TELEMETRY PREPROCESSOR PERFORMANCE

Gary A. Thom
Aydin Computer and Monitor Division
700 Dresher Road
Horsham, Pa. 19044

ABSTRACT

Today there are a number of equipment vendors offering modular, bus oriented Telemetry Preprocessor systems. The architecture of these systems varies greatly as does the actual performance. This paper discusses a method for specifying and evaluating Telemetry Preprocessor performance independent of the architectural implementation.

INTRODUCTION

The Telemetry Preprocessor is used in telemetry Systems to off-load the Host Computer by performing the real-time processing tasks. Early telemetry systems consisted of a Format Synchronizer and a Host Computer. The Format Synchronizer would synchronize to the incoming data stream, perform serial to parallel conversion to a fixed word length (usually 8 bits) and transfer the data thru a DMA device to the Host Computer. If the data rate was slow enough, the Host Computer could do some processing in real-time. If not, the Host Computer would store the data for later, non-real-time processing and analysis. As digital logic technology advanced, so did the amount of processing done external to the Host Computer. First Came Decommutation and Time Tagging. Next was Data Compression and Engineering Units Conversion. Today, multi-stream, real-time Telemetry Preprocessors perform all of these functions as well as Derived Parameter Processing, Quick took Graphics Displays, Data Archiving, and Data Analysis.

BASIC ARCHITECTURE

All Telemetry Preprocessors are not created equal. They do not use the same architecture and may perform differently in different applications. The early Telemetry Preprocessor
systems were made up of a number of serially connected boxes. Each box performed a particular function such as Decommutation, Data Compression, or Engineering Units Conversion. System performance was limited by the slowest box in the system.

Today, most Telemetry Preprocessors are based on some bus structure onto which functional modules are plugged. These systems usually implement a Data Driven Architecture which has proven to be an excellent environment for processing telemetry data and has significant advantages over traditional multiprocessor systems. These advantages have been widely documented in telemetry and computer science literature and will not be repeated in this paper.

The Data Driven Architecture allows multiple inputs to feed multiple processors both of which can feed multiple outputs. At the most basic level, a Telemetry Preprocessor consists of a Bus System, Input Ports, Processors, and Output Ports. The manner in which each of these items is implemented has a significant impact on system performance and life cycle cost. In addition, these parallel or distributed architectures make evaluating system performance and capability much more difficult.

**SYSTEM PERFORMANCE**

The Telemetry Preprocessor receives data, processes it, and outputs it to an external device. The performance of the Telemetry Preprocessor is not the rate at which it does any one of these tasks, but it is the combined rate at which it does all of them for a given application. The key performance measurement is Application Throughput in Parameters Per Second (P/S). Application Throughput is the measure of system throughput for a given application. This is related to the concept of computer benchmarks, where a benchmark program having an instruction mix approximating the final application program is run to determine the computers performance for the application. Similarly, a benchmark for the Telemetry Preprocessor needs to be used to get a valid measure of system throughput for the desired application.

The information needed to determine the Application Throughput varies from program to program, but essentially,
it is a model of the final application. The Application Model is a throughput model and not an implementation description and care must be taken to keep the two separate. Similarly, the Application Model is not the entire functional requirements, but is extracted from the functional requirements to produce a concurrent operation requirement. To develop this model, three areas need to be examined. These are the system inputs, parameter processing, and system outputs which occur simultaneously in the desired application.

System input requirements consist of the following:

- Number and type of input ports.
- Input parameter rate of each input port.
- Input parameter number system mix.
- Total number of unique input parameters.

Processing requirements consist of the following:

- EU Conversion algorithm mix.
- Data Compression algorithm mix.
- Derived parameter algorithm mix.
- Other processing requirements.
- Data precision.

System output requirements consist of the following:

- Number and type of output ports.
- Output parameter rate of each output port.
- Output parameter number system mix.

Evaluation of the system input requirements provides an aggregate input parameter rate in Parameters per Second, an number system conversion mix, and the number of input ID tags which are needed. The input parameter rate is then applied to the various processing mix requirements to determine how much processing power is required in the preprocessor. The number system mix is necessary to properly evaluate the processing requirements since conversion to a common number system may be required prior to any processing.

The number of unique input parameters along with the various processing mix requirements is used to determine the total
number of ID tags required in the system. This is an important evaluation because some preprocessors may be limited by available ID tags because many processes require new ID tags to be assigned. This includes preprocessors which can not chain algorithms or distribute a parameter to multiple algorithms without assigning a new ID tag, or which must ID tag partial results being passed to another processor.

