DOI 10.5162/etc2014/7.2

Moving Data Analysis into the Acquisition Hardware

Tian Fanzheng[1], Dave Buckley[2]
[1] China Flight Test Establishment, China
[2] Chief Architect, Curtiss-Wright, Ireland

Abstract

Data acquisition for flight test is typically handled by dedicated hardware which performs specific functions and targets specific interfaces and buses. Through the use of an FPGA state machine based design approach, performance and robustness can be guaranteed. Up to now sufficient flexibility has been provided by allowing the user to configure the hardware depending on the particular application. However by allowing custom algorithms to be run on the data acquisition hardware, far greater control and flexibility can be offered to the flight test engineer. As the volume of the acquired data increases, this extra control can be used to vastly reduce the amount of data to be recorded or telemetered. Also real-time analysis of test points can now be done where post processing would previously have been required. This paper examines examples of data acquisition, recording and processing and investigates where data reduction and time savings can be achieved by enabling the flight test engineer to run his own algorithms on the hardware.

Key words: data acquisition, data reduction, real-time analysis, custom algorithms, FFT.

Introduction

Modern data acquisition cards perform specific functions and target specific interfaces and buses. They perform one task or function repeatedly following a state machine based design approach. They are typically implemented on FPGAs to maximize performance and robustness. For the vast majority of acquired interfaces and parameters this is the ideal approach. Sufficient flexibility can be offered through the configuration of certain settings. For example the gain, sample rate or filter cut-off of an analog acquistion card can be programmed per channel. Similarly for bus monitors, the messages and parameters that are to be selected for recoding or telemetry can be preconfigured [1].

Typically all the acquired data will be recorded onboard and analysis tools will be used to postprocess the recorded data. Some analysis is also done in real-time on critical parameters. These critical parameters are telemetered to ground where they are analyzed as the parameters are received.

However the volume of acquired data is ever increasing. The extensive use of new technologies and materials, and the continuous improvement in aircraft capability and performance has led to a sharp increase in the variety and quantity of parameters acquired during a flight test campaign. Between fifteen and twenty thousand parameters are expected to be acquired during testing of the Comac C919 aircraft. Given that a large portion of these will be wideband parameters the total throughput of the acquisition network will be in the region of 300 Mbps. As a consequence a flight test sortie will record up to one terabyte of data. The task of managing and analyzing all this extra data is compounded by the requirement to shorten the flight test cycle [2].

Unfortunately the bandwidth available to telemetry has not kept pace with the increase of data acquired. This means that an ever decreasing subset of the acquired data is available to be telemetered to ground.

There are a number of ways of overcoming these problems. Data reduction is a means of reducing the amount of raw data that needs to be recorded or telemetered. Some form of processing is done on the data so that only the information contained within the parameter stream that is of interest is recorded or telemetered. This could be as simple as only selecting data around a given event or it could be the transfer of information into another domain so that the dominant frequencies of the measurement are all that need to be stored or transmitted to ground.

Another option would be to do the analysis on board. In this paradigm the same algorithms that would normally be run on ground would now be run on the test article and only the result of the analysis needs to be telemetered or recorded. Of course data reduction and onboard processing are already supported to some extent on existing data acquisition hardware. Bus monitors can be configured so that only certain messages or parameters are acquired.

interface as all the other data acquisition cards. This will allow it to be placed in any slot in any chassis.

To maintain flexibility and allow the card to process data from any bus or interface it should not directly acquire any data. Acquired data from any module in the chassis can be passed to the module for processing. The processed data can then be fed via the backplane to any transmitter in the chassis. In fact in a fully networked system the card can be placed

