Contents
1 Introduction
2 Overview of the ARM7500 LCD interface
3 Interface design examples:
   A. Monochrome dual STN panels (Example Hitachi LMG5675XUFC)
      A.1 Interface circuitry
      A.2 System functional description
      A.3 Cursor considerations
      A.4 Register values
   B. Colour single TFT panels, 256 colours (Example IBM ITSV34A)
      B.1 Interface circuitry
      B.2 System functional description
      B.3 Cursor considerations
      B.4 Register values
4 Products and services available

1 Introduction
This application note provides detailed information on how to attach a liquid crystal display panel (LCD) to the ARM7500 single chip computer.

ARM is a trademark of Advanced RISC Machines Ltd, ART is a trademark of Acorn Group Plc, IBM is trademark of International Business Machines Corporation, Hitachi is a trademark of Hitachi Europe Ltd.

This application note is intended only to assist the reader in the use of the ARM7500. ARM and Acorn shall not be liable for any loss or damage arising from the use of any information in this application note, or any error or omission in such information. The information contained herein may be changed without prior notice.

References: ARM7500 Data sheet (ARM DDI 0050), relevant display panel data sheet.
2 Overview of the ARM7500 LCD interface

A block diagram of the major elements involved in the data flow for video output from the ARM7500 is shown in figure O.1 below. The CPU creates the cursor images and the screen image in the area of DRAM reserved for use as a video frame buffer. It then programmes the DMA controller registers to carry out the screen image data transfer between DRAM and the video controller, on-demand from the video controller.

Separate DMA requests are made by the video controller for screen image data and cursor data. The DMA controller responds to these requests by holding-off CPU access memory and transferring data to the video controller's FIFO in quad-word blocks.

The diagram shows how the data output through the ED port is switched between a number of sources, as defined and controlled by a combination of ECLK and EREG[1:0]. In addition the data supplied by the 'Green LUT' and the 'External LUT' can be processed by the grey scaler on a frame-by-frame basis in order to generate a grey scale on a monochrome display.
Figure O.2 LCD Block Diagram of ARM7500 Video Controller LCD Components

The diagram shows how the ED port can source its data from either the green LUT, the red and green LUTs or the green and external LUTs with the option to enable the grey scaler.
2 Interface design examples

A. Monochrome dual STN panels (Example Hitachi LMG5675XUFC)

A.1 Interface circuitry

A dual STN panel, as its name implies, is made up of two separate panels one atop the other, requiring synchronous data streams. STN panels use a passive display matrix therefore the frame rate must be maintained at a level which minimises visible flicker. Grey scaling, achieved by pixel modulation, must be generated by the video controller driving the panel. The panel has no inherent means of generating grey scales, a pixel is either on or off.

A panel will typically have the following connections:

<table>
<thead>
<tr>
<th>Panel signal</th>
<th>ARM7500 signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>LD0 ~ LD3</td>
<td>ED0 ~ ED3</td>
</tr>
<tr>
<td>UD0 ~ UD3</td>
<td>ED4 ~ ED7</td>
</tr>
<tr>
<td>CP</td>
<td>ECLK</td>
</tr>
<tr>
<td>FRAME</td>
<td>VSYNC</td>
</tr>
<tr>
<td>LOAD</td>
<td>HSYNC</td>
</tr>
<tr>
<td>DISPLAY_ENABLE</td>
<td>External display enable control</td>
</tr>
</tbody>
</table>

If the system design must allow for the panel to be powered down, for example to save power during idle time, then it will be necessary to intercept some of the signals from the ARM7500 with tristate buffers.
Figure A.1.1 shows a typical interface circuit design.

Contrast voltage may be generated by the back-light inverter circuit. ED signals need not be buffered because they can be set to zero before Vcc is removed from the panel and kept at zero until after Vcc is restored.

**A.2 System functional description**

For the example panel (VGA 640 x 480 with 16 grey levels) the screen image in memory should be a 640 x 480 pixel image with depth of 4 bits per pixel (equating to 16 grey levels including black). The ARM7500 grey-scaler is fixed at 16 grey levels but 1 and 2 bit per pixel modes can be supported by palette mapping.