The first processing requirement is to convert the input data to a common internal number system (ie. 2'S Comp., IEEE Floating Point, Etc.). The input number system mix is used to size this requirement. The input number system mix is specified as a group of numbers which are the percentage of the aggregate input which is received in a particular number system. The total of these numbers should equal 100 percent. For example:

- 5% 32 Bit IEEE Floating Point
- 5% Mil-Std-1750 Floating Point
- 10% Discretes (no conversion required)
- 10% BCD
- 20% Offset Binary
- 20% Signed Magnitude
- 30% Two’s Complement

--------

100%

The remaining processing requirements should specify the mix of algorithms to be performed on the aggregate input parameter rate. This mix must include all of the processing expected to be performed on the data for the application. This includes Engineering Units Conversion, Derived Parameter Processing, Data Compression, and other application specific processing. The processing mix is also specified as a group of numbers which are the percentage of the aggregate input which are to be processed by a particular algorithm. Note that this is not the percentage of processed data but that it is the percentage of the aggregate input data since the processed data rate has not been determined. It is possible for the total of these numbers to exceed 100 percent. This is because multiple algorithms or processes may be applied to the same parameter. For example:
Engineering Units Conversion
  10% Table Lookup (32 point average)
  10% 5th Order Polynomial
  75% 1st Order Polynomial
Data Compression
  10% Pass In Limits
  10% Delta Slope
  10% Pass Bit Change
Derived Processing
  5% \((2X+3Y^2)\)
  1% \((\cos(x))\)
Other Processing
  1% Average (20 samples)
  5% Syllable Concentration
-----
137%

By applying these percentages to the aggregate input parameter rate, the parameter rate requirement for each type of algorithm can be determined.

The final processing requirement is to convert the common internal number system to the required output number system. The output number system mix and aggregate output parameter rate is used to size this requirement. The output number system mix is specified in the same manner as the input number system mix. This mix might typically be as follows:

  10% Offset Binary
  10% Discretes
  20% Two’s Complement
  60% DEC Floating Point
-----
100%

Evaluation of the system output requirements results in an individual output parameter rate for each output port. Output ports are functional ports and not necessarily physical ports. The physical output ports are highly dependent on the Telemetry Preprocessor architecture and the division of work between the Host Computer and the Telemetry Preprocessor. The aggregate output parameter rate can be greater than the aggregate input parameter rate because a single parameter may go to more than one output. Similarly,
both the raw (unprocessed) and processed data can be sent to the outputs. It is also possible for the aggregate output parameter rate to be less than the aggregate input parameter rate because of expected data compression or a limited output requirement (i.e., display only).

**PARAMETERS PER SECOND**

The most important concept in developing the Application Model is dealing with Parameters per Second as a unit of measure. Many Telemetry Preprocessor specifications as well as system requirements use measures which are meaningless. For instance, what is the true performance of a Telemetry Preprocessor which is specified as having a Bus System rate of 760 Million Bytes per Second? Bytes per Second is a meaningless concept in a Telemetry Preprocessor. This is because parameters vary in size from one bit to 32 or more bits. A transfer of data in the system is generally the same size regardless of the parameter size. For example, a 32 bit data bus will transfer 1 bit words or 32 bit words in a single transfer at the same rate. In this case, the parameter rate is the bus transfer rate. However, when the data bus size is less than the parameter size, multiple transfers need to be made for one parameter. Similarly, if a broadcast transfer mode is not used, multiple transfers are required to distribute a parameter to all of the modules which require it.

These factors tend to further reduce the actual performance of the preprocessor. So the proper measure of the Bus System speed is not Bytes per Second, Transfers per Second, or Hertz, but it is Parameters per Second.

This same concept also applies to the processors. What is the true performance of a Telemetry Preprocessor having a processing rate of 10 Million Instructions per Second (MIPS) or 33 Million Floating Point Operations per Second (MFLOPS). These measures are useful for relative comparisons of general purpose computers but are meaningless for Telemetry Preprocessors. The architecture of the processing elements will significantly influence their true performance. A commercial microprocessor board may have an impressive MIPS/MFLOPS statistic but be a poor performer in the Telemetry Preprocessor because a significant portion of the processors capacity is used up moving data into and out of
the board. A custom processing element with less impressive MIPS/MFLOPS statistics can easily out perform the commercial board. This is because optimized I/O logic and algorithm vectoring eliminate much of the processing overhead so that most of the processors capacity can be used for actual data processing.

