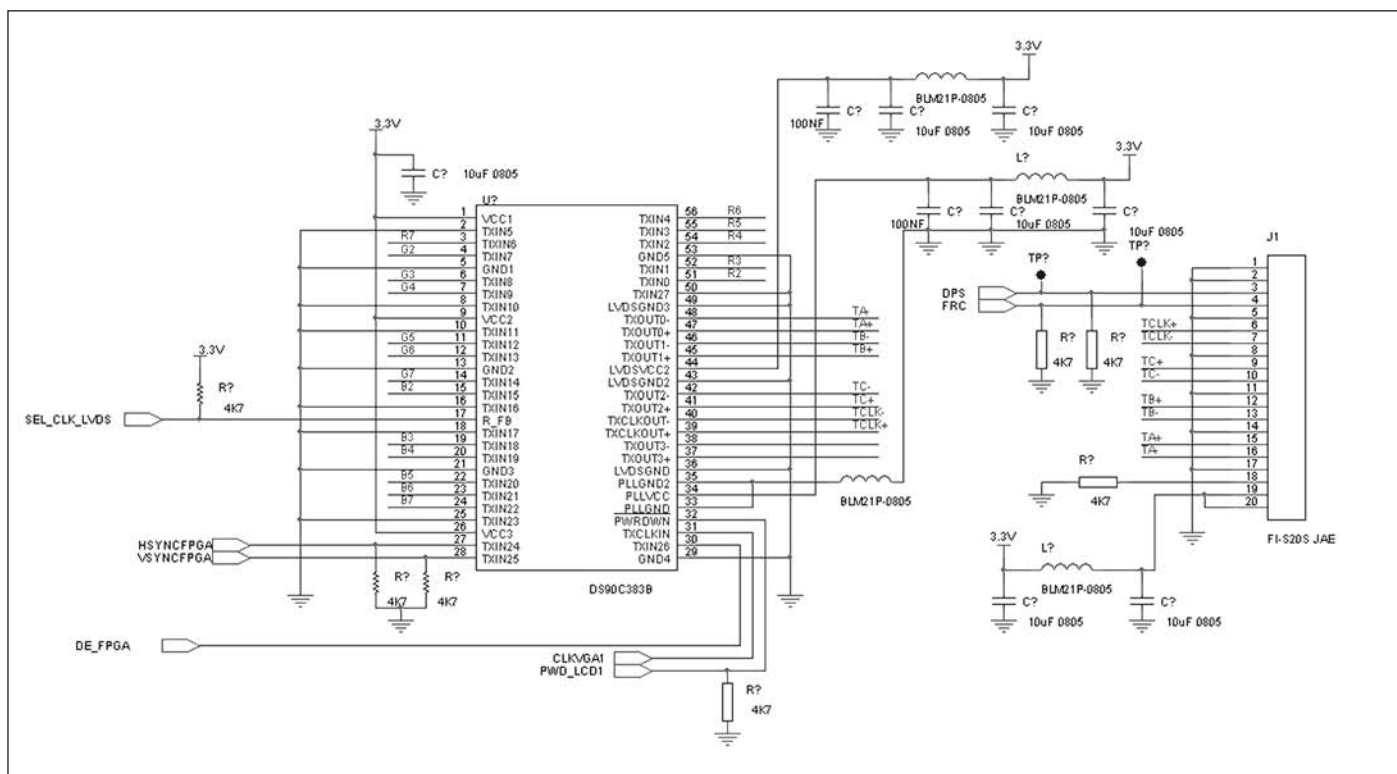


Figura 2: schema di collegamento per il passaggio da digitale a LVDS.

