P. 56

NASA Technical Memorandum 107634

AN EXTENSION TO SCHNEIDER'S GENERAL PARADIGM FOR FAULT-TOLERANT CLOCK SYNCHRONIZATION

Paul S. Miner

June 1992

NASA

National Aeronautics and Space Administration

Langley Research Center
Hampton, VA 23665

Contents

1 Introduction

In 1987, Schneider presented a general paradigm that provides a single proof of a number of fault-tolerant clock synchronization algorithms [1]. His proof was subsequently subjected to the rigor of mechanical verification by Shankar [2]. However, both Schneider and Shankar assumed a condition Shankar refers to as *bounded delay*. This condition states that the elapsed time between synchronization events (i.e. the time that the local process applies an adjustment to its logical clock) is bounded. This property is really a result of the algorithm and should not be assumed in a proof of correctness. The purpose of this paper is to remedy this by providing a general proof of this property in the context of the general paradigm proposed by Schneider. The argument given here is based on the proof of this property for the algorithm of Welch and Lynch [3, Section 6]. The notation used is from [2] except where noted.

2 Clock Definitions

Any implementation that satisfies the definitions and constraints in Shankar's report will provide the following guarantee [2].

Theorem 1 (bounded skew) *For any two clocks p and q that are nonfaulty at time t,*

$$|VC_p(t) - VC_q(t)| \leq \delta$$

That is, the difference in time observed by two non-faulty clocks is bounded by a small amount. This gives the leverage needed to reliably build a fault-tolerant system. This section presents the definitions and conditions to be met to guarantee this result. Much of it is taken from sections 2.1 and 2.2 of Shankar's report documenting his mechanization of Schneider's proof [2]. Modifications to the conditions needed for this revision of the theory are also presented .

2.1 Notation

A fault-tolerant clock synchronization system is composed of an interconnected collection of physically isolated clocks. Each redundant clock will incorporate a physical oscillator which marks passage of time. Each oscillator will drift with respect to real time by a small amount. Physical clocks derived from these oscillators will similarly drift with respect to each other. There are two different views of physical clocks relating different perceptions of time. Real time will be denoted by lower case letters, e.g. t, s: **Var time**. Typically, **time** is taken as ranging over the real numbers. Clock time will be represented by upper case letters, e.g. T, S: **Var Clocktime**. While **Clocktime** is often treated as ranging over the reals [3, 2, 4], a physical realization of a clock marks time in discrete intervals. It is more appropriate to treat values of type **Clocktime** as representing some integral number of ticks. There are two sets of functions associated with the physical clocks[1]: functions mapping real time to clock time for each process p,

$$PC_p : \mathsf{time} \rightarrow \mathsf{Clocktime};$$

and functions mapping clock time to real time,

$$pc_p : \mathsf{Clocktime} \rightarrow \mathsf{time}.$$

The intended semantics are for $PC_p(t)$ to represent the reading of p's clock at real time t, and for $pc_p(T)$ to denote the earliest real time that p's clock reads T. By definition, $PC_p(pc_p(T)) = T$, for all T. We assume nothing about the relationship of $pc_p(PC_p(t))$ to t.

The purpose of a clock synchronization algorithm is to make periodic adjustments to local (virtual) clocks to keep redundant clocks within a bounded skew of each other. This periodic adjustment makes analysis difficult, so an interval clock abstraction is used in the proofs. Each process p will have an infinite number of interval clocks associated with it, each of these will be indexed by the number of intervals since the beginning of the protocol. An interval corresponds to the elapsed time between adjustments to the virtual clock. These interval clocks are equivalent to a process' physical clock plus an offset. As with the physical clocks, they are characterized by two functions: $IC_p^i : \mathsf{time} \rightarrow \mathsf{Clocktime}$; and $ic_p^i : \mathsf{Clocktime} \rightarrow \mathsf{time}$. If we let $adj_p^i : \mathsf{Clocktime}$ denote the cumulative adjustment made to a clock as of the ith interval, we get the following definitions for the ith interval clock:

$$\begin{aligned} IC_p^i(t) &= PC_p(t) + adj_p^i \\ ic_p^i(T) &= pc_p(T - adj_p^i). \end{aligned}$$

From these definitions it is simple to show $IC_p^i(ic_p^i(T)) = PC_p(pc_p(T - adj_p^i)) + adj_p^i = T$, for all T. Sometimes it is more useful to refer to the incremental adjustment made in a particular interval than to use a cumulative adjustment. By letting $ADJ_p^i = adj_p^{i+1} - adj_p^i$ we get the following equations relating successive interval clocks:

$$\begin{aligned} IC_p^{i+1}(t) &= IC_p^i(t) + ADJ_p^i \\ ic_p^{i+1}(T) &= ic_p^i(T - ADJ_p^i). \end{aligned}$$

[1]Shankar's presentation includes only the mappings from time to Clocktime. The mappings from Clocktime to time are added here, because it is a more natural representation for some of the proofs.

A virtual clock, VC_p : time $\rightarrow$ Clocktime, is defined in terms of the interval clocks by the equation

$$VC_p(t) = IC_p^i(t), \text{ for } t_p^i \leq t < t_p^{i+1}.$$

The symbol t_p^i denotes the instant in real time that process p begins the ith interval clock. Notice that there is no mapping from Clocktime to time for the virtual clock. This is because VC_p is not necessarily monotonic; the inverse relation might not be a function for some synchronization protocols.

Synchronization protocols provide a mechanism for processes to read each others clocks. The adjustment is computed as a function of these readings. In Shankar's presentation, the readings of remote clocks are captured in function Θ_p^{i+1} : process $\rightarrow$ Clocktime, where $\Theta_p^{i+1}(q)$ denotes process p's estimate of q's ith interval clock at real time t_p^{i+1} (i.e. $IC_q^i(t_p^{i+1})$). Each process executes the same (higher-order) convergence function, cfn : (process, (process $\rightarrow$ Clocktime)) $\rightarrow$ Clocktime, to determine the proper correction to apply. Shankar defines the cumulative adjustment in terms of the convergence function as follows:

$$\begin{aligned} adj_p^{i+1} &= cfn(p, \Theta_p^{i+1}) - PC_p(t_p^{i+1}) \\ adj_p^0 &= 0. \end{aligned}$$

The following can be simply derived from the preceding definitions:

$$\begin{aligned} VC_p(t_p^{i+1}) &= IC_p^{i+1}(t_p^{i+1}) = cfn(p, \Theta_p^{i+1}) \\ IC_p^{i+1}(t) &= cfn(p, \Theta_p^{i+1}) + PC_p(t) - PC_p(t_p^{i+1}) \\ ADJ_p^i &= cfn(p, \Theta_p^{i+1}) - IC_p^i(t_p^{i+1}). \end{aligned}$$

Using some of these equations and the conditions presented in the next section, Shankar mechanically verified Schneider's paradigm. This paper presents a general argument for satisfying one of the assumptions of Shankar's proof. The argument requires some modifications to Shankar's constraints, and introduces a few new assumptions. In addition, some of the existing constraints are rendered unnecessary.

A new constant, R : Clocktime, is introduced which denotes the expected duration of a synchronization interval as measured by clock time (i.e. in the absence of drift and jitter, no correction is necessary for the clocks to remain synchronized. In this case the duration of an interval is exactly R ticks). We also introduce a collection of distinguished clock times S^i : Clocktime, such that $S^i = iR + S^0$ and S^0 is a particular clock time in the first synchronization interval. We also introduce the abbreviation s_p^i defined to equal $ic_p^i(S^i)$. The only constraints on S^i are that for each nonfaulty clock p, and real times t_1 and t_2,

$$(VC_p(t_1) = S^i) \wedge (VC_p(t_2) = S^i) \supset t_1 = t_2,$$

and there exists some real time t, such that

$$VC_p(t) = S^i.$$

The rationale for these constraints is that we want to unambiguously define a clock time in each synchronization interval to simplify the arguments necessary to bound separation of good clocks. If we choose a clock time near the instant that an adjustment is applied, it is possible that the VC will never read that value (because the clock has been adjusted ahead), or that the value will be reached twice (due to the clock being adjusted back). In [3], the chosen unambiguous event is the clock time that each good processor uses to initiate the exchange of clock values. For other algorithms, any clock time sufficiently removed from the time of the adjustment will suffice. A simple way to satisfy these constraints is to ensure for all i, $S^i + ADJ_p^i < T_p^{i+1} < S^{i+1} - ADJ_p^i$, where $T_p^{i+1} = IC_p^i(t_p^{i+1})$.

