ClimateTalk 2.0

Application Note: CT-485 AutoNet Pseudo-Random Number and Timing

Document revision: 01
Release: June 12, 2013

2400 Camino Ramon
Suite 375
San Ramon, CA 94583 USA

Updates

This application note may be updated at any time and may be superseded by a more recent version or amended to from time to time. Users should be certain they are using the current ClimateTalk version and the latest revision of the documents.

The released versions of all specifications are available at http://www.ClimateTalk.org

Revision History

ClimateTalk Version	Document Revision	Release Date	Comments
V 1.0	V01 R01	2011-04-12	Initial draft
	V01 DR02	2011-09-01	General edits and cleanup
V 2.0	00	2013-01-18	Version 2.0 Release – Increment version, no updates.
V 2.0	01	2013-06-12	No changes. Revision increment for package release.

Contributors

The following is a list of ClimateTalk Alliance member companies that were actively engaged in the development of this standard:

A.O. Smith Company

EDC

EWC Controls

Emerson Electric, Co.

Microchip Technologies, Inc.

Nogginhaus, LLC.

Research Products Corp.

Rheem Manufacturing Company

Zonefirst

Table of Contents

List of Figures

1.0 Overview

1.1 ClimateTalk Model

ClimateTalk is an open standard that defines a set of messages and commands to enable interoperability, enhanced user interface, and machine to machine control independent of the physical layer connecting the devices.

The messages and commands defined by ClimateTalk Information Model (CIM) are the presentation and application layers as defined by the OSI Model[1]. ClimateTalk Applications are fully defined at Layer 7 of the OSI model by a combination of a Device Specific Application Profile, the Generic Application Specification and the Command Reference.

ClimateTalk messages can be carried over any physical medium following the OSI model. The ClimateTalk Presentation Layer defines how messages are executed over the various physical mediums in use.

CT-485 and CT-LWP are wired serial physical and network layers designed to support the formation of ClimateTalk networks and transport ClimateTalk messages, but other OSI based protocols – including wireless transports - can be used as well.

1.2 Scope

CT-485 is a Physical, Data Link, and Networking set of specifications that define one of the physical media over which ClimateTalk messages are sent. CT-485 is a variant of EIA/TIA-485[2] standards with provisions against incorrect wiring and grounding requirements that meet the needs of residential systems.

The CT-485 AutoNet pseudorandom number and timing application note provides supporting information on generating pseudorandom numbers to support implementation of CT-485 networking.

This document provides some helpful guidelines for generating pseudorandom numbers and timing in support of a Network Layer implementation of ClimateTalk over a TIA-485[3]-based serial wired network, referred to as CT-485. The information in this document is intended as a guideline for a developer implementing the networking requirements for CT-485.

[1] http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=20269
[2] http://standardsdocuments.tiaonline.org/tia-tsb-89-a.htm
[3] Telecommunications Industry Association (TIA) publications are available from IHS, http://global.ihs.com. This standard was known previously as RS-485 and EIA-485 and is sometimes referred to by all three sets of initials as TIA/EIA -485.

2.0 Normative References

A good understanding of the following documents is required to apply the contents of this specification correctly.

ClimateTalk Generic Application Specification

ClimateTalk Command Reference

ClimateTalk CT-485 Application Protocol Interface

ClimateTalk CT-485 Networking Specification

ClimateTalk CT-485 Data Link Specification

ClimateTalk CT-485 Physical Specification

3.0 Terminology

3.1 Definitions

Pseudo-Random Number A set of values or elements that is statistically random, but is derived from a known starting point and is typically repeated over and over.

3.2 Acronyms

PRN Pseudo-Random Number

3.3 Word Usage

The conventions used in this document are modelled after the definitions of the *2009 IEEE Standards Style Manual*. The *IEEE Standards Style Manual* can be downloaded from https://development.standards.ieee.org/myproject/Public/mytools/draft/styleman.pdf.

can Equivalent to *is able to* or *is capable of*.

may Equivalent to *is permitted to* or *is allowed to*. The use of *may* means that something is optional and does not imply a requirement.

must Used to describe situations where no other course of action is possible.

shall Equivalent to *is required to*. Use of the word *shall* means that the specification shall be implemented exactly as described in order to ensure correct operation and interoperability with other devices.

should Equivalent to *is recommended that*. This is used in situations where there are several possible options, but one option is preferable to the others.

4.0 Pseudo-Random Numbers and Timing

4.1 Slot Delay Calculation Theory

The AutoNet addressing method used in CT-485 is predicated on the Slot Delay calculated by each device's Slot Delay being different enough every time to allow one device with a longer Slot Delay to 'see' the other device's transmission (shorter Slot Delay) on the network and hence aborting its own transmission.