precedenti tecnologie utilizzate. Per il controllo del video come sezione base si deve partire dalla scansione orizzontale e verticale. Questi due contatori devono generare il segnale di *blanking* sia verticale che orizzontale. Per chi non avesse dimestichezza, si indica che nel caso degli LCD dobbiamo considerare i segnali di *blanking* come dei reset per i contatori di pixel e di righe interne all'LCD. La loro durata (in genere attivi alti, ma dipende dall'LCD scelto) è ben specificata nei datasheet dei costruttori. Per esempio, guardando il datasheet della Newtec per l'LCD NFD057C, LCD di tipo TFT da 5,7 pollici 640x480 pixel, vediamo che il *blanking* orizzontale deve durare tipicamente 160 colpi di clock (il clock tipico per questi LCD è 24.576 MHz) ovvero circa 6,5  $\mu$ s. Mentre il *blanking* verticale viene espresso in numero di righe orizzontali, per cui sapendo che la riga orizzontale dura tipicamente ottocento colpi di clock, calcoliamo la sua durata in 32,55  $\mu$ s e infine, verificando che il *blanking* verticale richiesto è tipicamente di 45 righe, possiamo arrivare al valore finale di 32,55  $\mu$ s x 45 = 1,464 mS. Ora possiamo facilmente verificare come partendo in una FPGA da un paio di semplici contatori

che hanno come base di clock l'oscillatore utilizzato per pilotare lo stesso LCD (quindi il 24,576 MHz nel caso del TFT da 5,7 pollici), sia decisamente semplice generare i segnali di *blanking*. Nel **listato 1** è visualizzato un semplice segmento di codice che esplica la funzione di gestione del *blanking* orizzontale. Nell'esempio, il segnale di *blanking* viene posto attivo alto 32 pixel, dopo la fine della sezione attiva (H\_ACTIVE), che nel caso del VGA è di 640 pixel, e viene disabilitato giusto 32 pixel dopo l'inizio della riga totale. Il contatore generale di pixel riga effettua un conteggio da zero a H\_TOTAL-1, che per la VGA è di 800 pixel. Stesso meccanismo prevede la gestione del *blanking* verticale, dove però useremo come unità di misura non più il numero di pixel, ma le righe stesse (**listato 2**). Pertanto il contatore generale di righe, arrivando alla soglia stabilita di 22 righe, dopo la fine dell'area attiva V\_ACTIVE (480 righe per il VGA) abilita il *blanking* verticale. Lo stesso viene disabilitato alla riga 22 della scansione. Con il medesimo ragionamento e meccanismo è possibile, in sede di codice VHDL, generare i segnali di sincronismo riga e quadro. Considerazione particolare per gli LCD dell'ultima generazione, ri-

chiede il segnale DE come Data Enable. Questo segnale è utilizzato dagli LCD esattamente come abilitazione alla ricezione dei dati, sincrono con il clock video. Pertanto studiando le timing dell'LCD, potremo verificare che in corrispondenza della sua attivazione, l'LCD attende il primo pixel della riga, esattamente il colpo di clock successivo. Lo stesso DE verrà mantenuto disattivo, non solo per il tempo di *blanking* orizzontale, ma anche per il tempo di *blanking* verticale. In questo modo l'LCD azzererà i suoi circuiti di scansione, per ripartire dal primo pixel della prima riga alla successiva abilitazione di DE. Esso deve essere considerato come un sincronismo composito dell'LCD, dove però i dati devono essere forniti esattamente alla sua attivazione.

### Dimensionamento e tipologia dei dati video

Una prima domanda importate è che tipo di dati video vogliamo gestire e che profondità di colore ci interessa per la nostra applicazione. Un approccio semplice e diretto è definire la nostra immagine esattamente come una bitmap. La dimensione della memoria necessaria sarà direttamente proporzionale a questa: avre-