The performance of each real-time algorithm identified in the processing mix must be determined. This will give a Parameters per Second measure for each algorithm which can be compared to the required parameter rate established for the Application Model.

**THE APPLICATION MODEL**

Now that the information has been gathered, the Application Model can be generated. Figure 1 shows a simple Application Model. Multiple inputs consisting of PCM at 500K Parameters/Second, PAM at 250K Parameters/Second, and ADC at 100K Parameters/Second are combined into an aggregate input rate of 850K Parameters/Second. An evaluation of the input data types yields the associated input number system mix. The processing mix is also determined by evaluating the individual requirements and producing an aggregate mix. The processing rate for each individual algorithm can be determined by applying the process mix to the aggregate parameter rate. This is shown below in parameters per second.

**Engineering Units Conversion**

- 85K Table Lookup (32 point average)
- 85K 5th Order Polynomial
- 637.5K 1st Order Polynomial

**Data Compression**

- 85K Pass In Limits
- 85K Delta Slope
- 85K Pass Bit Change

**Derived Processing**

- 42.5K \((2X+3y^2)\)
- 8.5K \((\cos(x))\)

**Other Processing**

- 8.5K Average (20 samples)
- 42.5K Syllable Concatenation
This indicates a total of 1164.5K operations per second. However, a preprocessor which can support 1164.5K operations per second may not be able to support it for the given process mix. The individual numbers must be compared with telemetry preprocessor specifications to determine if it can handle the individual as well as the total processing requirement.

Although there are multiple inputs and outputs, it is basically a single path model in which performance and mixes are based on aggregate rates.

More accurate models are possible at the expense of simplicity. Figure 2 shows a more complex version of the Application Model. This allows the Application Model to more accurately reflect the processing requirements and number systems mixes as they relate to parameter rates and functional data paths.

In more complex systems or systems supporting multiple applications, more than one application model can be developed. This is especially useful if the requirements are mutually exclusive. Specifying them on the same Application Model would be misleading. However, using multiple models would identify exactly which requirements had to be concurrently satisfied.

Notice that the processing rate of 1164.5K parameters per second is significantly larger than the 850K parameter per second aggregate input rate. Specifying an 80% 1st Order Polynomial Conversion, 20% 5th Order Polynomial Conversion mix based on the aggregate input parameter rate would result in a system which is under sized for the actual requirement.

**PERFORMANCE EVALUATION**

A candidate Telemetry Preprocessor must be able to support the Application Throughput as specified by the Application Model. This requires an evaluation of the preprocessors architecture and actual performance. The preprocessor must be able to support the input parameter rates, number system conversions, processing rates, and output parameter rates. By comparing the actual performance of each algorithm in the processing mix to the specified performance from the
application model, a determination can be made as to whether the preprocessor can support the processing requirements.

Any input or output number system conversions must also be accounted for. This must be done in evaluating the various available Telemetry Preprocessors and cannot be specified without implying some implementation. For example, if a preprocessor's internal format is IEEE Floating Point, then all input number systems must be converted to that format before any processing can take place. The preprocessor's actual number system conversion performance must be weighted by the input number system mix and input aggregate parameter rate. This is then added to the total processing requirements and increases the processing rate requirement from the Application Model. The output number system mix must be evaluated in a similar manner.

CONCLUSION

The Application Model gives a clear picture of the Telemetry Preprocessor performance requirements. This allows vendors to have a better understanding of the requirements and goals of the system. A better understanding of the requirements will reduce false starts and other misunderstandings that can occur in evaluating both system requirements and equipment specifications. The other benefit is that the Application Model is implementation independent so that the resulting Telemetry Preprocessor configuration is not burdened by non-essential requirements due to a specific vendor's hardware implementation.

Another benefit of the Application Model is that it can be used as the basis for acceptance of the Telemetry Preprocessor for the specified application. It proves system performance and assures compatibility with the intended application. A Telemetry Preprocessor may meet the individual requirements but may not be able to meet the combined requirements for the actual application. The Application Model, however, cannot be used as the only measure of compliance with the requirements. Each individual requirement has its own minimum and maximum specifications which must be evaluated. But these can be evaluated independent of the other specifications.
A TYPICAL TELEMETRY PREPROCESSOR APPLICATION MODEL

FIGURE 1.

MULTIPLE PATH APPLICATION MODEL

FIGURE 2.