For 4 bit/pixel modes screen data is fetched from memory in quad-word packets representing 32 pixels, each 4 bit nibble representing the grey level of a single pixel.
Attaching an LCD to ARM7500

It is the function of the grey-scaler to convert the 4 bit depth of each pixel into a 1 bit depth pixel with pulse width modulation. The modulation level is defined by the pixel's 4 bit value.

For dual panels such as the Hitachi LMG5675XUFC the video DMA registers must be set up so that alternate quad-words for upper and lower panel halves are transferred from memory (see ARM7500 data sheet for further details). The data output from the ARM7500 ED port represents 1 bit/pixel data for four pixels of both the upper and lower half panels. Upper panel data is mapped through the bottom 4 bits of the green LUT and lower panel data is mapped through the 4 bit Ext LUT before the grey scaling, allowing palette mapping.

Each byte from the ED port thus represents 4 pixel of the upper panel and 4 pixel of the lower panel (for single panels only the top four bits of the ED port are used). ECLK runs at one-quarter of the pixel rate.

The relationship of ECLK to the pixel data is show in figure A.2.1.

\[ t_{\text{sample}} \] is typically \( \frac{1}{4} \) of the ECLK clock period (approx 62ns at 4MHz). The above schematic shows an RC delay circuit on ECLK which can be used to shift the edge slightly if required.
A.3 Cursor considerations

Cursor data is processed in exactly the same manner as screen data but special measures must be taken when the cursor is required to straddle the join of a dual panel display.

The hardware cursor is 32 pixels wide with a resolution of 2 bits/pixel. Cursor height is not restricted but it will typically be between 32 and 64 rasters long. Cursor data transfers are quad-word transfers and therefore occur once every two raster scans on rasters where the cursor is displayed. The quad-word transfers must be aligned to quad-word boundaries which necessitates two versions of the cursor, a standard version and a version which is offset one line for use when the cursor image starts on a raster line which does not have a cursor data transfer. The table below shows an 8 line cursor (shown as only 8 pixels wide for brevity) starting on an even raster line.

<table>
<thead>
<tr>
<th>Cursor transfer line no.</th>
<th>Image</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Y 2</td>
<td>VCSR</td>
</tr>
<tr>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Y 4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Y 6</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
</tr>
<tr>
<td>Y 8</td>
<td>VCER</td>
</tr>
<tr>
<td>9</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
</tr>
</tbody>
</table>

(VCSR = Video Cursor Start Register, VCER = Video Cursor End Register)

The corresponding memory image for the cursor would be:

<table>
<thead>
<tr>
<th>Address</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>$00000000</td>
<td>11111111 11111111 11111111 11111111</td>
</tr>
<tr>
<td>$00000004</td>
<td>11111111 11111111 11111111 11111111</td>
</tr>
<tr>
<td>$00000008</td>
<td>22222222 22222222 22222222 22222222</td>
</tr>
<tr>
<td>$0000000C</td>
<td>22222222 22222222 22222222 22222222</td>
</tr>
<tr>
<td>$00000010</td>
<td>33333333 etc</td>
</tr>
</tbody>
</table>

If, however, the cursor image moves and subsequently needs to start on raster 3, data transfer must still start on raster 2, so a second version of the cursor image is required starting with a transparent first line and ending with a transparent line. The memory image for the second version of the cursor would be:
## Attaching an LCD to ARM7500

<table>
<thead>
<tr>
<th>Address</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;00000000</td>
<td>oooooooooo oooooooooo oooooooooo</td>
</tr>
<tr>
<td>&amp;00000004</td>
<td>oooooooooo oooooooooo oooooooooo</td>
</tr>
<tr>
<td>&amp;00000008</td>
<td>11111111 11111111 11111111 11111111</td>
</tr>
<tr>
<td>&amp;0000000C</td>
<td>11111111 11111111 11111111 11111111</td>
</tr>
<tr>
<td>&amp;00000010</td>
<td>22222222 etc</td>
</tr>
<tr>
<td></td>
<td>o = colour transparent</td>
</tr>
</tbody>
</table>
For dual panel LCDs there is the additional problem of when the cursor is required to straddle the panel boundary thus:

<table>
<thead>
<tr>
<th>Cursor</th>
<th>Raster</th>
<th>Image</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>transfer line no.</td>
</tr>
<tr>
<td>Y</td>
<td>234</td>
<td>upper</td>
</tr>
<tr>
<td></td>
<td>235</td>
<td>upper</td>
</tr>
<tr>
<td>Y</td>
<td>236</td>
<td>upper</td>
</tr>
<tr>
<td></td>
<td>237</td>
<td>upper</td>
</tr>
<tr>
<td>Y</td>
<td>238</td>
<td>upper</td>
</tr>
<tr>
<td></td>
<td>239</td>
<td>upper</td>
</tr>
<tr>
<td>Y</td>
<td>0</td>
<td>lower</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>lower</td>
</tr>
<tr>
<td>Y</td>
<td>2</td>
<td>lower</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>lower</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>lower</td>
</tr>
</tbody>
</table>

The ARM7500 must be programmed to change the mode of the VCSR and VCER registers so that cursor data is displayed from the first lower panel raster until VCSR is reached and then on the upper panel from VCER until last upper panel raster is reached. (See ARM7500 data sheet for further details). A pair of cursor images is required allowing the DMA transfers to start at any position in the cursor image and transfer a complete contiguous image.

```
&00000000
11111111
22222222
33333333
44444444
```

Lower panel raster 0 starts cursor data here --->

```
55555555
66666666
77777777
```

VCSR points here --->

```
88888888
```

```
&00000040
11111111
22222222
33333333
```

VCER points here --->

```
44444444
55555555
66666666
77777777
88888888
```

Upper panel raster 239 will end cursor here ----->
As with the case when the cursor is fully contained within one half of the dual panel, a second version of the dual cursor image will be required which is offset by half of a quad-word when the cursor starts on alternate rasters, leading to 4 distinct cursor images.
## A.4 Register Values

For the Hitachi LMG5675XUFC the ARM7500 registers should be set to the following values:

<table>
<thead>
<tr>
<th>Register</th>
<th>Address</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Video palette</td>
<td>&amp;10000000 ; palette pointer=0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&amp;00000000 ; 0 White</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&amp;02002002 ; 1 set first 16 entries in all 4 palettes to 16 grey levels</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&amp;04040404 ; 2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&amp;06060606 ; 3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&amp;08080808 ; 4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&amp;0A0A0A0A ; 5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&amp;0C0C0C0C ; 6</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&amp;0F0F0F0F ; 7 black</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&amp;0B0B0B0B ; 8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&amp;03030303 ; 9</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&amp;08080808 ; 10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&amp;0B0B0B0B ; 11</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&amp;01010101 ; 12</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&amp;09090909 ; 13</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&amp;04040404 ; 14</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&amp;06060606 ; 15</td>
<td>; hres = 640, vres = 480, psz = 1 for dual, 0 for single</td>
</tr>
<tr>
<td>LCD offset register 0</td>
<td>&amp;300000000+((vres/(1+psz)+3)MOD5)*2 ; LCD Off_ 5=6</td>
<td></td>
</tr>
<tr>
<td>LCD offset register 1</td>
<td>&amp;310000000+((3*vres/(1+psz)+8)MOD15)<em>16+(7</em>vres/(1+psz)+4)MOD9 ;LCD Off_15=8</td>
<td></td>
</tr>
<tr>
<td>Border colour register</td>
<td>&amp;40000000                                   ; border colour set to black</td>
<td></td>
</tr>
<tr>
<td>Cursor colour 1</td>
<td>&amp;55050505</td>
<td></td>
</tr>
<tr>
<td>Cursor colour 2</td>
<td>&amp;6A0A0A0A</td>
<td></td>
</tr>
<tr>
<td>Cursor colour 3</td>
<td>&amp;7F0F0F0F</td>
<td></td>
</tr>
<tr>
<td>Horizontal cycle reg</td>
<td>&amp;80000000+ hcr ; hcr = 820-8 must be multiple of 4</td>
<td></td>
</tr>
<tr>
<td>Horizontal sync width reg</td>
<td>&amp;81000000+hswr ; hswr = 58 -8 must be even</td>
<td></td>
</tr>
<tr>
<td>Horizontal border start reg</td>
<td>&amp;82000000+hbsr ; hbsr = 120-12 must be even</td>
<td></td>
</tr>
<tr>
<td>Horizontal display start reg</td>
<td>&amp;83000000+hdsr ; hdsr = 140-18 must be even</td>
<td></td>
</tr>
<tr>
<td>Horizontal display end reg</td>
<td>&amp;84000000+hder ; hder = 780-18 must be even (horizontal resolution = 640)</td>
<td></td>
</tr>
<tr>
<td>Horizontal border end reg</td>
<td>&amp;85000000+hber ; hber = 792-12 must be even</td>
<td></td>
</tr>
<tr>
<td>Vertical cycle reg</td>
<td>&amp;90000000+ vcr ; vcr = 243-2</td>
<td></td>
</tr>
<tr>
<td>Vertical sync width reg</td>
<td>&amp;91000000+vswr ; vswr = 2-1</td>
<td></td>
</tr>
<tr>
<td>Vertical border start reg</td>
<td>&amp;92000000+vbwr ; vbwr = 2-1</td>
<td></td>
</tr>
<tr>
<td>Vertical display start reg</td>
<td>&amp;93000000+vdwr ; vdwr = 2-1</td>
<td></td>
</tr>
<tr>
<td>Vertical display end reg</td>
<td>&amp;94000000+vder ; vder = 241-1 (vertical resolution = 240 for dual panel)</td>
<td></td>
</tr>
<tr>
<td>Vertical border end reg</td>
<td>&amp;95000000+vber ; vber = 242-1</td>
<td></td>
</tr>
<tr>
<td>Vertical cursor start reg</td>
<td>&amp;96000000+address of cursor start ; see notes above regarding</td>
<td></td>
</tr>
<tr>
<td>Vertical cursor end reg</td>
<td>&amp;97000000+address of cursor end ; setting these registers</td>
<td></td>
</tr>
<tr>
<td>External register &amp;C0002005 ; EREG LCDon DACoff Eclk fifo4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Frequency synthesis register</td>
<td>&amp;D00000000+modV*256+modR ; PLL prescaler if used, this panel uses RCLK</td>
<td></td>
</tr>
</tbody>
</table>
Attaching an LCD to ARM7500