Slot Delay is specified as a time between 100 ms and 2.5 seconds. This timing can be calculated in many ways, but should be done in such a way as to provide a pseudo-random value with more than 11 bits of resolution. A precision and accuracy of one millisecond is sufficient for this requirement and therefore is the recommended method of calculating Slot Delay. The value for Slot Delay should be stored in a 16-bit unsigned integer that contains a value no greater than 2500 (in ms). Simply limiting the value is acceptable to ensure that the value is within range.

Every network node contains subtle differences of physical components and other real-world differences that theoretically make it possible to differentiate between physical PCB assemblies within the network. Each network node executes its firmware on a physical microcontroller that has variances based on, but not limited to, the following parameters:

1. Silicon IC timing variances within the IC (i.e. Watchdog Timer, A/D, etc.)
2. Silicon variances in EEPROM write times per part and temperature range
3. Input Voltage levels vary per component and temperature vs. slew rate
4. Production variances per part number, lot number, date code, and individual die physics

In addition to the microcontroller itself, each Printed Circuit Board (PCB) assembly performs with slightly different parameters based on the following variances:

5. Voltage regulator Vout tolerances based on variances in the voltage regulator IC, silicon diodes, inductors, and capacitors
6. Vout tolerances based on current consumption at specific moments in time in addition to particular physical locations of a circuit and the current state of the circuit
7. Communication slew rates and voltage levels (i.e. capacitor and resistor values). Small oscillator frequency tolerances affect CPU speed and baud rate drifts within the overall system tolerance.
8. Physical characteristics (i.e. temperature, wire type/lengths, PCB traces, etc.)
9. Each Node Type provides different boot up and system timings based on the SCT interface device's timings
10. Network latency timings differences can be very different from node to node based on wire lengths and use of other mediums, modulation techniques, routers, or network bridges (i.e. iPLC, RF, IR, etc.)
11. Localized real world sensors can yield distinctive data readings that are unique to a specific Node Type's characteristics. Reading of a temperature sensor, a localized motor's RPM, or a dedicated resistor/capacitor charge timing can yield exceptional results in providing unique finger-printing of a specific node within the network.

NOTE 1: Many single node systems will exploit the technique of sampling the AC power supply as a method of determining a random number seed. This works quite well for single node systems. However, due to the nature of a network containing many nodes, power phase angle data is NOT recommended for seeding a pseudo-random number system in a CT-485 network.

4.2 Localized System Timing Theory

Command execution localized within a node primarily varies based on the timing realities outlined below.

4.2.1 Absolute Communication Timings

Absolute communication timings are timings that are deterministic and localized to a specific node's perspective. Whenever an AutoNet Client is idle and waiting for a command from the subnet's AutoNet Server, there are certain timings that are absolute and predictable. Timings for each particular node can be forecasted reliably and assumed consistent from session to session based on its Node Type and attributes.

4.2.2 Relative Communication Timings

Relative communication timings are timings that are more difficult to forecast and are not consistent between nodes from the AutoNet Server's perspective. The concept of relativity from one node's perspective to another is critical for dynamic addressing to be successful. Some of the causes of relative timings within a network system are based on the following real-world attributes:

- Speed of the signal is based primarily on the speed of light = C[4]
- Relative observation of one signal by multiple nodes will yield different perspectives of packet timings from an absolute node's reference basis
- Latency of wire-line networks yields approximately 11.8 nanoseconds per foot @ C
- Latency of other mediums like iPLC or wireless can be a few milliseconds to hundreds of milliseconds.

  NOTE 2: CT-485 currently allows for up to 2.5 seconds latency.

- Relative timings between multiple discrete nodes (from power up) yield different alignments between identical main loops and interrupt driver ISR routines in firmware

To achieve a reliable Slot Delay calculation, as many of the known inconsistencies within the system as possible need to be exploited. The current implementation of Slot Delay calculations attempts to take into account each of the above variances within the system.

[4] http://en.wikipedia.org/wiki/Speed_of_light

Slot Delay is calculated in such a way as to yield an integer value that appears to be random based on subtle differences in system timings. In other words, a minor variance in system clock cycles could yield variances of only a few hundred nanoseconds. However, these minor differences are important! This is key because any subtle differences from node to node, no matter how insignificant, can be used to calculate a value of Slot Delay that can vary over the complete AutoNet timing spectrum (100 ms to 2.5 seconds).