$PC_p(t)$	The reading of p's physical clock at real time t.
$pc_p(T)$	The earliest real time that p's physical clock reads T.
$VC_p(t)$	The reading of p's virtual clock at time t. This is the logical time used by the system.
$IC_p^i(t)$	The reading of p's ith interval clock at real time t
$ic_p^i(T)$	The earliest real time that p's ith interval clock reads T.
t_p^i	The real time that processor p begins the ith synchronization interval.
adj_p^i	Cumulative adjustment to p's physical clock up to and including t_p^i.
ADJ_p^i	$adj_p^{i+1} - adj_p^i$
Θ_p^{i+1}	An array of clock readings (local to p) such that $\Theta_p^i(q)$ is p's reading of q's ith interval clock at t_p^i.
$cfn(p, \Theta_p^{i+1})$	Convergence function executed by p to establish correct $VC_p(t_p^{i+1})$.

Table 1: Clock Notation

Table 1 summarizes the notation for the key elements required for a verified clock synchronization algorithm.

2.2 The Conditions

This section introduces the conditions required by Shankar's mechanical proof of Schneider's Theory. The changes needed for the general extension to the theory are also introduced here. The first condition defines initial skew, δ_S, which is a bound on the difference between good clocks at the beginning of the protocol.

> **Old Condition 1 (initial skew)** *For nonfaulty processors p and q*
>
> $$|PC_p(0) - PC_q(0)| \leq \delta_S$$

This condition will be replaced by the following;

> **New Condition 1 (bounded delay init)** *For nonfaulty processes p and q*
>
> $$|s_p^0 - s_q^0| \leq \beta'$$

a constraint similar to the original condition can be easily derived from this new condition using the constraint on clock drift. Given suitable constraints on the convergence function, it will be shown that for nonfaulty processes p and q, and all i,

$$|s_p^i - s_q^i| = |ic_p^i(S^i) - ic_q^i(S^i)| \leq \beta'.$$

That is, β' will be shown to bound the separation of clocks at a particular Clocktime in each interval.

The rate at which a good clock can drift from real-time is bounded by a small constant ρ.

> **Old Condition 2 (bounded drift)** *There is a nonnegative constant ρ such that if clock p is nonfaulty at time $s, s \geq t$, then*
>
> $$(1-\rho)(s-t) \leq PC_p(s) - PC_p(t) \leq (1+\rho)(s-t)$$

This characterization of drift is not quite accurate, and is only valid if Clocktime ranges over the rationals or reals. If we treat Clocktime as an integer, the inequality does not hold for all s, t, or ρ. We restate the condition for the mapping from Clocktime to time. To allow for future modifications to the theory which allow for recovery from transient faults, we also remove the implicit assumption that non-faulty clocks have been so since the beginning of the protocol.

New Condition 2 (bounded drift) *There is a nonnegative constant ρ such that if p's clock is nonfaulty during the interval from T to $S, (S \geq T)$, then*

$$(S - T)/(1 + \rho) \leq pc_p(S) - pc_p(T) \leq (1 + \rho)(S - T)$$

The benefit of changing the lower bound to $(S - T)/(1 + \rho)$ is that we can derive the following constraint on the mapping from time to Clocktime:

$$(pc_p(S) - pc_p(T))/(1 + \rho) \leq PC_p(pc_p(S)) - PC_p(pc_p(T)) \leq (1 + \rho)(pc_p(S) - pc_p(T))$$

This is not as strong an assumption as Shankar's original condition. However, if the unit of time is taken to be a tick of Clocktime and Clocktime ranges over the integers, we can then derive the following bound on drift that is sufficient for preserving Shankar's mechanical proof (with minor modifications):

$$\lfloor (s - t)/(1 + \rho) \rfloor \leq PC_p(s) - PC_p(t) \leq \lceil (1 + \rho)(s - t) \rceil .$$

Note that using Shankar's algebraic relations defining various components of clocks, we can use these constraints to bound the drift of any interval clock (ic_p^i) for any i.

The following corollary to *bounded drift* limits the amount two good clocks can drift with respect to each other during the interval from T to S.

$$|pc_p(S) - pc_q(S)| \leq |pc_p(T) - pc_q(T)| + 2\rho(S - T)$$

Shankar stated the above corollary with respect to the original formulation of bounded drift.

We can also derive an additional corollary (this adapted from lemma 2 of [3]).

$$|(pc_p(S) - S) - (pc_p(T) - T)| \leq \rho |S - T|$$

A similar relation holds for PC.

Shankar assumes a bound on the duration of the synchronization interval.

Old Condition 3 (bounded interval) *For nonfaulty clock p*

$$0 < r_{min} \leq t_p^{i+1} - t_p^i \leq r_{max}$$

The terms r_{min} and r_{max} are uninstantiated constants. In our formulation, we assume that a nominal duration (R) of an interval is determined from the implementation. We set a lower bound on R by placing restrictions on the events S^i. The term $\alpha(\beta' + 2\Lambda')$ will be shown to bound ADJ_p^i for nonfaulty process p. The function α is introduced in condition 11, β' is a bound on the separation of clocks at a particular Clocktime in each interval, and Λ' bounds the error in estimating the value of a remote clock.

New Condition 3 (bounded interval) *For nonfaulty clock p,*

$$S^i + \alpha(\beta' + 2\Lambda') < T_p^{i+1} < S^{i+1} - \alpha(\beta' + 2\Lambda')$$

A trivial consequence is that $R > 2\alpha(\beta' + 2\Lambda')$. Clearly, we can let $r_{min} = (R - \alpha(\beta' + 2\Lambda'))/(1+\rho)$ and $r_{max} = (1+\rho)(R + \alpha(\beta' + 2\Lambda'))$. The values for Λ', β', and $\alpha()$ will be determined by the implementation. The constraints on these values will be presented later.

Shankar and Schneider both assume the following in their proofs. The condition states that the elapsed time between two processes starting their ith interval clock is bounded. This property is closely related to the end result of the general theory (bounded skew), and should be derived in the context of an arbitrary algorithm.

Old Condition 4 (bounded delay) *For nonfaulty clocks p and q*

$$|t_q^i - t_p^i| \leq \beta$$

The related property, that for nonfaulty clocks p and q,

$$|s_q^i - s_p^i| \leq \beta'$$

is proven independently of the algorithm in section 3. This gives sufficient information to prove bounded delay directly from the algorithm, however, this proof depends upon the interpretation of T_p^i. Two interpretations and their corresponding proofs are given later.

The next condition states that all good clocks begin executing the protocol at the same instant of real time (and defines that time to be 0).

Old Condition 5 (initial synchronization) *For nonfaulty clock p*

$$t_p^0 = 0$$

This is clearly unsatisfiable, and will be discarded. It is used in proving the base case of the induction proof which establishes that good clocks are within δ_S of other good clocks, immediately following applying a correction. A satisfiable condition for that proof is that

New Condition 5 (initial synchronization) *For nonfaulty clock p*

$$IC_p^0(t_p^0) = T^0$$

where T^0 is some constant clock time known to all good clocks (i.e. T^0 is the clock time in the initial state). This just states that all nonfaulty clocks start the protocol at the same Clocktime. It is possible that this condition can be eliminated entirely.