Control register &E0000000+F224A ;DUP=1,fifo_loads=16,bits/pixel=4,pixel_rate=clk/3,source=rclk
Data control register &F0011000+(psz+1)*hres*4/32 ; set to twice number of words per raster for dual

Other relevant registers are:-

<table>
<thead>
<tr>
<th>Register</th>
<th>Address Value</th>
<th>Value Notes</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>VIDMUX</td>
<td>&amp;03200000</td>
<td>%000000x0</td>
<td>Set mux off, x depends on sound format required</td>
</tr>
<tr>
<td>CURSCUR</td>
<td>&amp;032001C0</td>
<td>system use</td>
<td></td>
</tr>
<tr>
<td>CURSINIT</td>
<td>&amp;032001C4</td>
<td>address of cursor data start</td>
<td></td>
</tr>
<tr>
<td>VIDEURB</td>
<td>&amp;032001C8</td>
<td>system use</td>
<td></td>
</tr>
<tr>
<td>VIDCURA</td>
<td>&amp;032001D0</td>
<td>system use</td>
<td></td>
</tr>
<tr>
<td>VIDEND</td>
<td>&amp;032001D4</td>
<td>address of last quad-word of video frame</td>
<td></td>
</tr>
<tr>
<td>VIDSTART</td>
<td>&amp;032001D8</td>
<td>address of first quad-word of video frame</td>
<td></td>
</tr>
<tr>
<td>VIDINITA</td>
<td>&amp;032001E0</td>
<td>&amp;F0 Dual panel mode, video &amp; cursor DMA enabled</td>
<td></td>
</tr>
<tr>
<td>VIDINITB</td>
<td>&amp;032001E8</td>
<td>address of first quad-word to be displayed in lower panel</td>
<td></td>
</tr>
</tbody>
</table>

Creating a screen image and setting of DMA registers is identical to the method used for a regular CRT display with the exceptions of where extra DMA registers are required to be set for a dual panel. Creating a screen image and setting of DMA registers is beyond the scope of this document.

