



EPICS Support Module for Efficient UDP Communication with FPGAs

Martin Konrad

MICHIGAN STATE
UNIVERSITY

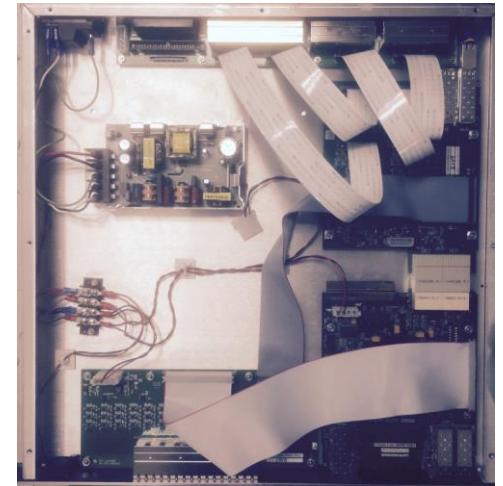


U.S. DEPARTMENT OF
ENERGY

Office of
Science

Hardware and Requirements

- 350 low-level RF controllers
- 55 machine protection nodes → driver needs to be reliable
- FPGA based, firmware is continuously improved
→ driver needs to be flexible
- Soft-core runs a C program handling Ethernet communication
→ limited to UDP
- IOC runs on remote machine



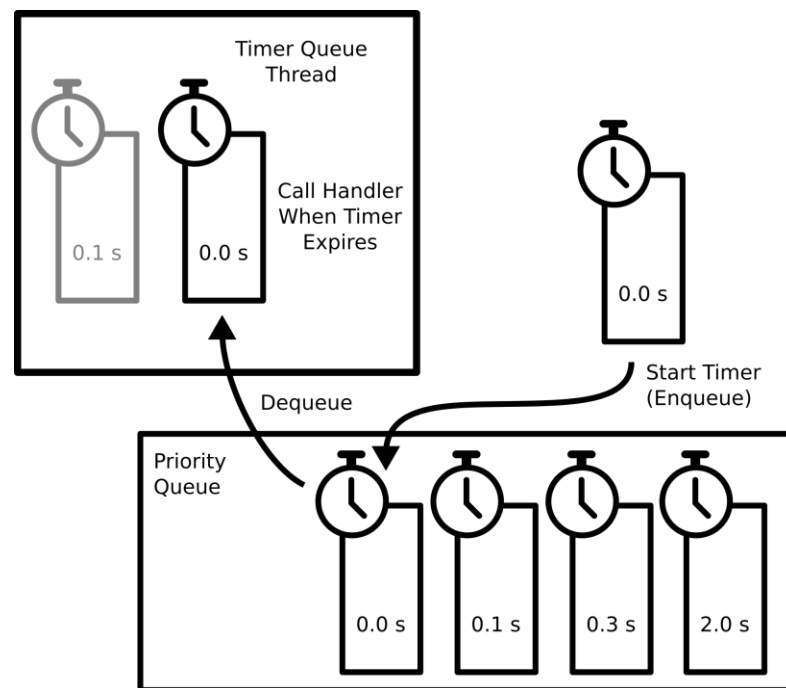
Features

- Read/write scalars of various data types
 - Signed int
 - Unsigned int
 - Fixed point 16.16
 - ...
- Read/write waveforms
- Driver doesn't need to be recompiled if registers are added/modified
- Firmware upgrade through waveform records

Firmware				
Memory Write Protection		Write Allowed		
Image		Progress		SHA1
		Write	Read	
Multi-boot Header	Write...	0%	100%	f5649a4193a9802eab9f0f59534a330a4fc2e5a8
Golden FPGA	Write...	0%	100%	81e68cba5a1f8cda90d0e02f850aa0ceddf254e4
Golden MicroBlaze	Write...	0%	100%	82d02088565ceb6f7e1ac3cdc8e2cd669d6924b0
Primary FPGA	Write...	0%	100%	81e68cba5a1f8cda90d0e02f850aa0ceddf254e4
Primary MicroBlaze	Write...	0%	100%	82d02088565ceb6f7e1ac3cdc8e2cd669d6924b0

Support Module is Timer Driven

- Asyn is used to simplify implementation of asynchronous I/O
- Timer are used to perform the following operations
 - Periodic read-out of registers
 - Incremental array read/write
 - Detecting communication problems
 - ...
- `epicsTimerQueue` is managing the timers
- I/O operations are performed in the context of the timer queue thread
 - No separate locking required
- Much easier to write fast unit tests

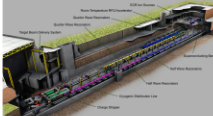

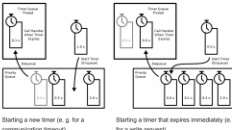




Thanks!