Since we do not want process q to start its $(i+1)$th clock before process p starts its ith, Shankar states a nonoverlap condition

Old Condition 6 (nonoverlap)

$$\beta \leq r_{min}$$

This, with *bounded interval* and *bounded delay*, ensures that for good clocks p and q, $t_p^i \leq t_q^{i+1}$. We restate the condition in terms related to this presentation

New Condition 6 (nonoverlap)

$$\beta \leq (R - \alpha(\beta' + 2\Lambda'))/(1+\rho)$$

This essentially defines an additional constraint on R; namely that $R \geq (1+\rho)\beta + \alpha(\beta' + 2\Lambda')$.

All clock synchronization protocols require each process to obtain an estimate of the clock values for other processes within the system. Error in this estimate can be bounded, but not eliminated.

Old Condition 7 (reading error) *For nonfaulty clocks p and q*

$$|IC_q^i(t_p^{i+1}) - \Theta_p^{i+1}(q)| \leq \Lambda$$

However, in stating this condition an important consideration was overlooked. In some protocols, the ability to accurately read another processor's clock is dependent upon those clocks being already synchronized. Therefore, we add a precondition to the condition. Another useful observation is that an estimate of a remote clock's value is subject to two interpretations. It can be used to approximate the difference in Clocktime that two clocks show at an instant of real time, or it can be used to approximate the separation in real time that two clocks show the same Clocktime.

New Condition 7 (reading error) *For nonfaulty clocks p and q, if $|s_p^i - s_q^i| \le \beta'$,*

1. $|IC_q^i(t_p^{i+1}) - \Theta_p^{i+1}(q)| = |(\Theta_p^{i+1}(q) - IC_p^i(t_p^{i+1})) - (IC_q^i(t_p^{i+1}) - IC_p^i(t_p^{i+1}))| \le \Lambda$
2. $|(\Theta_p^{i+1}(q) - IC_p^i(t_p^{i+1})) - (ic_p^i(T_p^{i+1}) - ic_q^i(T_p^{i+1}))| \le \Lambda$
3. $|(\Theta_p^{i+1}(q) - IC_p^i(t_p^{i+1})) - (ic_p^i(S^i) - ic_q^i(S^i))| \le \Lambda'$

The first clause just restates the existing read error condition to illustrate that the read error can also be viewed as the error in an estimate of the difference in readings of Clocktime, i.e. the estimate allows us to approximately determine another clocks reading at a particular instant of time. The second clause recognizes that this difference can also be used to obtain an estimate of the time that a remote clock shows a particular Clocktime. The third clause is the one used in this paper; it relates real time separation of clocks when they read S^i to the estimated difference when the correction is applied. A bound on this could be derived from the second clause, but it is likely that a tighter bound can be derived from the implementation. Since the guaranteed skew is derived, in part, from the read error, we wish this bound to be as tight as possible. For this reason, we add it as an assumption to be satisfied in the context of a particular implementation.

The remaining constraints are unaltered in this presentation. They are exactly as Shankar stated them. The first of these is that there is bound to the number of faults which can be tolerated.

Old Condition 8 (bounded faults) *At any time t, the number of faulty processes is at most F.*

Synchronization algorithms execute a convergence function $cfn(p, \theta)$ which must satisfy the conditions of *translation invariance*, *precision enhancement*, and *accuracy preservation* irrespective of the physical constraints on the system. Shankar mechanically proves that Lamport and Melliar-Smith's Interactive Convergence function [5] satisfies these three conditions [2]. A mechanically checked proof that the fault-tolerant midpoint function used by Welch and Lynch [3] satisfies these conditions is presented in [6]. Schneider presents proofs that a number of other protocols satisfy these properties in [1].

Translation invariance states that the value obtained by adding x to the result of the convergence function should be the same as adding x to each of the clock readings used in evaluating the convergence function.

Old Condition 9 (translation invariance) *For any function θ mapping clocks to clock values,*

$$cfn(p, (\lambda n : \theta(n) + x)) = cfn(p, \theta) + x$$

Precision enhancement is a formalization of the concept that, after executing the convergence function, the values of interest should be close together.

Old Condition 10 (precision enhancement) *Given any subset C of the N clocks with $|C| \geq N - F$, and clocks p and q in C, then for any readings γ and θ satisfying the conditions*

1. *for any l in C, $|\gamma(\ell) - \theta(\ell)| \leq x$*
2. *for any l, m in C, $|\gamma(\ell) - \gamma(m)| \leq y$*
3. *for any l, m in C, $|\theta(\ell) - \theta(m)| \leq y$*

there is a bound $\pi(x, y)$ such that

$$|cfn(p, \gamma) - cfn(q, \theta)| \leq \pi(x, y)$$

Accuracy preservation formalizes the notion that there should be a bound on the amount of correction applied in any synchronization interval.

Old Condition 11 (accuracy preservation) *Given any subset C of the N clocks with $|C| \geq N - F$, and clock readings θ such that for any l and m in C, the bound $|\theta(\ell) - \theta(m)| \leq x$ holds, there is a bound $\alpha(x)$ such that for any q in C*

$$|cfn(p, \theta) - \theta(q)| \leq \alpha(x)$$

In the course of his proof of Theorem 1, Shankar derives the following additional conditions for an algorithm to be verified in this theory.

1. $\pi(2\Lambda + 2\beta\rho, \delta_S + 2\rho(r_{max} + \beta) + 2\Lambda) \leq \delta_S$
2. $\delta_S + 2\rho r_{max} \leq \delta$
3. $\alpha(\delta_S + 2\rho(r_{max} + \beta) + 2\Lambda) + \Lambda + 2\rho\beta \leq \delta$

These prevent trivial bounds for the properties of *precision enhancement* and *accuracy preservation*. Future plans include revisiting Shankar's proof to try to improve on these constraints. The next section uses the new conditions presented in this section, along with the old constraints on the convergence function to provide a general proof of bounded delay.

3 A General Solution for Bounded Delay

Schneider's schema assumes that $|t_p^i - t_q^i| \leq \beta$ for good clocks p and q, where t_p^i denotes the real time that clock p begins its ith interval clock (this is condition 4 in Shankar's presentation). Anyone wishing to use the generalized proof to verify an implementation correct must prove that this property is satisfied in the context of their implementation. In the case of the algorithm presented in [3], this is a non-trivial proof.

The difficulty stems, in part, from the inherent ambiguity in the interpretation of t_p^{i+1} in the context of an arbitrary algorithm. Relating the event to a particular clock time is difficult because it serves as a crossover point between two interval clocks. The logical clock implemented by the algorithm undergoes an instantaneous shift in its representation of time. Thus the local clock readings surrounding the time of adjustment may show a particular clock time twice, or never. The event t_p^{i+1} is determined by the algorithm to occur when $IC_p^i(t) = T_p^{i+1}$, i.e. T_p^{i+1} is the clock time for applying the adjustment $ADJ_p^i = (adj_p^{i+1} - adj_p^i)$. This also means that $t_p^{i+1} = ic_p^i(T_p^{i+1})$. In an instantaneous adjustment algorithm there are at least two possibilities:

1. $T_p^{i+1} = (i+1)R + T^0$, or
2. $T_p^{i+1} = (i+1)R + T^0 - ADJ_p^i$.

A more stable frame of reference is needed for bounding the separation of events. Welch and Lynch exploit their mechanism for reading remote clocks to provide this frame of reference. Every clock in the system sends a synchronization pulse when its virtual clock reads $S^i = iR + S^0$, where S^0 denotes the first exchange of clock values. Let s_p^i denote the earliest real time that $IC_p^i(t) = S^i$. If we ignore any implied interpretation of event s_p^i, and just select S^i which satisfy condition 3 we have sufficient information to prove bounded delay for an arbitrary algorithm.