B.1 Interface circuitry

Colour TFT panels will generally have a parallel interface data connection with a number of data bits for each of the red, green and blue primary colours. In addition there will be a number of control signals and power supply connections. In the case of the IBM ITSV34A 10.4" Colour TFT LCD panel there are the following connections:-

<table>
<thead>
<tr>
<th>Signal</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Red_data_0 ~ Red_data_5</td>
<td>6 bits of red data</td>
</tr>
<tr>
<td>Green_data_0 ~ Green_data_5</td>
<td>6 bits of green data</td>
</tr>
<tr>
<td>Blue_data_0 ~ Blue_data_5</td>
<td>6 bits of blue data</td>
</tr>
<tr>
<td>DTCLK</td>
<td>Clock at pixel rate</td>
</tr>
<tr>
<td>DSPTMG</td>
<td>Signal indicating valid screen data</td>
</tr>
<tr>
<td>VSYNC</td>
<td>Vertical synchronisation</td>
</tr>
<tr>
<td>HSYNC</td>
<td>Horizontal synchronisation</td>
</tr>
<tr>
<td>CONT1~CONT3</td>
<td>Contrast control (optional)</td>
</tr>
<tr>
<td>VDD</td>
<td>+5V supply</td>
</tr>
<tr>
<td>V33</td>
<td>+3.3V supply</td>
</tr>
<tr>
<td>GND</td>
<td>Signal ground</td>
</tr>
</tbody>
</table>

Although this panel is capable of producing a ¼ million colours the interface design described below limits the user to 256 colours. This is primarily because the screen image data must be routed through the ARM7500 video palette limiting the number of colours to 256, although they may be selected from 32 thousand colours. The limit of 32
thousand colours is set by the ARM7500 being able to produce a maximum of 16 bits of data per pixel clock cycle (5 bits per RGB and one control bit).

Figure B.1.1 shows an example of an interface design.

In the above design a PAL is used to control an external data latch (required to demultiplex the ED port data, see below) and also to provide a means of tri-stating the signals from the ARM7500 should it be a requirement of the design to be able to power down the LCD panel. The logic code for the PAL is given below. The PAL provides a necessary delay to the edge of ECLK which is used to control the latch. An RC delay may also be used.
DEVICE PANEL1 (P16L8)
PIN
ECLK             = 1 (INPUT combinatorial)
ECLKc            = 2 (INPUT combinatorial)
VnCS             = 3 (INPUT combinatorial)
HS               = 4 (INPUT combinatorial)
Envcclcd1        = 5 (INPUT combinatorial)
NC6              = 6 (INPUT combinatorial)
NC7              = 7 (INPUT combinatorial)
NC8              = 8 (INPUT combinatorial)
NC9              = 9 (INPUT combinatorial)
Envcclcd2        = 11 (INPUT combinatorial)
/FETctrl         = 12 (OUTPUT combinatorial active_low)
/LatchOE         = 13 (OUTPUT combinatorial active_low)
/V3enable        = 14 (OUTPUT combinatorial active_low)
/DTCLK           = 15 (OUTPUT combinatorial active_low)
/LatchClk        = 16 (OUTPUT combinatorial active_low)
/HSYNC           = 17 (OUTPUT combinatorial active_low)
/VSYNC           = 18 (OUTPUT combinatorial active_low)
/NC19            = 19 (OUTPUT combinatorial active_low);

BEGIN  "Logic Definitions"
FETctrl = /Envcclcd1;
LatchOE = /Envcclcd1;
V3enable = Envcclcd1;
LatchClk = /ECLKc;

IF (/Envcclcd1) THEN
BEGIN
    HSYNC = /HS;
    VSYNC = /VnCS;
    DTCLK = /ECLKc;
END;
ELSE
BEGIN
    HSYNC = 1;
    VSYNC = 1;
    DTCLK = 1;
END;

ENABLE(FETctrl, LatchOE, LatchClk, V3enable);
ENABLE(DTCLK, HSYNC, VSYNC);

END. "End of file"
B.2 System Functional Description

To generate a 256 colour image on a panel such as the IBM ITSV34A the ARM7500 video system should be setup for 8 bit/pixel mode at a resolution matching that of the target panel (in this case 800 x 600). In 8 bit mode the pixel data byte will be presented to each of the red, green and blue palettes. If the ARM7500 is set into duplex mode then two bytes will be output through the ED port for each ECLK cycle. These bytes will be the outputs from the red palette and the green palette. The fact that these are red and green outputs does not mean that they provide data only for the red and green inputs of the panel. Instead they should be considered as the two halves of a 16 bit word which contains 5 bits for each of the three colours and one control signal (DT, denoted as dsptmg in the above schematic).