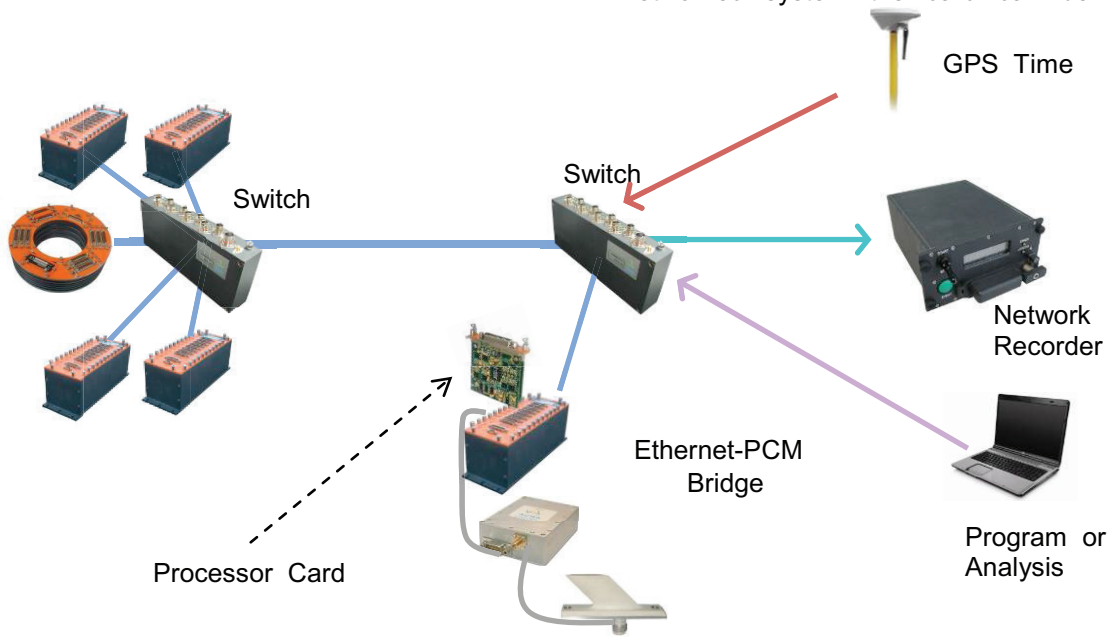

Fig. 1. Processor card in a networked system

Recording can be enabled or disabled based on certain trigger conditions. Analog data is typically oversampled and filtered before being transmitted or recorded at the desired sample rate.

However with existing data acquisition hardware the flight test engineer is limited to the options provided by the data acquisition vendor. Many of the scenarios under which he would like to do data reduction or onboard processing may not be available to him in off the shelf hardware. For these scenarios a more flexible module, which allows the flight test engineer to run his own algorithms on the acquisition system, is required.

This paper examines what platform a flexible module should provide, how it could be deployed in the data acquisition system and discusses some example use cases.

System Requirements

The first requirement of a flexible module would be that it can be placed in any location in any configuration. To this end the module should have the same form factor and backplane

anywhere in the system and data from any acquisition card in any chassis in the system can be routed to it. The processed data can then be routed back to any card in any chassis in the system for the purposes of recording or telemetry.

In figure 1 the processor card is placed in the chassis which houses the Ethernet to PCM bridge. This is a good location as all the parameters that are intended for telemetry to ground are available to the processor card via the chassis backplane. However the processor card can in fact be placed in any of the chassis in the system and the processed data can be routed to the analysis laptop, recorder or Ethernet to PCM bridge.

Hardware Platform

The next consideration is the hardware which would provide the platform for the flight test engineer to run his own algorithms. Due to constraints on the test article data acquisition cards are typically designed to be small, light and low power. Any processor used in a data acquisition card would need to minimize size and power consumption. Also given that the processing required on the data would include

digital filtering and FFTs a DSP co-processor would significantly increase the number and complexity of algorithms that could be performed.

During the initial development phase of this flexible card many microprocessors were considered and their strengths and weaknesses were compared.

The processor chosen was the Texas Instruments OMAP L138 microprocessor. TI developed this processor as a lower power application processor which has gained wide traction in the industry due to the wide range of peripherals with low power requirements.

The processor has multiple cores including a general purpose ARM9 core. This will be familiar to anyone with exposure to modern processors. It is commonly used on embedded processors and a typical mobile phone will have one or more of these cores.

There is a powerful DSP co-processor onboard. This is based on the TI family of DSPs and supports both fixed and floating point instruction sets.

Of course the most attractive feature of this processor is the low power consumption. In typical applications the power drawn by the device is less than 350mW.

While no data is acquired by this module the flight test engineer may want to have direct access to the module in order to debug his algorithm in the lab. The two most common interfaces for debug are UART and Ethernet. The UART interface allows the flight test engineer to easily print status and debug information to a serial console. Similarly the Ethernet interface can be used to generate debug Ethernet packets with diagnostic and status information.

Software Platform

Of course on top of the hardware platform the flight test engineer needs a software platform to abstract him from the internal workings of the hardware so he can focus on writing application code to manipulate the acquired parameters.

While the OMAP processor supports a number of operating systems, Linux was chosen as the most suitable operating system. TI maintains a freely available SDK for Linux which supports a full set of pretested components for the OMAP device. These include the kernel, boot loader, device drivers, DSP software libraries and build tools.

As Linux supports a wide set of peripherals and comes with the build tools needed to develop the applications, the flight test engineer can focus on getting their applications up and running in the minimum amount of time. Also as developers will be familiar and comfortable with the development environments, this will help in reducing the learning curve.

Developing Algorithms

The supported environment is an Ubuntu 10.04 Linux platform. The user would typically run a Ubuntu virtual machine on their PC. Inside this virtual machine they would have access to GCC which supports cross compiling to the OMAP processor. Texas Instruments also has developed and supplies, as part of the development environment, a number of utilities for leveraging the DSP. These include a number of libraries which have been written to use the DSP and a number of applications that will optimise standard C code to run on the DSP. This makes the use of the DSP relatively transparent to the flight test engineer who will not have to worry about the passing of data between the processors.