The general proof follows closely the argument given in [3]. The proof adapted is that of Theorem 4 of [3, section 6]. We wish to prove for good clocks p and q that $|t_p^i - t_q^i| \leq \beta$. To establish this we first prove the following:

Theorem 2 *(bounded delay offset) For nonfaulty clocks p and q, and for $i \geq 0$.*

(a) If $i \geq 1$, then $|ADJ_p^{i-1}| \leq \alpha(\beta' + 2\Lambda')$.
(b) $|s_p^i - s_q^i| \leq \beta'$.

Proof: By induction on i. The base case ($i = 0$) is trivial; part (a) is vacuously true and (b) is true by assumption.

Assuming that (a) and (b) are true for i we proceed by showing they hold for $i + 1$

(a)

We begin by recognizing that (a) is an instance of *accuracy preservation*. $ADJ_p^{(i+1)-1} = adj_p^{i+1} - adj_p^i = cfn(p, \Theta_p^{i+1}) - IC_p^i(t_p^{i+1})$. Since $IC_p^i(t_p^{i+1}) = \Theta_p^{i+1}(p)$ (no error in reading own clock), we have an instance of accuracy preservation:

$$|cfn(p, \Theta_p^{i+1}) - \Theta_p^{i+1}(p)| \leq \alpha(x).$$

All that is required is to show that $\beta' + 2\Lambda'$ substituted for x satisfies the hypotheses of accuracy preservation.

We need to establish that for good ℓ, m,

$$|\Theta_p^{i+1}(\ell) - \Theta_p^{i+1}(m)| \leq \beta' + 2\Lambda'$$

We know from the induction hypothesis that for good clocks p and q,

$$|s_p^i - s_q^i| = |ic_p^i(S^i) - ic_q^i(S^i)| \leq \beta'$$

Using reading error and the induction hypothesis we get for nonfaulty clocks p and q

$$|(\Theta_p^{i+1}(q) - IC_p^i(t_p^{i+1})) - (ic_p^i(S^i) - ic_q^i(S^i))| \leq \Lambda'$$

We proceed as follows:

$$\begin{aligned}
&|\Theta_p^{i+1}(\ell) - \Theta_p^{i+1}(m)| \\
&\quad = \quad |(\Theta_p^{i+1}(\ell) - \Theta_p^{i+1}(m)) + (IC_p^i(t_p^{i+1}) - IC_p^i(t_p^{i+1})) \\
&\qquad\quad + (ic_p^i(S^i) - ic_p^i(S^i)) + (ic_\ell^i(S^i) - ic_\ell^i(S^i)) + (ic_m^i(S^i) - ic_m^i(S^i))| \\
&\quad \leq \quad |ic_l^i(S^i) - ic_m^i(S^i)| + |(\Theta_p^{i+1}(\ell) - IC_p^i(t_p^{i+1})) - (ic_p^i(S^i) - ic_\ell^i(S^i))| \\
&\qquad\quad + |(\Theta_p^{i+1}(m) - IC_p^i(t_p^{i+1})) - (ic_p^i(S^i) - ic_m^i(S^i))| \\
&\quad \leq \quad \beta' + 2\Lambda'
\end{aligned}$$

We get the last step by substituting ℓ and m for p and q respectively in the induction hypothesis, then using reading error twice, substituting first ℓ for q and then m for q.

(b)

All supporting lemmas introduced in this section implicitly assume both the induction hypothesis and part (a) for $i+1$. In Welch and Lynch's presentation they introduce a variant of precision enhancement. We restate it here in the context of the general protocol:

Lemma 1 *For good clocks p and q,*

$$|(ic_p^i(S^i) - ic_q^i(S^i)) - (ADJ_p^i - ADJ_q^i)| \leq \pi(2\Lambda' + 2, \beta' + 2\Lambda')$$

Proof: We begin by recognizing that $ADJ_p^i = cfn(p, (\lambda\ell.\Theta_p^{i+1}(\ell) - IC_p^i(t_p^{i+1})))$ (and similarly for ADJ_q^i). A simple rearrangement of the terms give us

$$\begin{aligned}
&|(ic_p^i(S^i) - ic_q^i(S^i)) - (ADJ_p^i - ADJ_q^i)| \\
&\quad = \ |(ADJ_p^i - ic_p^i(S^i)) - (ADJ_q^i - ic_q^i(S^i))|
\end{aligned}$$

To use translation invariance, we first need to convert the terms $ic_p^i(S^i)$ and $ic_q^i(S^i)$ to Clocktime. We do this via the integer floor and ceiling functions. Without loss of generality, assume that $(ADJ_p^i - ic_p^i(S^i)) \geq (ADJ_q^i - ic_q^i(S^i))$.

$$\begin{aligned}
&|(ADJ_p^i - ic_p^i(S^i)) - (ADJ_q^i - ic_q^i(S^i))| \\
&\leq \ |(ADJ_p^i - \lfloor ic_p^i(S^i)\rfloor) - (ADJ_q^i - \lceil ic_q^i(S^i)\rceil)| \\
&= \ |cfn(p, (\lambda\ell.\Theta_p^{i+1}(\ell) - IC_p^i(t_p^{i+1}) - \lfloor ic_p^i(S^i)\rfloor)) - cfn(q, (\lambda\ell.\Theta_q^{i+1}(\ell) - IC_q^i(t_q^{i+1}) - \lceil ic_q^i(S^i)\rceil))|
\end{aligned}$$

All that is required is to demonstrate that if $(\lambda\ell.\Theta_p^{i+1}(\ell) - IC_p^i(t_p^{i+1}) - \lfloor ic_p^i(S^i)\rfloor) = \gamma$ and $(\lambda\ell.\Theta_q^{i+1}(\ell) - IC_q^i(t_q^{i+1}) - \lceil ic_q^i(S^i)\rceil) = \theta$, they satisfy the hypotheses of precision enhancement.

We know from reading error and the induction hypothesis that

$$|(\Theta_p^{i+1}(\ell) - IC_p^i(t_p^{i+1})) - (ic_p^i(S^i) - ic_\ell^i(S^i))| \leq \Lambda'$$

To satisfy the first hypothesis of precision enhancement we notice that

$$\begin{aligned}
&|(\lambda\ell.\Theta_p^{i+1}(\ell) - IC_p^i(t_p^{i+1}) - \lfloor ic_p^i(S^i)\rfloor)(\ell) - (\lambda\ell.\Theta_q^{i+1}(\ell) - IC_q^i(t_q^{i+1}) - \lceil ic_q^i(S^i)\rceil)(\ell)| \\
&\quad = \ |(\Theta_p^{i+1}(\ell) - IC_p^i(t_p^{i+1}) - \lfloor ic_p^i(S^i)\rfloor) - (\Theta_q^{i+1}(\ell) - IC_q^i(t_q^{i+1}) - \lceil ic_q^i(S^i)\rceil)| \\
&\quad = \ |((\Theta_p^{i+1}(\ell) - IC_p^i(t_p^{i+1})) - (\lfloor ic_p^i(S^i)\rfloor - ic_\ell^i(S^i))) \\
&\qquad\ -((\Theta_q^{i+1}(\ell) - IC_q^i(t_q^{i+1})) - (\lceil ic_q^i(S^i)\rceil - ic_\ell^i(S^i)))| \\
&\quad \leq \ 2\Lambda' + 2
\end{aligned}$$

Therefore, we can substitute $2\Lambda' + 2$ for x to satisfy the first hypothesis of precision enhancement.