mo quindi una memoria da 76800 pixel per un QVGA 320x240, oppure 307 K pixel circa per una VGA 640x480. Abbiamo volutamente parlato di pixel, poiché la dimensione della parola sarà direttamente legata alla profondità di colori desiderata: 256 colori (già adatti per un'interfaccia utente di tipo industriale) richiederebbero memoria per 76,8 KB. Utilizzando una memoria standard da 256Kx8 sarebbe possibile avere fino a tre pagine video. Qualora si volessero gestire immagini fino a 16 bit per pixel (65 K colori), per un LCD 320x240 pixel potremmo utilizzare una memoria da 256Kx16 mantenendo le tre pagine video. Più problematico è il discorso per il VGA 640x480, poiché si dovrebbe migrare su una memoria da 512K x 16 SRAM alta velocità (vedremo nella sezione successiva le problematiche legate all'accesso sulla memoria), le quali hanno sul mercato un prezzo elevato. Per mantenere i costi della soluzione convenienti, diventa obbligatorio in questo caso valutare memorie RAM dinamiche sincrone (SDRAM), che forniscono densità elevate a basso costo. Purtroppo nel caso di tale scelta bisognerà integrare sulla FPGA uno SDRAM controller, obbligatorio per questo tipo di memorie. Nel caso di Actel sono disponibili gratuitamente macro già pronte per il controllo di questi dispositivi, senza però dimenticare che tale scelta ci obbligherà all'uso di una FPGA di taglio sufficiente a contenere sia questa macro che tutto il resto del progetto. Un'alternativa alla bitmap diretta in memoria potrebbe essere una gestione indiretta, che utilizzi non il singolo pixel, ma blocchi di pixel, per esempio da 8x8. In questo tipo di controllo verranno caricati in RAM non il valore del singolo pixel, ma il puntatore al blocco di pixel che si desidera portare a video. Pensando l'immagine come un mosaico, caricheremo nella memoria video il numero di

puntatori necessari alla composizione dell'immagine. Per esempio nel caso della QVGA 320x240 avremo 1200 puntatori da caricare e aggiornare per gestire dinamicamente l'immagine. Con il medesimo calcolo verifichiamo che i puntatori per la VGA 640x480 saranno 4800. La dimensione del puntatore in termini di bit ci darà il numero di blocchi o font grafici disponibili per comporre l'immagine. Per esempio con una RAM a 16 bit potremmo gestire indirettamente fino a 65K blocchi diversi, con cui costruire la nostra immagine e utilizzeremo la word caricata nella RAM statica come indirizzo, per esempio, per una memoria Flash da 4Mx16 bit. La FPGA avrà in questo caso il compito di processare in automatico questa fase di lettura e invio dei dati all'LCD per la loro visualizzazione, tenendo conto che in questa condizione non dovrà solo leggere un pixel, ma l'intero blocco di 64 avendo cura di disporli in posizione corretta (otto pixel per otto righe). Appare chiaro che in questo modo l'aggiornamento dell'immagine potrebbe essere possibile anche utilizzando il tempo di *blanking* video, considerando il ristretto numero di scrittura da eseguire rispetto a una gestione a bitmap. Limiti di questa soluzione sono la necessità di creare il font dei caratteri o blocchi da memorizzare nella memoria flash e l'impossibilità di movimenti nell'immagine, che non siano a passi della dimensione orizzontale o verticale del blocco pixel.

### La gestione dei dati video

La gestione dei dati video è la parte più delicata del progetto. Infatti i dati devono essere aggiornati dal microprocessore di sistema e contemporaneamente devono essere mandati a video. Appare quindi chiaro che è necessario gestire e sincronizzare queste due attività. Nel caso degli LCD più semplici come QVGA da 320x240 pixel, la gestione, intesa come scrittura e lettura