The mapping of palette-output to panel-input is as follows:-

| Palette-output | G7 G6 G5 G4 G3 G2 G1 G0 R7 R6 R5 R4 R3 R2 R1 R0 |
| Panel-input    | DT B5 B4 B3 B2 B1 G5 G4 G3 G2 G1 R5 R4 R3 R2 R1 |

The control signal DT must be high whenever there is valid screen data, therefore this bit must be set to 1 for all green palette entries and cursor colour entries, but set to zero for border colours.

The pixel clock (ECLK) rate is far less critical with TFT panels than with STN panels although panel manufacturers will often give typical timing specifications that are equivalent to CRT timings.

Figure B.2.1 shows the pixel data timing.

$t_{clock}$ is typically $\frac{1}{4}$ of the ECLK clock period (approx 6ns at 40MHz).
B.3 Cursor considerations

With a single panel LCD there are no special cursor considerations above those required for a regular CRT display, ie non-quad-word aligned cursor DMA starts, which are solved by having two images of the cursor as described in section A.3.
B.4 Software example

The following program extracts show the register values used to drive the IBM ITSV34A panel. The full program, written in BBC BASIC, which includes a 256 colour palette is available on request from Acorn Risc Technologies.

```
DEFPROC panel_parms
  hres%=800:vres%=600:REM single panel 800x600
  REM modV%=3 :modR%=4 :REM Vclk=25.8MHz
  REM modV%=6 :modR%=8:REM Vclk=24MHz
  modV%=5:modR%=4:REM Vclk=40MHz
  fifo%=&0200:pixrate%=&0:clksrc%=0:
  lcdbpp%=&60:REM 8 bits/pixel
  REM subtractions taken care of in assembler
  hswr%= 128         :REM  even
  hbsr%= 88 +hswr% :REM  even
  hdsr%= 0 +hbsr% :REM  even
  hder%=hres% +hdsr% :REM  even
  hber%= 0 +hder% :REM  even
  hcr%=   40 +hber% :REM  div4 (N-8)

  vswr%=   4         :REM
  vbsr%=   23 +vswr% :REM
  vdsr%=   0 +vbsr% :REM
  vder%=vres% +vdsr% :REM
  vber%=  0 +vder% :REM
  vcr%=    1  +vber% :REM
ENDPROC

DEFPROC make_palette
  FORI=1TO256
    READ R,G,B
    R=R DIV 8:G=G DIV 8:B=B DIV 8
    Y=R+(G*32)+(B*1024)+&8000
    !(vidclist%+(I*4))=Y
  NEXTI
DATA    0 ,  0,   0
DATA   17,  17,  17
DATA   34,  34,  34
DATA   51,  51,  51
DATA  204, 204, 204
DATA  221, 221, 221
DATA  238, 238, 238
DATA  255, 255, 255
ENDPROC

DEFPROC asssm
  FOR pass=0 TO 2 STEP 2:P%=code
```