To satisfy the second and third hypothesis we proceed as follows (the argument presented is for $(\lambda\ell.\Theta_p^{i+1}(\ell) - IC_p^i(t_p^{i+1}) - \lfloor ic_p^i(S^i)\rfloor) = \gamma$). We need a y such that

$$|(\lambda\ell.\Theta_p^{i+1}(\ell) - IC_p^i(t_p^{i+1}) - \lfloor ic_p^i(S^i)\rfloor)(\ell) - (\lambda\ell.\Theta_p^{i+1}(\ell) - IC_p^i(t_p^{i+1}) - \lfloor ic_p^i(S^i)\rfloor)(m)| \leq y.$$

We know that

$$\begin{aligned}
&|(\lambda\ell.\Theta_p^{i+1}(\ell) - IC_p^i(t_p^{i+1}) - \lfloor ic_p^i(S^i)\rfloor)(\ell) - (\lambda\ell.\Theta_p^{i+1}(\ell) - IC_p^i(t_p^{i+1}) - \lfloor ic_p^i(S^i)\rfloor)(m)| \\
&= |(\Theta_p^{i+1}(\ell) - IC_p^i(t_p^{i+1}) - \lfloor ic_p^i(S^i)\rfloor) - (\Theta_p^{i+1}(m) - IC_p^i(t_p^{i+1}) - \lfloor ic_p^i(S^i)\rfloor) \\
&= |\Theta_p^{i+1}(\ell) - \Theta_p^{i+1}(m)|.
\end{aligned}$$

The argument in part (a) shows that this value is bounded by $\beta' + 2\Lambda'$ which is the desired y for the remaining hypotheses of precision enhancement. ■

Now we bound the separation of $ic_p^{i+1}(T)$ and $ic_q^{i+1}(T)$ for all T.

Lemma 2 *For good clocks p and q, and clock time T,*

$$|ic_p^{i+1}(T) - ic_q^{i+1}(T)| \leq 2\rho(|T - S^i| + \alpha(\beta' + 2\Lambda')) + \pi(2\Lambda' + 2, \beta' + 2\Lambda')$$

Proof: The proof is taken verbatim (modulo notational differences) from [3, Lemma 10]. Note that $ic_p^{i+1}(T) = ic_p^i(T - ADJ_p^i)$ and $ic_q^{i+1}(T) = ic_q^i(T - ADJ_q^i)$. Now

$$\begin{aligned}
&|ic_p^{i+1}(T) - ic_q^{i+1}(T)| \\
&\leq |ic_p^i(T - ADJ_p^i) - ic_p^i(S^i) - (T - ADJ_p^i - S^i)| \\
&\quad + |ic_q^i(T - ADJ_q^i) - ic_q^i(S^i) - (T - ADJ_q^i - S^i)| \\
&\quad + |(ic_p^i(S^i) - ic_q^i(S^i)) - (ADJ_p^i - ADJ_q^i)|
\end{aligned}$$

The three terms are bounded separately. By the second corollary of bounded drift we get

$$\begin{aligned}
&|ic_p^i(T - ADJ_p^i) - ic_p^i(S^i) - (T - ADJ_p^i - S^i)| \\
&\leq \rho|T - S^i - ADJ_p^i| \\
&\leq \rho(|T - S^i| + \alpha(\beta' + 2\Lambda')), \text{ from part (a) for } i+1.
\end{aligned}$$

The second term is similarly bounded. Lemma 1 bounds the third term. Adding the bounds and simplifying gives the result. ■

This leads to the desired result:

Lemma 3 *For good clocks p and q,*

$$|s_p^{i+1} - s_q^{i+1}| \leq 2\rho(R + \alpha(\beta' + 2\Lambda')) + \pi(2\Lambda' + 2, \beta' + 2\Lambda') \leq \beta'$$

Proof: This is simply an instance of Lemma 2 with S^{i+1} substituted for T. ∎

This completes the proof of Theorem 2. Algebraic manipulations on the inequality

$$2\rho(R + \alpha(\beta' + 2\Lambda')) + \pi(2\Lambda' + 2, \beta' + 2\Lambda') \leq \beta'$$

give us an upper bound for R.

3.1 Relationship to Shankar's Mechanical Proof

We begin by noticing that both instantaneous adjustment schemes presented in this paper allow for a simple derivation of a β that satisfies the condition of bounded delay. These are sufficient to establish condition 4. Notice that knowledge of the algorithm is required in order to fully establish this property.

1. When $T_p^{i+1} = (i+1)R + T^0$, let $\beta = \beta' + 2\rho(T_p^{i+1} - S^i)$.
2. When $T_p^{i+1} = (i+1)R + T^0 - ADJ_p^i$, let $\beta = \beta' - 2\rho(S^i - IC_p^i(t_p^i))$.

This implies that all down stream proofs need not be altered. However, it is possible that some bounds and arguments can be improved. This leaves us with a set of conditions which are much easier to satisfy for a particular implementation. A proof that an implementation is an instance of this extended theory requires the following:

- Prove the properties of translation invariance, precision enhancement and accuracy preservation for the chosen convergence function.
- Identify data structures in the implementation which correspond to the algebraic definitions of clocks. Prove that the structures used in the implementation satisfy the definitions.
- Prove that the implementation correctly executes a variation of the following algorithm:

  $i \leftarrow 0$
  do forever {
      exchange clock values
      determine adjustment for this interval
      determine T^{i+1} (local time to apply correction)
      when $IC^i(t) = T^{i+1}$ apply correction; $i \leftarrow i + 1$
  }

- Prove the new condition of read error in the context of the algorithm.
- Solve the four (three from [2], one from above) derived inequalities using values determined from the implementation.

- Prove correct a mechanism for establishing initial synchronization ($|s_p^0 - s_q^0| \leq \beta'$). Ensure that β' is as small as possible within the constraints of the aforementioned inequalities.
- If the protocol does not behave in the manner of either instantaneous adjustment option presented in this paper, it will be necessary to use another means to establish $\forall i : |t_p^i - t_q^i| \leq \beta$ from $\forall i : |s_p^i - s_q^i| \leq \beta'$.

3.2 EHDM Proofs of Bounded Delay

The EHDM (version 5.2) proofs and supporting definitions and axioms are in the modules **delay**, **delay2**, **delay3** and **delay4**. LaTeX formatted listings of these modules are in the appendix.[2] Some of the revised constraints presented in section 2 are in module **delay**. The most difficult aspect of the proofs was determining a reasonable predicate to express *nonfaulty clocks*. Since we would like to express transient fault recovery in the theory, it is necessary to avoid the axiom **correct_closed** from Shankar's module **clockassumptions**[3] The notion of non-faulty clocks is expressed by the following from module **delay**.

$$
\begin{array}{l}
\mathsf{correct\_during}\text{: } \mathsf{function}[\mathsf{process}, \mathsf{time}, \mathsf{time} \rightarrow \mathsf{bool}] = \\
\quad (\, \lambda\, p, t, s : t \leq s \wedge (\, \forall\, t_1 : t \leq t_1 \wedge t_1 \leq s \supset \mathsf{correct}(p, t_1))) \\
\mathsf{wpred}\text{: } \mathsf{function}[\mathsf{event} \rightarrow \mathsf{function}[\mathsf{process} \rightarrow \mathsf{bool}]] \\
\mathsf{rpred}\text{: } \mathsf{function}[\mathsf{event} \rightarrow \mathsf{function}[\mathsf{process} \rightarrow \mathsf{bool}]] \\
\mathsf{wvr\_pred}\text{: } \mathsf{function}[\mathsf{event} \rightarrow \mathsf{function}[\mathsf{process} \rightarrow \mathsf{bool}]] = \\
\quad (\, \lambda\, i : (\, \lambda\, p : \mathsf{wpred}(i)(p) \vee \mathsf{rpred}(i)(p)))
\end{array}
$$

$$\mathsf{wpred\_ax}\text{: } \textbf{Axiom } \mathsf{count}(\mathsf{wpred}(i), N) \geq N - F$$

$$\mathsf{wpred\_correct}\text{: } \textbf{Axiom } \mathsf{wpred}(i)(p) \supset \mathsf{correct\_during}(p, t_p^i, t_p^{i+1})$$

$$\mathsf{wpred\_preceding}\text{: } \textbf{Axiom } \mathsf{wpred}(i+1)(p) \supset \mathsf{wpred}(i)(p) \vee \mathsf{rpred}(i)(p)$$

$$\mathsf{wpred\_rpred\_disjoint}\text{: } \textbf{Axiom } \neg(\mathsf{wpred}(i)(p) \wedge \mathsf{rpred}(i)(p))$$

$$
\begin{array}{l}
\mathsf{wpred\_bridge}\text{: } \textbf{Axiom} \\
\quad \mathsf{wvr\_pred}(i)(p) \wedge \mathsf{correct\_during}(p, t_p^{i+1}, t_p^{i+2}) \supset \mathsf{wpred}(i+1)(p)
\end{array}
$$

Also, module **delay3** states the following axiom:

$$
\begin{array}{l}
\mathsf{recovery\_lemma}\text{: } \textbf{Axiom} \\
\quad \mathsf{delay\_pred}(i) \wedge \mathsf{ADJ\_pred}(i+1) \\
\qquad\qquad \wedge\, \mathsf{rpred}(i)(p) \wedge \mathsf{correct\_during}(p, t_p^{i+1}, t_p^{i+2}) \wedge \mathsf{wpred}(i+1)(q) \\
\qquad \supset |s_p^{i+1} - s_q^{i+1}| \leq \beta'
\end{array}
$$

[2] A slightly modified version of Shankar's module **clockassumptions** is also included in the appendix for completeness.

