# Q·Kernel
## The new generation RTOS

A new breed of embedded applications is rapidly evolving. Traditional DSP applications are adding networking and other control functionality. But at the same time, the typical MCU control application will often include High Dataflow requirements like streaming media and other DSP functions. An emerging solution for this new class of 'hybrid' application is the convergent processor. This design approach combines both DSP and RISC/microcontroller capabilities into a single, unified architecture.

A processor that implements this architecture can operate as a DSP engine, be totally dedicated to a control application, or can operate somewhere in between. This makes those processors suitable for everything from industrial control to portable devices. Single convergent processors are an attractive alternative to the larger and more costly RISC and DSP processors. The dsPIC and PIC32MX from Microchip are convergent processors that can replace the RISC and DSP processors.

While Microchip implemented the hardware for convergent processing, the software is often lacking. A traditional multithreading RTOS adds enormous overhead to the DSP portion of the application. While a simple scheduler may work fine for the DSP or High Dataflow part of the application, it is not a good solution for a control application.

## The Traditional RTOS

The traditional RTOS requires a stack for every thread, so it can block while waiting for an event and can be preempted by a higher priority thread. The switching between threads, known as context switching, is an expensive operation especially for processors with lots of registers like the dsPIC/PIC24 (20) and PIC32 (40). The larger the context, the longer it takes to switch the context.

DSP and High Dataflow applications typically read a block of data, operate an algorithm on the data, and then send the data to another programming unit for further processing. Due to the real time nature of the data, the algorithm must start within a very tight window once the data becomes available. Developers often design their own custom executive that handles the High Dataflow based on a corporative scheduling model. To combine this model with a traditional RTOS they run the algorithms in a high priority thread which adds a lot of overhead.

## The Dual-Mode RTOS

*Q·Kernel* combines the traditional thread-based kernel architecture for real-time control processing with specialized fibers for High Dataflow operations. The architecture accommodates the different needs for both domains by separating them. *Q·Kernel* enables both types of application code to run fully optimized on a single processor, and both fibers and threads use a common API.

In order to meet real-time requirements, the DSP and High Dataflow processes run as fibers, at a priority that is higher than control threads which ensures they get access to the CPU. These fibers are lightweight because they have no context, making the switch from fiber to fiber very fast. Furthermore, fibers run at a priority just below that of interrupt handlers, a position that tends to reduce startup latency and minimize jitter.

# QuasarSoft