Pong (PONG)

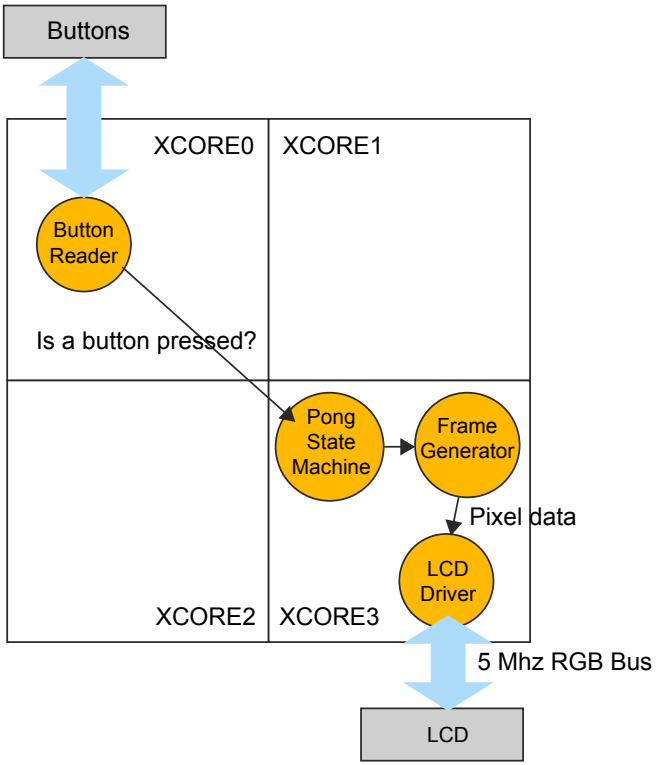
Pong is a simple application that uses four threads.

The state machine polls the button thread to see if a button is currently pressed, makes the main game calculations, updates the positions of the pads and ball and passes them to the frame generator.

The frame generator draws the raw pixel data, and passes it through a channel to the LCD driver thread to output to the screen.

The demonstration is divided across four XCores due to the device mapping on the XDK.

Use the two left buttons to move the left paddle up and down, and the two right buttons to move the right paddle.

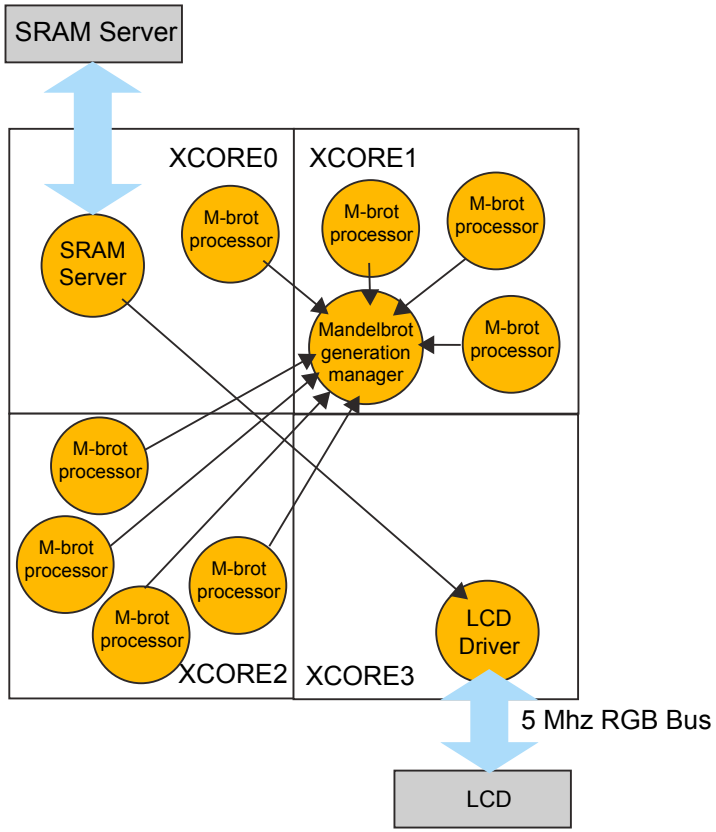


Mandelbrot (MBROT)

The Mandelbrot demonstration uses the parallel feature of the XMOS architecture to farm out the processing to multiple threads running across the XCores.

The generation manager requests Mandelbrot calculations from eight threads running the escape time algorithm. It then passes the resulting pixel data to the SRAM server for storage.

In parallel, the SRAM server pipes the RGB data to the LCD driver, which displays the image on the LCD.

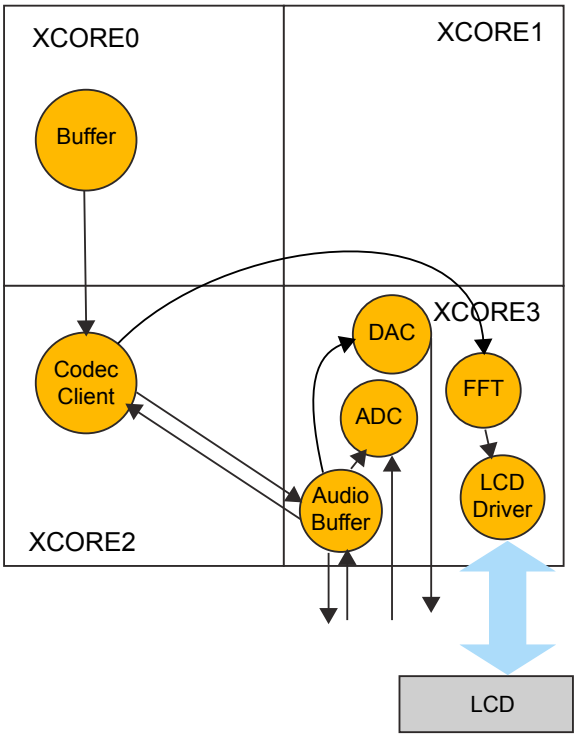


Audio Frequency Analyser (FREQ)

The FREQ program reads audio input, and visualises the power in each frequency band in real-time on the screen. Connect a microphone or MP3 player to the line input to run the program.

The horizontal axis of the screen is time, and the vertical axis represents the frequency bands. The sound intensity is represented by colour—dark colours indicate little power in the band, and bright colours indicate high power. The power spectrum is updated in real-time using a sliding window, which represents about a second worth of audio.

Internally, the program computes a 512-sample FFT, and uses a lookup table to map the magnitude of each frequency onto a colour. The FFT is computed 200 times per second with a 75% overlap—128 new samples are appended to 384 of the previous samples and the FFT is computed over these 512 samples. The real and imaginary components are squared and summed, the sine and cosine parts are then added, and the resulting value is used to lookup a colour.



Menu system, loader and LCD screen (XDK)

The XDK menu program is provided for users who want to change the menu for their own use. If you run XDK.XB, the XDK will appear to reset itself, as it is re-runs the menu; no other behaviour is expected.

The program uses SRAM, LCD, and SD/FAT16 files system software components to display the menu system.

The program uses the SD/FAT16 component to load images from the SD card into frame buffers in the SRAM.

The LCD component includes a thread, which reads frame data from the SRAM and buffers it locally. A *character buffer* thread calculates pixel values using a local character buffer. For each pixel in a frame, these two threads output data through a channel to a *blender thread* that does a simple blending operation. The *blender thread* outputs the blended pixel data to the LCD server thread, which provides the interface to the LCD.

The main menu program thread uses two channels to communicate with the LCD component, one for text and the other for graphics. The text channel adds characters to, and removes them from, the character buffer. The graphics channel is used for frame update commands and other system requirements.