---

**Applications Note 35**

ARM DAI 0035 B
Attaching an LCD to ARM7500

[ OPT pass
ALIGN
EQUD 0

; *** Switch on LCD panel ***
.lcdnon
SWI EnterOS ;enter supervisor mode
TEQP pc,#I_bit+SVC_mode ;kill interrupts
;
; read and store CRT DMA register value and then switch off DMA
MOV R2, #$03200000 ;IOMD base
LDR R0, [R2,#&1E0] ;Video DMA reg
STR R0, dmareg% ;save old DMAreg value
MOV R0, #$0 ;DMA off
STR R0, [R2,#&1E0] ;Video DMA reg
MOV R0, #$03
STR R0, [R2, #&06C] ;VIDMUX set to one
;
; then transfer VIDC control register values listed below into VIDC
MOV R1, #$03400000 ;Vidc base
ORR R1,R1,#&80000000 ;phys space
LDR R3, reg_start
LDR R4, reg_end
.vloop
LDR R0, [R3], #4
STR R0, [R1] ;program VIDC with list values
CMP R3, R4 ;between vidclist% and vidcend%
BNE vloop
MOV R0, #$70 ;select Single panel, DMA on
STR R0, [R2,#&1E0] ;Video DMA reg
B exit
;
; *** Register values for VIDC in LCD panel mode ***
.vidclist% EQUD &10000000 ; palette pointer=0
;
; VIDC20 Palette to Panel data mapping
;    32
;    0 0 0 0 E E E E B B B B B B B B G G G G G G G R R R R R R R R R R
;       | \   / \   / \   / \   / \   / \   / \   / \   /   \     /    
;         \     blue   green   red
;              DSPTMG 5 4 3 2 1 5 4 3 2 1 5 4 3 2 1
; Colours 0 to 255
EQUD $00000800 ; 0 black

EQUD $0000FFFF ; 255 white
;
; and the rest of the vidc registers
EQUD $30000000 ;offset for 5 and 2 frame duty cycle grey scales
EQUD $31000000 ;offset for 15 and 9 frame duty cycle grey scales
;all set to zero for colour panel
;

Applications Note 35
18
Attaching an LCD to ARM7500

EQUD $40000000 ; border colour = black, DSPTMG = low
;
EQUD $5000FF20 ; Cursor Palette colour 1
EQUD $6000FC00 ; Cursor Palette colour 2
EQUD $7000FC00 ; Cursor Palette colour 3
;
EQUD $80000000+hcr%-8 ; HCR even
EQUD $81000000+hswr%-8 ; HSWR even
EQUD $82000000+hbsr%-12 ; HBSR even
EQUD $83000000+hdsr%-18 ; HDSR even
EQUD $84000000+hdcr%-18 ; HDCR even
EQUD $85000000+hber%-12 ; HBER div 4
;
EQUD $90000000+vcr%-2 ; VCR
EQUD $91000000+vswr%-1 ; VSWR
EQUD $92000000+vbsr%-1 ; VBSR
EQUD $93000000+vdsr%-1 ; VDSR
EQUD $94000000+vder%-1 ; VDER
EQUD $95000000+vber%-1 ; VBER
;
EQUD $C0000004 ; VSYNC HSYNC HiRes=off grey-scale=off DACs=off
; RGB-pedestals=off EREG(7.4)=0000 ECLK=on EREG(1.0)=00
;
EQUD $D0000000+modV%*256+modR% ; PLL prescalers, sets Vclk
;
EQUD $E0000000+fifo%+lcdbpp%+pixrate%+clksrc% ; control register
;
EQUD $F0011000+hres%/4 ; data control register
;
.reg_start EQU vidclist%
.reg_end EQU vidcend%

ENDPROC
REM *** end of file ***
4 Products and services available

Various products and support services for the ARM7500 are available from both Advanced RISC Machines Ltd (ARM) and Acorn RISC Technologies (ART).

**Services available from Advanced RISC Machines Ltd (ARM)**

- An experienced consultancy department offers design services at chip, board and product levels.
- Asic design and CoDesign and verification tools.
- Application software group offers off the shelf components (soft modems, telecommunications, TCP/IP etc)
- Real time operating systems
- Software development tools (including compilers, assembler, debugger and project manager)
- ARM training, in all aspects of ARM CPUs

For more information on these services see the ARM web site:

http://www.arm.com/

**Services available from Acorn RISC Technologies (ART)**

- Standard products and components based on ARM technology.
- Development and prototype systems
- Licensing of RiscOS (a mature desktop operating system) in full, or in part.
- Consultancy and technical support
- ARM based production management and production support
- Full schematics for Stork and NewPAD designs (both ARM7500 based portables)

The example BASIC driver programs for the LCDs mentioned in this note are available from ART.

For more information on these services see the Acorn web site:

http://www.acorn.com/