artdaq - Idea #7244

Create a reusable UDP FragmentGenerator

10/30/2014 12:08 PM - Kurt Biery

Status:	Closed	**Start date:**	10/30/2014
Priority:	Normal	**Due date:**	
Assignee:	Eric Flumerfelt	**% Done:**	90%
Category:		**Estimated time:**	40.00 hours
Target version:	v1_12_13		
Experiment:	-		

Description

In our discussions with potential *artdaq* users, we've heard that it would be nice to have *artdaq* easily handle the receiving of UDP data packets from hardware modules. Given that, it would probably be worthwhile to create a simple UDP packet FragmentGenerator that would listen for UDP packets and stuff them into artdaq::Fragments, as usual. If we make this FragmentGenerator part of the *artdaq* package, then it could simply be used out-of-the-box when someone wants such functionality.

Of course, there should probably be associated work to define some of the details of the UDP packet. For example, if the UDP packets that are sent to this FragmentGenerator all have a sequence number in a well-defined place, the FragmentGenerator could use that as the artdaq::Fragment sequence number, or at least cross-check the two.

History

#1 - 10/30/2014 12:09 PM - Kurt Biery

- Estimated time set to 40.00 h

#2 - 11/21/2014 04:58 PM - Eric Flumerfelt

- Assignee set to Eric Flumerfelt

#3 - 05/04/2015 01:07 PM - Kurt Biery

- Target version changed from 577 to v1_12_10

#4 - 05/04/2015 01:41 PM - Eric Flumerfelt

The implementation of a UDPReceiver_generator class has been done for the OTS DAQ proof-of-concept (https://cdcvs.fnal.gov/redmine/projects/otsdaqpoc). Once the reference implementation is complete, it will be ported into one of the artdaq repositories (probably artdaq-demo). The UDPReceiver_generator implements the CAPTAN UDP protocol, which uses the first two bytes out of a standard MTU 1500 UDP packet to describe the data, and the rest of the packet is user-configured data payload.
A summary of the CAPTAN protocol:
Control Packet: From ARTDAQ to UDP-sending hardware:
Byte 0: OpCode: Defined values:

0	Read Register
1	Write Register
2	Burst Data Start (Start Run)
3	Burst Data Stop (End Run)

Byte 1: CmdSize: Size of data to Read/Write (if Write, data is from the payload of the packet)
Bytes 2-9: Address: 64-bit register address for Read/Write

Data packet: From UDP-sending hardware to software:
Byte 0: Packet Type:
First Hex: Data stream type:

0	Read Response
1	First in burst
2	Mid-burst
3	Last in burst

Second Hex: Data payload format:

0	Raw data
1	JSON string data

Byte 1: Sequence ID: Should increment for each packet sent (for UDP QOS, increments mid-burst as well)

#5 - 06/02/2015 05:49 AM - Kurt Biery

- Target version changed from v1_12_10 to v1_12_11

#6 - 06/05/2015 10:04 AM - Eric Flumerfelt

- Status changed from New to Assigned

- % Done changed from 0 to 90

Code has been imported from artdaq-ots to artdaq-demo (branch feature/UDPReceiver). A corresponding UDPFragment class has also been created for artdaq-core-demo (same branch name). The code should be vetted before being included in the official artdaq-demo releases.

#7 - 08/03/2015 10:42 AM - Kurt Biery

- Target version changed from v1_12_11 to v1_12_13

#8 - 10/30/2015 09:37 AM - Kurt Biery

- Status changed from Assigned to Resolved

This functionality has been available in the artdaq-demo for some time.

#9 - 05/23/2016 10:23 AM - Eric Flumerfelt

- Status changed from Resolved to Closed