dei medesimi, appare più semplice grazie alle frequenze in gioco più rilassate. Infatti guardando le specifiche tecniche di un QVGA da 320x240 3,5 pollici, riscontriamo un clock video di circa 6 MHz. Tenendo conto che in una FPGA standard è quasi sempre presente un PLL, possiamo ipotizzare di elevare la frequenza portandola per esempio a 64 MHz. In questa condizione costruendo una macchina a stati interna potremo, grazie alla maggior velocità, aggiornare la pagina video, utilizzando un'unica memoria e scrivendo i nuovi dati immediatamente dopo la loro lettura da parte del blocco logico, che provvede al loro invio verso l'LCD. Per separare i due clock di sistema, quello interno a 64 MHz e quello dell'LCD a 6 MHz potremo utilizzare un blocco Dual Port RAM disponibile in FPGA. La dimensione di questo blocco di memoria deve essere sufficiente a contenere la riga video (320 pixel o 640 pixel), con una profondità legata al numero di colori desiderato. In questo modo, poiché la scansione di ogni singolo pixel dura circa 160 ns, utilizzeremo questo tempo per gestire la scrittura del nuovo dato pixel in arrivo dal microprocessore esterno. Utilizzando, per esempio, il *blanking* di quadro come interrupt verso il microprocessore, potremo partire nell'aggiornamento della pagina video, qualora fosse richiesto immediatamente alla ricezione di quest'ultimo. Nel caso della VGA 640x480 il tipo di gestione sopra descritto appare più problematico. Infatti, con un clock video di circa 25 MHz e per alcuni modelli anche 33 MHz, la possibilità di gestire lettura e scrittura richiederebbe funzionalità di tipo DMA, da parte del microprocessore di sistema, oltre che un aumento della frequenza interna di lavoro ad almeno un centinaio di MHz. Una soluzione per rilassare il sistema può essere la gestione di un doppio banco di memoria video. Utilizzando due banchi di memoria distinti (oppure sullo stesso chip fisico, ma con due pagine video separate) è possibile ottenere una gestione dei dati, dove non viene posto al microprocessore di sistema l'obbligo di aggiornare completamente la pagina video, prima della successiva scansione di quadro (16,66 ms a 60 Hz di frequenza quadro). Infatti utilizzando un semplice flag hardware, il sistema potrà segnalare alla logica quale dei due banchi è in scansione e quale invece in scrittura. Ter-

## RIFERIMENTI

- ■ ■ A3PN Family Hand Book, <http://www.actel.com/products/pa3nano/docs.aspx>
- ■ ■ Colibri Starter Kit, <http://eshop.acmesystems.it/?id=colibri>
- ■ ■ Libero 8.6 suite, <http://www.actel.com/download/software/libero/default.aspx>
- ■ ■ Richiesta Progetto completo/Libero 8.6 su DVD [marketing@latechnikadue.com](http://marketing@latechnikadue.com)
- ■ ■ I listati sono scaricabili dal sito [www.fwonline.it](http://www.fwonline.it)

minato il caricamento della nuova pagina video, il sistema commuterà tale flag, per segnalare alla FPGA di utilizzare il banco appena scritto. Tale operazione di swap avverrà durante il primo *blanking* di quadro e l'FPGA segnalerà al sistema l'avvenuta commutazione, pilotando a sua volta un flag HW verso il microprocessore, per confermare il banco ora disponibile alla scrittura. Ovviamente, per quello che è stato spiegato nel precedente paragrafo, quanto qui esposto si applica solo a una gestione a bitmap, mentre per le applicazioni a blocchi potrebbe essere possibile una gestione da concludere in un unico tempo di quadro, per esempio, utilizzando un trasferimento dati di tipo DMA per la scrittura dei 4860 puntatori alla memoria flash di sistema.

### La conversione dei dati video

Gli LCD presi in considerazione dispongono tipicamente di ingressi digitali CMOS, pertanto è assolutamente plausibile la connessione diretta delle uscite FPGA a esso. Qualora fossimo in presenza di modelli con ingresso LVDS, sarà necessario l'utilizzo di un convertitore parallelo serie con uscite LVDS. La quasi totalità degli LCD con ingressi LVDS si interfaccia in modalità standard con integrati come il DS90C383B della National, oppure FIN3383B della Fairchild. In **figura 2** lo schema di collegamento per il passaggio da digitale a LVDS. Pur supportando fino a 24 bit di colore, è possibile il collegamento standard a 18 bit, dove tipicamente si collegano a zero i bit meno significativi del rosso e del blu. Per coloro che fossero interessati a una conversione analogica per il pilotaggio di monitor standard VGA con uscita analogica, evidenziamo che per conversioni a 16 bit è sufficiente una rete pesata R2R in uscita direttamente sui pin della FPGA, disaccoppiando i tre colori generati con dei semplici transistor di tipo NPN e portando i segnali di sincronismo verticale e orizzontale dalla FPGA direttamente al connettore VGA. Per applicazioni di maggior qualità è possibile l'uso di convertitori dedicati come ADV7125 di Analog Devices.