[3] This axiom has not yet been removed from the general theory. None of the proofs of ***bounded delay offset*** depend on it, however.

There are two predicates defined, wpred and rpred. Wpred is used to denote a working clock, i.e. it is not faulty and is in the proper state. Rpred denotes a process that is not faulty, but has not yet recovered proper state information. Correct is a predicate taken from Shankar's proof which states whether or not a clock is fault-free at a particular instance of real time. Correct_during is used to denote correctness of a clock over an interval of time. In order to reason about transient recovery it is necessary to provide an rpred that satisfies these relationships. If we do not plan on establishing transient recovery, let rpred$(i) = (\lambda p : \mathit{false})$. In this case, axioms recovery_lemma and wpred_rpred_disjoint are vacuously true, and the remaining axiom are analogous to Shankar's correct_closed. This reduces to a system in which the only correct clocks are those that have been so since the beginning of the protocol. This is precisely what should be true if no recovery is possible.

The restated property of bounded drift is captured by axioms RATE_1 and RATE_2. The new constraints for bounded interval are rts_new_1 and rts_new_2. Bounded delay init is expressed by bnd_delay_init. The third clause of the new reading error is reading_error3. The other two clauses are not used in this proof. An additional assumption not included in the constraints given in section 2 is that there is no error in reading your own clock. This is captured by read_self. In addition there were a few assumptions included defining interrelationships of some of the constants required by the theory.

The statement of Theorem 2 is bnd_delay_offset in module delay2. The main step of the inductive proof for part (a) is captured by good_Readclock. This, with accuracy preservation was sufficient to establish bnd_delay_offset_ind_a. Part (b) is more involved. Lemma delay_prec_enh in module delay2 is the machine checked version of lemma 1. Module delay3 contains the remaining proofs for part (b). Lemma 2 is presented as bound_future. The first two terms in the proof are bounded by lemma bound_future1, the third by delay_prec_enh. Lemma bound_FIXTIME completes the proof.

Module delay4 contains the proofs that each of the proposed substitutions for β satisfy the condition of bounded delay. Option 1 is captured by option1_bounded_delay, and option 2 is expressed by option2_bounded_delay. The EHDM proof chain status, demonstrating that all proof obligations have been met, can be found in the appendix. The task of mechanically verifying the proofs also forced some minor revisions to some hand proofs in an earlier draft of this paper. The errors revealed by the mechanical proof included invalid substitution of reals for integers, and arithmetic sign errors.

4 Concluding Remarks

This paper presents a mechanically confirmed proof for satisfying the condition *bounded delay* in the context of an arbitrary clock synchronization algorithm. The general theory presented by Schneider (and mechanically verified by Shankar) assumes this property. However, for some clock synchronization algorithms, the difficulty of the proof required to establish this property is comparable to that of directly proving the algorithm correct. If we wish to use Schneider's paradigm to simplify the verification of clock synchronization systems, a general proof of bounded delay is required. The proof given by Welch and Lynch for a related property was generalized and recast in the context of Schneider's general theory. In addition, changes to the underlying assumptions of the theory were given. These changes should ease the task of satisfying the assumptions in the course of verifying an implementation. The proofs presented here were sufficient to convince EHDM that the property of bounded delay can be satisfied in a general manner. Furthermore, Shankar's mechanically checked proofs still hold for the modified theory (modulo minor changes). It is possible that reworking Shankar's proofs using the new constraints will yield better bounds on the derived constraints.