The randomize method should be loaded with a seed value that incorporates bit-encoded data from the following perspectives:

- The number of foreground loops since power up (rolling value over time), which provides a relative timing perspective based on received packet timings
- MAC Address, which forces a physical variance between an otherwise identical nodes' calculation
- Node Type, which is a known value for each AutoNet Client node
- CmdSeed, which is a concept of physical PCB differences that can be seen in EEPROM writing timings

It is important to ensure that the Slot Delay Time is not simply a compilation of the above information. The above information should be represented in the creation of the Randomization Seed and then the random algorithm selected should create an 11-bit value based on the derived seed. The 11-bit value can be generated as follows:

> SEED = [MacAdd[2]:NodeType:CmdSeed:ForeGroundLoopCnt], which yields a random number between 0 and 1.
>
> Slot Delay = ((RANDOM{SEED} x 65535) & 2047) + 100, which yields a randim number between 100 and 2147

The most important concept for each implementation is to test the physical system of your Slot Delay calculation and sample approximately 100 Slot Delay values (on the same PCB) while running in a real system. The Slot Delay Time should be distributed throughout the range of 0 to 2500. The goal is to not repeat the same value very often and more importantly, to see successive calculated values yielding entirely different values distributed throughout the AutoNet Slot Delay Time spectrum of 0 to 2500 milliseconds. If the distribution of the Slot Delay values does not distribute pseudo-randomly across the spectrum, then the algorithm should be reviewed and revised.

The final test is to attach several PCB nodes that are identical in all aspects except MAC Address and Node Type. The next step is to allow the AutoNet Server to address all of the AutoNet Client nodes in the test setup. The Slot Delay for each AutoNet Client node should be noted and this test should be run repeatedly to verify that the individual nodes' Slot Delay Times do not overlap regularly. Primarily, the goal of the algorithm is to not have any overlap in Slot Delay values per Node Type for more than three sequential command sequences in a row.

4.3 Pseudo-Random Number (PRN) Generator Primer

The AutoNet Addressing Mechanism requires pseudo-random numbers with specified ranges to be generated for both Slot Delay and Session ID, and possibly even MAC ID for some devices. This section describes an implementation of a linear feedback shift register that facilitates the required pseudo-random number generation for these purposes.

The concept of generating a Pseudo-Random Number (PRN) is meant to provide a sequence of numbers that appear to be random. The "pseudo" indicates that the random nature is not real at all. If an observer watches the pattern, eventually it will repeat. A common method for generating a sequence of numbers that appear to be random is called the 'Linear Feedback Shift Register' (LFSR) Method. A 16-bit LFSR yields 65,536 different patterns before repeating.

The initial value of the LFSR is called the Seed. Because the operation of the register is deterministic, the sequence of values produced by the register is determined completely by its current (or previous) state. Likewise, because the register has a finite number of possible states, it eventually must enter a repeating cycle. However, a LFSR with a well-chosen feedback function can produce a sequence of bits, which appears random and which has a very long cycle.

One implementation of the LFSR is the Galois LFSR. A LFSR in Galois configuration is a popular structure that can generate pseudo-random number sequences using a mathematical LFSR technique. When the system is clocked in the Galois configuration, bits that are not taps are shifted. The taps are XOR'd with the new output, which also becomes the new input. To generate the same sequence, the order of the taps is the reverse of the order for the conventional LFSR.

Galois LFSRs do not concatenate every tap to produce the new input. The XOR'ing is done within the LFSR and no XORs are run in serial. Therefore, the propagation times are reduced to that of one XOR rather than a whole chain. It then is possible for each tap to be computed in parallel, increasing the speed of execution.

In a software implementation of an LFSR, the Galois form is more efficient as the XOR operations can be implemented a word at a time. Only the output bit must be examined individually.

Figure 1 – Galois LFSR

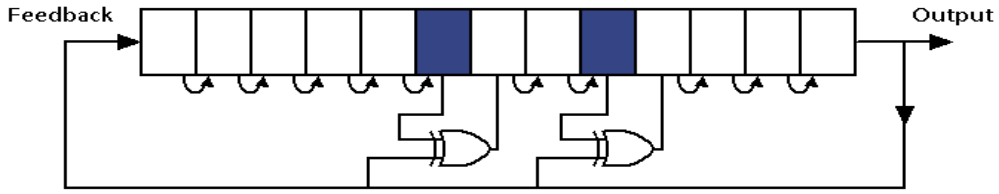

Figure 1 – Galois LFSR

shows an example of Galois LFSR Implementation for Pseudo-Random Number Generation.

5.0 Annex A – Bibliography

"TIA-485 (Revision A), Electrical Characteristics of Generators and Receivers for Use in Balanced Digital Multipoint Systems" *Telecommunications Industry Association*, 1998.

Zimmermann, Hubert (April 1980). "OSI Reference Model — The ISO Model of Architecture for Open Systems Interconnection". *IEEE Transactions on Communications*

Michael Luby, Pseudo-Randomness and Cryptographic Applications, Princeton Univ. Press, 1996. A definitive source of techniques for provably random sequences.

J. Viega, Practical Random Number Generation in Software, in Proc. 19th Annual Computer Security Applications Conference, Dec. 2003.