# Test System Graphics Capability

The test system is limited to 672x384x8 as a maximum graphics resolution. One may wonder why as the display device is a widescreen TV with a resolution of 1366x768x24.

A sample calculation of the bandwidth required for a full resolution display shows why the test system’s capabilities are limited.

## Sample calc's

* 5, 24 bit pixels fit into 1, 128 bit memory strip

1365 pixels (rounding to a multiple of 5) = 273x5 so 273 memory strips per pixel row

273 \* 768 pixel rows = 209664 fetches per screen

209664 fetches in 60 Hz = 0.0166666 /209664 = 79 ns per fetch

79ns @ 50MHz memory clock rate = 2.5 clock cycles. (The 50 MHz clock rate is set by the DDR2 interface component. It reflects the fact that the DDR2 memory is accessed in burst mode with a burst length of 8, so that 16 bytes are fetched per burst).

To be safe we round down to 2 clock cycles.

Since it takes a number of clock cycles (eg 6-10) to access memory this is greater than

the max that could be supported. However, also the video buffer is only

256 kbytes and 1365x768x24 would require a 4 megabyte buffer.

Using the fact that the development board only supports 12 bpp output, the required memory bandwidth and buffer size can be halved. Even at ½ bandwidth it still too much for the system to support. A memory fetch would have to take place every 5 clock cycles. There would be no bandwidth left for the CPU or graphics accelerator.

The next step is to reduce the resolution requirements. But to what ? A simple solution is to use a ½ resolution screen of 683x384. Doing so also reduces the memory requirements to ¼ that of a full resolution screen. In order to reduce the screen buffer size requirement at high resolution the display is limited to 8bpp. This mean a 256kB screen buffer can be used.

Paradoxically, locating the screen buffer in block RAM offers potentially more available bandwidth. However block RAM resources are limited.

Using block RAM rather than the DDR memory may allow for some snazzier graphics. For instance roto-zoom should be possible.

### So at 672x384x8bpp

* 16, 8 bit pixels fit into 1, 128 bit memory strip

672 pixels (rounding to a multiple of 16) = 42x16 so 42 memory strips per pixel row

42 \* 384 pixel rows = 16128 fetches per screen

16128 fetches in 60 Hz = 0.0166666 /16128 = 1.033us per fetch

1.033us @ 50MHz memory clock rate = 51.7 clock cycles. Note this is a much lower memory bandwidth requirement and should be easily achievable.

To be safe we round down to <=50 clock cycles.

Even if it takes a dozen clock cycles per memory access there’s now ample room left over to support a CPU and graphics accelerator.