A Proof Chain Status

```
Terse proof chains for module delay4

Use of the formula
  delay.RATE_lemma1_iclock
requires the following TCCs to be proven
  delay_tcc.RATE_2_TCC1
  delay_tcc.RATE_2_iclock_TCC1
  delay_tcc.rate_simplify_TCC1

Use of the formula
  division.div_ineq
requires the following TCCs to be proven
  division_tcc.mult_div_1_TCC1
  division_tcc.mult_div_TCC1
  division_tcc.div_cancel_TCC1
  division_tcc.ceil_mult_div_TCC1
  division_tcc.div_nonnegative_TCC1
  division_tcc.div_ineq_TCC1
  division_tcc.div_minus_1_TCC1

Use of the formula
  delay2.bnd_delay_offset
requires the following TCCs to be proven
  delay2_tcc.ADJ_pred_TCC1
  delay2_tcc.ADJ_pred_TCC2

Use of the formula
  natinduction.induction
requires the following TCCs to be proven
  natinduction_tcc.ind_m_proof_TCC1

Use of the formula
  noetherian[naturalnumber, natinduction.less].general_induction
requires the following assumptions to be discharged
  noetherian[naturalnumber, natinduction.less].well_founded

                        SUMMARY

The proof chain is complete

The axioms and assumptions at the base are:
  clockassumptions.IClock_defn
  clockassumptions.accuracy_preservation_ax
  clockassumptions.precision_enhancement_ax
  clockassumptions.rho_0
```

```
  clockassumptions.translation_invariance
  delay.RATE_1
  delay.RATE_2
  delay.R_FIX_SYNC_0
  delay.bnd_delay_init
  delay.fix_between_sync
  delay.read_self
  delay.reading_error3
  delay.rts_new_1
  delay.rts_new_2
  delay.synctime_defn
  delay.wpred_ax
  delay.wpred_correct
  delay.wpred_preceding
  delay3.betaprime_ax
  delay3.recovery_lemma
  delay4.option1_alg
  delay4.option2_alg
  division.mult_div_1
  division.mult_div_2
  division.mult_div_3
  floor_ceil.ceil_defn
  floor_ceil.floor_defn
  multiplication.mult_non_neg
  multiplication.mult_pos
  noetherian[EXPR, EXPR].general_induction
Total: 30

The definitions and type-constraints are:
  absmod.abs
  clockassumptions.Adj
  clockassumptions.okay_Readpred
  clockassumptions.okay_pairs
  delay.ADJ
  delay.FIXTIME
  delay.correct_during
  delay.fixtime
  delay.iclock
  delay2.ADJ_pred
  delay2.delay_pred
  delay3.good_interval
  multiplication.mult
Total: 13

The formulae used are:
  absmod.abs_3_bnd
  absmod.abs_com
```

```
absmod.abs_ge0
absmod.abs_plus
delay.ADJ_lem1
delay.ADJ_lem2
delay.FIXTIME_bound
delay.RATE_1_iclock
delay.RATE_2_simplify
delay.RATE_2_simplify_iclock
delay.RATE_lemma1_iclock
delay.RATE_lemma1_iclock_sym
delay.RATE_lemma2
delay.RATE_lemma2_iclock
delay.R11hack
delay.correct_during_hi
delay.correct_during_sub_left
delay.correct_during_sub_right
delay.correct_during_trans
delay.diff_squares
delay.iclock_ADJ_lem
delay.iclock_defn
delay.mult_abs_hack
delay.mult_assoc
delay.rate_simplify
delay.rate_simplify_step
delay.wpred_fixtime
delay.wpred_fixtime_low
delay.wpred_hi_lem
delay2.ADJ_hack
delay2.abs_hack
delay2.absceil
delay2.absfloor
delay2.abshack2
delay2.abshack3
delay2.abshack4
delay2.abshack5
delay2.abshack6a
delay2.abshack6b
delay2.abshack7
delay2.bnd_delay_offset
delay2.bnd_delay_offset_0
delay2.bnd_delay_offset_ind
delay2.bnd_delay_offset_ind_a
delay2.bnd_delay_offset_ind_b
delay2.ceil_hack
delay2.delay_prec_enh
delay2.delay_prec_enh_step1
delay2.delay_prec_enh_step1_sym
```

```
delay2.floor_hack
delay2.good_ReadClock
delay2.prec_enh_hyp1
delay2.prec_enh_hyp_2
delay2.prec_enh_hyp_3
delay2_tcc.ADJ_pred_TCC1
delay2_tcc.ADJ_pred_TCC2
delay3.ADJ_bound
delay3.Alpha_0
delay3.R_0_hack
delay3.R_0_lem
delay3.abs_0
delay3.abs_minus
delay3.abshack
delay3.abshack2
delay3.abshack_future
delay3.bound_FIXTIME
delay3.bound_FIXTIME2
delay3.bound_future
delay3.bound_future1
delay3.bound_future1_step
delay3.bound_future1_step_a
delay3.bound_future1_step_b
delay3.delay_offset
delay3.good_interval_lem
delay4.option2_convert_lemma
delay4.option2_good_interval
delay_tcc.RATE_2_TCC1
delay_tcc.RATE_2_iclock_TCC1
delay_tcc.rate_simplify_TCC1
division.div_cancel
division.div_ineq
division.mult_div
division_tcc.ceil_mult_div_TCC1
division_tcc.div_cancel_TCC1
division_tcc.div_ineq_TCC1
division_tcc.div_minus_1_TCC1
division_tcc.div_nonnegative_TCC1
division_tcc.mult_div_1_TCC1
division_tcc.mult_div_TCC1
multiplication.distrib
multiplication.distrib_minus
multiplication.mult_com
multiplication.mult_gt
multiplication.mult_ldistrib
multiplication.mult_ldistrib_minus
multiplication.mult_leq_2
```

```
  multiplication.mult_lident
  multiplication.mult_rident
  multiplication.pos_product
  natinduction.induction
  natinduction_tcc.ind_m_proof_TCC1
  noetherian[naturalnumber, natinduction.less].well_founded
Total: 102

The completed proofs are:
  absmod.abs_3_bnd_proof
  absmod.abs_com_proof
  absmod.abs_ge0_proof
  absmod.abs_plus_pr
  delay.ADJ_lem1_pr
  delay.ADJ_lem2_pr
  delay.FIXTIME_bound_pr
  delay.RATE_1_iclock_pr
  delay.RATE_2_simplify_iclock_pr
  delay.RATE_2_simplify_pr
  delay.RATE_lemma1_iclock_pr
  delay.RATE_lemma1_iclock_sym_pr
  delay.RATE_lemma2_iclock_pr
  delay.RATE_lemma2_pr
  delay.R11hack_pr
  delay.correct_during_hi_pr
  delay.correct_during_sub_left_pr
  delay.correct_during_sub_right_pr
  delay.correct_during_trans_pr
  delay.diff_squares_pr
  delay.iclock_ADJ_lem_pr
  delay.iclock_defn_pr
  delay.mult_abs_hack_pr
  delay.mult_assoc_pr
  delay.rate_simplify_pr
  delay.rate_simplify_step_pr
  delay.wpred_fixtime_low_pr
  delay.wpred_fixtime_pr
  delay.wpred_hi_lem_pr
  delay2.ADJ_hack_pr
  delay2.abs_hack_pr
  delay2.absceil_pr
  delay2.absfloor_pr
  delay2.abshack2_pr
  delay2.abshack3_pr
  delay2.abshack4_pr
  delay2.abshack5_pr
  delay2.abshack6a_pr
```

```
delay2.abshack6b_pr
delay2.abshack7_pr
delay2.bnd_del_off_0_pr
delay2.bnd_del_off_ind_a_pr
delay2.bnd_delay_offset_ind_pr
delay2.bnd_delay_offset_pr
delay2.ceil_hack_pr
delay2.delay_prec_enh_pr
delay2.delay_prec_enh_step1_pr
delay2.delay_prec_enh_step1_sym_pr
delay2.floor_hack_pr
delay2.good_ReadClock_pr
delay2.prec_enh_hyp1_pr
delay2.prec_enh_hyp_2_pr
delay2.prec_enh_hyp_3_pr
delay2_tcc.ADJ_pred_TCC1_PROOF
delay2_tcc.ADJ_pred_TCC2_PROOF
delay3.ADJ_bound_pr
delay3.Alpha_0_pr
delay3.R_0_hack_pr
delay3.R_0_lem_pr
delay3.abs_0_pr
delay3.abs_minus_pr
delay3.abshack2_pr
delay3.abshack_future_pr
delay3.abshack_pr
delay3.bnd_delay_offset_ind_b_pr
delay3.bound_FIXTIME2_pr
delay3.bound_FIXTIME_pr
delay3.bound_future1_pr
delay3.bound_future1_step_a_pr
delay3.bound_future1_step_b_pr
delay3.bound_future1_step_pr
delay3.bound_future_pr
delay3.delay_offset_pr
delay3.good_interval_lem_pr
delay4.option1_bounded_delay_pr
delay4.option2_bounded_delay_pr
delay4.option2_convert_lemma_pr
delay4.option2_good_interval_pr
division.div_cancel_pr
division.div_ineq_pr
division.mult_div_pr
division_tcc.ceil_mult_div_TCC1_PROOF
division_tcc.div_cancel_TCC1_PROOF
division_tcc.div_ineq_TCC1_PROOF
division_tcc.div_minus_1_TCC1_PROOF
```

```
  division_tcc.div_nonnegative_TCC1_PROOF
  division_tcc.mult_div_1_TCC1_PROOF
  division_tcc.mult_div_TCC1_PROOF
  multiplication.distrib_minus_pr
  multiplication.distrib_proof
  multiplication.mult_com_pr
  multiplication.mult_gt_pr
  multiplication.mult_ldistrib_minus_proof
  multiplication.mult_ldistrib_proof
  multiplication.mult_leq_2_pr
  multiplication.mult_lident_proof
  multiplication.mult_rident_proof
  multiplication.pos_product_pr
  natinduction.discharge
  natinduction.ind_proof
  natinduction_tcc.ind_m_proof_TCC1_PROOF
  tcc_delay.RATE_2_TCC1_PROOF
  tcc_delay.RATE_2_iclock_TCC1_PROOF
  tcc_delay.rate_simplify_TCC1_PROOF
Total: 104
```