Before writing custom algorithms, the flight test engineer will decide which parameters are inputs to their algorithm and which parameters they wish to create as outputs. The flight test engineer will use the same setup software used to configure their data acquisition hardware. In a networked system all parameters within the system will be available to the flight test engineer to select from the GUI of the setup software. The engineer will simply need to select the parameters and the setup software will ensure that they get transferred in a timely and coherent manner to the module for use in the flight test engineers algorithm. Similarly the parameters the engineer creates from their algorithm will be available to any sink of data in the system.

After configuring the parameters, the setup software will create a C header file. This file defines the parameters in readable format that can be used as an input to the compiler. The underlying hardware uses this information to move the parameters into memory addresses available to the OMAP processor. This header file is then used when developing the C application.

Once the algorithm is developed, it is compiled and the resulting binary is loaded into the hardware in the same way as all other configuration settings. The flight test engineer is simply required to point to the file from the setup software GUI.

Latency and Coherency

Two of the biggest considerations when acquiring data for flight test are latency and coherency [3]. For critical parameters, it is important that the latency between sensor and screen is kept to an acceptable limit - often in the region of 250 ms.

In order to correlate measurements across the entire test article, it is vital that all channels are sampled at the same time. As well as the fact that all channels need to be sampled at the same time, it is important that the analysis software knows which sample from each channel correspond to the same sampling instant. In IRIG 106 chapter 4 PCM the analysis software can extrapolate this from the position of the parameter in the PCM frame. In Ethernet streams the analysis software can extrapolate this from the timestamp on the packet and the position of the parameter in the packet. If the analysis software can always work out which sampling instant a given sample corresponds to, the data stream is said to be coherent.

The processing of acquired data in the acquisition hardware will add latency. It is important to manage this latency addition so that it is minimized and deterministic. Minimizing the latency allows you to keep the sensor to screen delay to an acceptable limit. Ensuring that the latency is deterministic will guarantee coherency and allow the analysis software to correlate processed parameters with other processed parameters and with raw parameters.

Curtiss-Wright hardware uses the concept of an Acquisition Cycle. All channels are sampled at the beginning of an acquistion cycle and equal intervals thereafter depending on the sample rate. The processor card works on acquisition cycle boundaries. The data is acquired in one acquisition cycle, processed in the next acquisition cycle and then the processed parameters are transferred in the second next acquisition cyle. The length of the acquisition cycle is known in advance so the delay between raw parameters and processed parameters is fixed and therefore deterministic. This guarantees a coherent data stream where raw and processed parameters from all channels can be correlated. It also allows the flight test engineer to tailor their latency by keeping the acquisition cycle short.

Separating the acquistion, processing and transmission of the parameters into separate acquisition cyles also isolates the engineer who writes the algorithm on the data from the internal timings of data transfers within the hardware. The engineer does not need to know when each of their samples were acquired. The engineer has a buffer of data on which they can run their algorithm. They simply need to know that their algorithm can be carried out within the acquisition cycle. In fact if this is not the case the engineer has the option of running the algorithm over multiple acquisition cycles.

Throughput

The question of how many algorithms can be run in one acquisition cycle depends on the size of the acquisition cycle and the complexity of the algorithm. There is a 256KB buffer that is used to store raw parameters which are inputs to the algorithms and processed parameters, which are the outputs. The amount of data that can be processed in any one acquisition cycle is limited to what can be fit into this 256KB buffer.

Tab. 1: Throughput of OMAP

Sample size	Execution for FFT (16 bits) in uS
32	973
64	935
128	958
256	965
512	1025
1024	1129
2048	1273
4096	1521
8192	2062
16384	4094
32768	9747

The OMAP processor was benchmarked to see how many operations could be done in given time window. The execution time was measured for different sizes of FFT as shown in figure 1. As shown, the execution time remains at around one milisecond until the size of the FFT reaches over 8K samples at which point the time of execution starts to double.

As an example if we had a 125 mS acquisition cycle then 100 1Ksample FFTs could be executed every acquisition cycle. This could be 100 channels of analog data all processed with one processor card.

Applications

Next we consider some of the applications that are best suited to the use of custom algorithms

as written by the flight test engineer. When we think of data reduction an obvious area where telemetry bandwidth and recorder capacity savings can be made is in wideband analog.

In existing flight test campaigns, wideband signals such as vibration measurements are not typically analysed in real-time via telemetry. A data acquisition card, which samples each channel at 100Ksamples per second, will produce 1.6 Mbps of data per channel.If all this data is to be telemetered to ground then a telemetry bandwidth of, for example 10 Mbps, will be used up by a handful of vibration channels.