### Implementazione pratica

Come accennato nell'introduzione, il test dell'applicazione è stato effettuato utilizzando la scheda Colibrì prodotta dalla Acme Systems (**figura 3**). Questa scheda, svi-

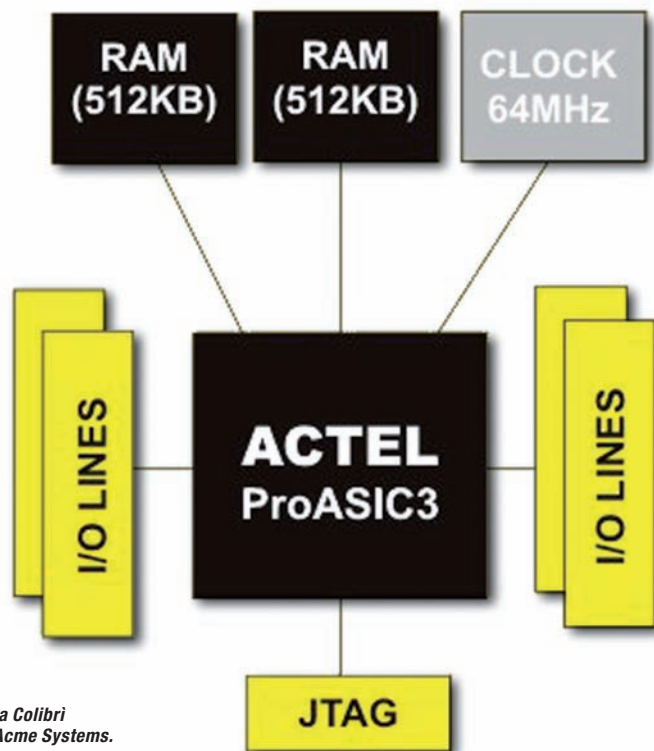


Figura 3: scheda Colibrì prodotta dalla Acme Systems.

luppata per essere utilizzata come generica interfaccia FPGA per un Tiny PC su base Linux sempre prodotto dalla Acme Systems (Fox Board LX832), si prestava molto bene al nostro test poiché già prevedeva due bank RAM statici da 256Kx16 con velocità di accesso di 12 ns. Essendo nata per collegarsi facilmente verso il mondo esterno, dispone di numerosi connettori a passo 2,54 mm; un'ulteriore punto a favore nella scelta di questa board è stato proprio il fatto di essere predisposta per collegarsi alla LX832, la quale disponendo di sistema operativo e hard disk allo stato solido sufficiente a mantenere le bitmap di test ha facilitato la fase di collaudo. L'LCD di prova (NEWTEC NFD057C 640x480 pixel 18 bit colore ingressi CMOS) è stato connesso direttamente a uno dei quattro connettori della Colibrì attraverso un piccolo adattatore per poter utilizzare il connettore FFC con cui l'LCD è predisposto per la connessione su scheda. Poiché la scheda Colibrì dispone di massimo due bank da 256Kx16, si è dovuto fare una scelta di comodo e ridurre l'immagine da 640x480 pixel a 640x400, centrandola e fornendo dalla FPGA il dato pixel 0x0000 per le prime e ultime 40 righe, corrispondente al

colore nero. Per la prova effettuata, avendo la scheda Colibrì un oscillatore quarzato a 64 MHz già montato, si è usato il PLL interno solo per la generazione del clock video a 24,576 MHz.

### Conclusioni

L'analisi finale del progetto ha evidenziato che per le funzionalità di base per il controllo del display la richiesta in termini di gate è decisamente bassa. Soluzioni per QVGA da 320x240 si prestano molto bene a questo tipo di sviluppo, poiché possono essere implementate con costi di materiale assolutamente competitivi rispetto a controller grafici dedicati; non ultimo liberano il progettista dal sempre presente problema dell'obsolescenza, che per questi tipi di dispositivi è molto frequente. Per le applicazioni da VGA 640x480 e superiori, il vantaggio in termini di costo si riduce, dovendo usare una FPGA di densità e I/O maggiori. Resta valido il discorso dell'obsolescenza precoce dei controller video dedicati, che nel caso di soluzioni basate su FPGA non sussiste come problema nel medio periodo. ■