I have a PIT chore logging 2 ADC channels @ 1 sps (16 bit, via QSPI @ 200kHz) which is working in parallel with the acquisition of large (32kB) buffers of RS232 data coming from a TPU UART.

it is not clear to me the role of:
1) interactions between QSPI and TPU Uart, or Timer and TPU Uart.
2) Priority of PIT?
3) I knew that RTC is related to PIT and QSPI, but not the TPU Uart. Am I missing something?

=======================================

The PIT, QSPI, and TPU are entirely separate modules within the 68332. The only interactions between these involve competition for CPU resources to process data or interrupts, or inter-dependencies created in the application program to move data between the modules.

The TPU UARTs are simply channel-assigned functions performed by the TPU module.

The RTC on the CF2 is actually two clocks running in parallel. One runs in the MSP430 supervisor IC and the second is implemented with a custom TPU function permanently assigned to TPU channel 0 and managed by PicoDOS to provide real-time clock capabilities. Any RTC function synchronizes the TPU clock to the MSP430 RTC clock

The CF2 default priorities are defined in the “IEV Interrupt Exception Vectors” header file <_cfx_internals.h> and initialized by PicoDOS with the default values at startup. In most CF2 applications, the PIT controls timing for data acquisition operations, so it is given the highest priority (6) to minimize jitter noise. In exchange, PIT ISRs or chores are expected to be short (hundreds of microseconds) or otherwise dynamically manage the priority mask levels or rearrange the priorities entirely.

The other modules have the following interrupt priority levels: QSPI=5, SCI UART=4, and TPU=3 which seems to best suit most CF2 applications when the normal conventions are followed.

A PIT ISR or chore that executes for longer than the time between receipt of two UART (either SCI or TPU or the same channel) characters may cause the loss of some UART data. Usually the best way to deal with that is to limit the PIT operation to initiating the critically timed acquisition operation and exiting after setting a global flag to tell the main process to handle the result of the acquisition with relaxed timing. Most of the Persistor ADC examples use a PIT interrupt to start the QSPI and immediately exit, then rely on the QSPI completion interrupt to set a global flag telling the main process that new data is ready for processing. The simplest example of this is the LowPowerAD target from the A-DExamples project.

Section 6 "Exception Processing" of the CPU32 Reference Manual describes the interrupt mechanics in full detail.