By doing onboard frequency analysis of these wideband acquisitions, the key information can be sent to ground using only a fraction of the bandwidth. For example an FFT can be run on the processor card and the top ten frequencies or the frequencies of interest can be telemetered to ground.

Of course the on board network has a much higher bandwidth than the telemetry link so all of the raw data could be recorded for later processing. However a large number of wideband channels this may lead to the requirement for hundreds of GigaBytes of data to be recorded. Post processing this data could be a time consuming process. The ability to process the data from each channel in the hardware and only record certain frequencies can drastically reduce the amount of recorded data and the time to extract and display that data in the ground based analysis software.

It may be that for wideband analog all the raw parameters are required but only for small time periods. For example the flight test engineer may want to record all the parameters one second either side of a certain event. In this case the processor card can be setup to store data from a certain channel for the last second. It can then trigger the transmission to the recorder of that stored data plus the next second of data from that channel based on some predefined event. This event could be an exceedance on the channel itself or the value of a parameter received on an avionics bus or the toggling of a discrete input. In this case only the raw parameters one second either side of the event are recorded, saving huge bandwidth compared to a scenario where all parameters from that channel from an entire flight need to be recorded.

Downloading and post proceesing up to 1TB of test data, can take up to 6 before before the flight test engineer is able to judge the quality of the test process and find any potential problems with the test. This long analysis time may mean that only one that only one flight test sortie can be made per day. Using some of the data reduction techniques discussed above the downloading and post processing time could be reduced to 30 minutes. This would allow multiple sorties per day which would have the knock on effect of shortening the entire flight test campaign.

The processor card could also be used to do custom bus monitoring. Taking NMEA messages as an example, a standard RS422 bus monitor will parse individual messages based on the characters in certain locations in the message. These messages are typically of variable length and comma delimited, each message potentially carrying different GPS information.

It may be desirable to extract only key parameters from the message rather than telemeter or record the entire message. Due to the variable length of the message it will not be possible to know the postion of the parameter based on character location in the message. Therefore it may not be possible to extract only the parameter of interest using a standard bus monitor. However with a processor card, a custom algorithm could be written to that uses regular expressions to pull out key information from the NMEA messages and record and telemeter only the parameters of interest.

Another possible application would be the aggregation of a particular parameter on board. For example the fuel burn from the start of the flight could be monitored. The processor card could take the fuel flow message from the ARINC 429 bus, convert it into the appropriate units and aggregate the value at each acquisition to give a fuel burn from the start of the flight. Of course the aggregator could be reset by any external event, for example the receipt of another bus message or the toggling of a discrete signal. In this way it would give the fuel burn since a particular event.

Also given that you are monitoring the total burn since a given point in time the accumulated value should be stored in non volatile memory so that if a power dip to the data acquisition hardware occurs mid flight the accumulated value is not lost.

Conclusion

Data acquisition for flight test is handled by dedicated hardware which performs specific functions and targets specific interfaces and buses. While this approach offers significant flexibility by allowing the flight test engineer to configure each data acquisition card for his particular requirements, further control and

flexibility can be offered by allowing the flight test engineer to run their own algorithms on the data acquisition hardware. Particularly as the volume of the acquired data increases, this extra control can be used to vastly reduce the amount of data to be recorded or telemetered.

This paper considered various requirements in the design of a module that would be used for on board processing. The system considerations include the requirement for the module to interface to any source and sink of data in the entire network so that any parameter destined for any transmitting device or recording media can be processed.

The hardware and software platform were considered, focusing in particular on the isolation of the flight test engineer from the internal workings of the hardware so that the engineer can focus purely on the processing algorithm. It is of equal importance to ensure a very short learning curve by using a development environment that is widely used and a processing platform that has a wide range of existing libraries and applications.

Particular emphasis was placed on the issues of latency and coherency. It is important that latency can be controlled and that it is deterministic so that correlation of processed parameters and raw parameters is achievable.

Finally, this paper discusses some of the typical applications. However one of the key functions of a card that allows flight test engineers to run their own algorithms is to provide a platform for all the atypical applications. The functions the processor card will actually be used for can be left to the imagination of all of the engineers within the flight test community.

References

[1] D. Buckley, New Paradigms for Modern Avionics Buses , in Proceedings of the International Telemetering Conference, Las Vegas, NV, 2013

[2] N. Cranley, Challenges and Solutions for Complex Gigabit FTI Networks, in Proceedings of the International Telemetering Conference, Las Vegas, NV, 2011

[3] P. Quinn, Addressing the Challenges Created by Large Ethernet Networked FTI Systems, in Proceedings of the European Telemetry Conference, Nuremburg, Germany, 2014