Poster MOPHA075 on Monday

EPICS Support Module for Efficient UDP Communication With FPGAs

Martin Konrad, Enrique Bernal, Mark Davis
Facility for Rare Isotope Beams (FRIB), Michigan State University, East Lansing, MI 48824 USA

FRIB <ul style="list-style-type: none">New heavy ion accelerator facility for nuclear physicsDriver linac is designed to accelerate all stable ions to energies ~200 MeV/uBeam power on target is up to 400 kW350 cavities, 300 IOC's, thousands of devices 	Hardware and Requirements <ul style="list-style-type: none">Used with two in-house developed systems based on a low-cost pizza-box design with a Spartan 6 FPGA<ul style="list-style-type: none">~350 low-level RF controllers~55 machine-protection nodes→ Driver needs to be very reliableSoft-core runs a C program which handles Ethernet communication<ul style="list-style-type: none">No operating system, no IP stack → need to use UDPLLRF firmware is improved continuously, registers are added/modified frequently→ Driver needs to be flexible 	Design Decisions <ul style="list-style-type: none">AsynPortDriver is used to simplify implementation of asynchronous operationsAsyn parameters are dynamically created while loading the IOC database<ul style="list-style-type: none">Register addresses and data types (signed int, unsigned int, fixed point 16, 16 etc.) are read from INP/OUT fieldsStrings are parsed using std::regex (C++11)Registers are read out periodically as a large array rather than transferring them one by one to improve efficiencyTimers are leveraged for implementing operations not supported by Asyn
UDP Protocol <ul style="list-style-type: none">IOC initiates communication by sending a UDP request package, the device generally responds with one or more UDP packagesSpecified commands:<ul style="list-style-type: none">Read registersWrite registersRead waveformRead a block from persistent memoryErase a block of persistent memoryWrite a block of persistent memoryRequest write accessThree different memory regions are defined for registers<ul style="list-style-type: none">Read-only memory (e.g. read out a sensor)Read/Write memory (e.g. read/write set-point)"Write-once" memory (for commands triggering some action like "start ramp")Multiple clients can read from the device at the same time but only one of them can have write access<ul style="list-style-type: none">Session ID in each package helps to distinguish clientsWrite access expires when not used for a while	Support Module is Timer Driven <ul style="list-style-type: none">Timers are used to perform the following operations<ul style="list-style-type: none">Periodic read-out of registers (10 Hz)Incremental array read/write without blocking other communicationDetecting communication problems (timeouts)Periodically sending a request to keep write access (not needed when writes are performed)An instance of epicsTimerQueue is managing the timers<ul style="list-style-type: none">Timers are stored in a priority queue (ordered by expiration time)The timer-queue thread waits until the first timer in the queue expires before calling its handlerEach device has its own timer queue resulting in one additional thread per deviceWrite operations start a timer that expires immediately<ul style="list-style-type: none">It is inserted in the front of the timer queue and the handler is called right awayWrites are thus effectively handled with higher priority than reads and waveform transfers  <p>Starting a new timer (e.g. for a communication timeout)</p> <p>Starting a timer that expires immediately (e.g. for a write request)</p> <p>Advantages</p> <ul style="list-style-type: none">All IOC operations are performed in the context of the timer-queue thread (no separate locking required)Asynchronous implementation with timers makes the code easier to test (avoids "sleep" statements that would slow down unit tests)	
Waveform Read-out <ul style="list-style-type: none">Reading a waveform record initiates an asynchronous read<ul style="list-style-type: none">1. The PACT flag of the waveform record is set when the record is processed2. The driver requests the entire array3. The device transmits the array chunk by chunk4. If a chunk is lost a timeout expires and the missing part is requested again until all blocks have been received5. The assembled array is passed on to Asyn6. Asyn clears the PACT flag and triggers processing of the recordDevices can support a command for freezing circular buffers. This can be implemented on the IOC as a chain of records that freeze, read and unfreeze a buffer.  <ul style="list-style-type: none">A read-out mode geared towards streaming applications is under development.	Firmware Update <ul style="list-style-type: none">Users read/write firmware images through waveform recordsProcessing the read/write waveform record initiates asynchronous read/write<ul style="list-style-type: none">1. Asyn sets PACT flag before it calls write routine2. Data is read/written block by block (timer driven), existing blocks before writing them3. When all data has been read/written, the data is passed to Asyn (Asyn clears PACT flag)4. When firmware write has completed, the firmware image is automatically read to update read recordsAn aSub record calculates the SHA1 hash of the data read back. This hash can be compared to the output of "sha1sum firmware bin" on the command line<ul style="list-style-type: none">The hash is archived as a record of firmware updatesIn the future the hash value could be used to revoke run permit while invalid firmware is loadedFPGA engineers can update firmware themselvesAccess security prevents non-experts from modifying firmware 	Development Tools <ul style="list-style-type: none">Compiler: GCC/clang using C++14Build system: CMake, CMake+EPICSCode coverage report: gcovUnit testing framework: Google Test/Google Mock<ul style="list-style-type: none">Mock asynOctalSyncIO functionality to verify that the support module calls Asyn's read/write functions correctlyMock object is injected for unit tests and asynOctalSyncIO for production <p>Conclusion</p> <ul style="list-style-type: none">Support module is used successfully for LLRF controllersProtocol and support module are generic (device/application agnostic)Both scalar read/write and waveform readout are supportedFirmware can be programmed and verified over Channel AccessDriver handles large number of devices efficiently (largest IOC has 168 devices and 220,000 records)

FRIB  Facility for Rare Isotope Beams
U.S. Department of Energy Office of Science
Michigan State University

This material is based upon work supported by the U.S. Department of Energy Office of Science under Cooperative Agreement DE-SC0009801.
Michigan State University designs and establishes FRIB as a DOE Office of Science National User Facility in support of the mission of the Office of Nuclear Physics.



Facility for Rare Isotope Beams
U.S. Department of Energy Office of Science
Michigan State University