B LaTeX Formatted Listings

clockassumptions: **Module**

Using arith, countmod

Exporting all with countmod, arith

Theory

N: nat

N_0: **Axiom** $N > 0$

process: **Type is** nat
event: **Type is** nat
time: **Type is** number
Clocktime: **Type is** integer
$l, m, n, p, q, p_1, p_2, q_1, q_2, p_3, q_3$: **Var** process
i, j, k: **Var** event
x, y, z, r, s, t: **Var** time
X, Y, Z, R, S, T: **Var** Clocktime
γ, θ: **Var** function[process $\rightarrow$ Clocktime]
$\delta, \rho, r_{min}, r_{max}, \beta$: number
Λ, μ: Clocktime
$PC_{\star 1}(\star 2), VC_{\star 1}(\star 2)$: function[process, time $\rightarrow$ Clocktime]
$t_{\star 1}^{\star 2}$: function[process, event $\rightarrow$ time]
$\Theta_{\star 1}^{\star 2}$: function[process, event $\rightarrow$ function[process $\rightarrow$ Clocktime]]
$IC_{\star 1}^{\star 2}(\star 3)$: function[process, event, time $\rightarrow$ Clocktime]
correct: function[process, time $\rightarrow$ bool]
cfn: function[process, function[process $\rightarrow$ Clocktime] $\rightarrow$ Clocktime]
π: function[number, number $\rightarrow$ number]
α: function[number $\rightarrow$ number]

delta_0: **Axiom** $\delta \geq 0$

mu_0: **Axiom** $\mu \geq 0$

rho_0: **Axiom** $\rho \geq 0$

rho_1: **Axiom** $\rho < 1$

rmin_0: **Axiom** $r_{min} > 0$

rmax_0: **Axiom** $r_{max} > 0$

beta_0: **Axiom** $\beta \geq 0$

lamb_0: **Axiom** $\Lambda \geq 0$

init: **Axiom** correct$(p, 0) \supset PC_p(0) \geq 0 \wedge PC_p(0) \leq \mu$

correct_closed: **Axiom** $s \geq t \wedge \text{correct}(p, s) \supset \text{correct}(p, t)$

rate_1: **Axiom** $\text{correct}(p, s) \wedge s \geq t \supset PC_p(s) - PC_p(t) \leq \lceil (s - t) \star (1 + \rho) \rceil$

rate_2: **Axiom** $\text{correct}(p, s) \wedge s \geq t \supset PC_p(s) - PC_p(t) \geq \lfloor (s - t) \star (1 - \rho) \rfloor$

rts0: **Axiom** $\text{correct}(p, t) \wedge t \leq t_p^{i+1} \supset t - t_p^i \leq r_{max}$

rts1: **Axiom** $\text{correct}(p, t) \wedge t \geq t_p^{i+1} \supset t - t_p^i \geq r_{min}$

rts_0: **Lemma** $\text{correct}(p, t_p^{i+1}) \supset t_p^{i+1} - t_p^i \leq r_{max}$

rts_1: **Lemma** $\text{correct}(p, t_p^{i+1}) \supset t_p^{i+1} - t_p^i \geq r_{min}$

rts2: **Axiom** $\text{correct}(p, t) \wedge t \geq t_q^i + \beta \wedge \text{correct}(q, t) \supset t \geq t_p^i$

rts_2: **Axiom** $\text{correct}(p, t_p^i) \wedge \text{correct}(q, t_q^i) \supset t_p^i - t_q^i \leq \beta$

synctime_0: **Axiom** $t_p^0 = 0$

VClock_defn: **Axiom**
$\text{correct}(p, t) \wedge t \geq t_p^i \wedge t < t_p^{i+1} \supset VC_p(t) = IC_p^i(t)$

$adj_{\star 1}^{\star 2}$: function[process, event → Clocktime] =
$(\lambda\, p, i : (\textbf{ if } i > 0 \textbf{ then } cfn(p, \Theta_p^i) - PC_p(t_p^i) \textbf{ else } 0 \textbf{ end if}))$

IClock_defn: **Axiom** $\text{correct}(p, t) \supset IC_p^i(t) = PC_p(t) + adj_p^i$

Readerror: **Axiom** $\text{correct}(p, t_p^{i+1}) \wedge \text{correct}(q, t_p^{i+1})$
$\supset |\Theta_p^{i+1}(q) - IC_q^i(t_p^{i+1})| \leq \Lambda$

translation_invariance: **Axiom**
$cfn(p, (\lambda\, p_1 \rightarrow \text{Clocktime} : \gamma(p_1) + X)) = cfn(p, \gamma) + X$

ppred: **Var** function[process → bool]
F: process
okay_Readpred: function[function[process → Clocktime], number,
function[process → bool] → bool] =
$(\lambda\, \gamma, y, \text{ppred} : (\forall\, l, m : \text{ppred}(l) \wedge \text{ppred}(m) \supset |\gamma(l) - \gamma(m)| \leq y))$
okay_pairs: function[function[process → Clocktime],
function[process → Clocktime], number,
function[process → bool] → bool] =
$(\lambda\, \gamma, \theta, x, \text{ppred} : (\forall\, p_3 : \text{ppred}(p_3) \supset |\gamma(p_3) - \theta(p_3)| \leq x))$

N_maxfaults: **Axiom** $F \leq N$

precision_enhancement_ax: **Axiom**
$\text{count}(\text{ppred}, N) \geq N - F$
$\wedge$ okay_Readpred$(\gamma, y, \text{ppred})$
$\wedge$ okay_Readpred$(\theta, y, \text{ppred})$
$\wedge$ okay_pairs$(\gamma, \theta, x, \text{ppred}) \wedge \text{ppred}(p) \wedge \text{ppred}(q)$
$\supset |cfn(p, \gamma) - cfn(q, \theta)| \leq \pi(x, y)$

correct_count: **Axiom** count(($\lambda\, p$: correct(p, t)), N) $\geq N - F$

okay_Reading: function[function[process $\rightarrow$ Clocktime], number, time
$\rightarrow$ bool] =
($\lambda\, \gamma, y, t$: ($\forall\, p_1, q_1$:
correct(p_1, t) $\wedge$ correct(q_1, t) $\supset |\gamma(p_1) - \gamma(q_1)| \leq y$))

okay_Readvars: function[function[process $\rightarrow$ Clocktime],
function[process $\rightarrow$ Clocktime], number, time
$\rightarrow$ bool] =
($\lambda\, \gamma, \theta, x, t$: ($\forall\, p_3$: correct(p_3, t) $\supset |\gamma(p_3) - \theta(p_3)| \leq x$))

okay_Readpred_Reading: **Lemma**
okay_Reading(γ, y, t) $\supset$ okay_Readpred(γ, y, ($\lambda\, p$: correct(p, t)))

okay_pairs_Readvars: **Lemma**
okay_Readvars(γ, θ, x, t) $\supset$ okay_pairs(γ, θ, x, ($\lambda\, p$: correct(p, t)))

precision_enhancement: **Lemma**
okay_Reading(γ, y, t_p^{i+1})
$\wedge$ okay_Reading(θ, y, t_p^{i+1})
$\wedge$ okay_Readvars($\gamma, \theta, x, t_p^{i+1}$)
$\wedge$ correct(p, t_p^{i+1}) $\wedge$ correct(q, t_p^{i+1})
$\supset |cfn(p, \gamma) - cfn(q, \theta)| \leq \pi(x, y)$

okay_Reading_defn_lr: **Lemma**
okay_Reading(γ, y, t)
$\supset$ ($\forall\, p_1, q_1$: correct(p_1, t) $\wedge$ correct(q_1, t) $\supset |\gamma(p_1) - \gamma(q_1)| \leq y$)

okay_Reading_defn_rl: **Lemma**
($\forall\, p_1, q_1$: correct(p_1, t) $\wedge$ correct(q_1, t) $\supset |\gamma(p_1) - \gamma(q_1)| \leq y$)
$\supset$ okay_Reading(γ, y, t)

okay_Readvars_defn_lr: **Lemma**
okay_Readvars(γ, θ, x, t) $\supset$ ($\forall\, p_3$: correct(p_3, t) $\supset |\gamma(p_3) - \theta(p_3)| \leq x$)

okay_Readvars_defn_rl: **Lemma**
($\forall\, p_3$: correct(p_3, t) $\supset |\gamma(p_3) - \theta(p_3)| \leq x$) $\supset$ okay_Readvars(γ, θ, x, t)

accuracy_preservation_ax: **Axiom**
okay_Readpred(γ, x, ppred) $\wedge$ count(ppred, N) $\geq N - F \wedge$ ppred(p) $\wedge$ ppred(q)
$\supset |cfn(p, \gamma) - \gamma(q)| \leq \alpha(x)$

Proof

okay_Reading_defn_rl_pr: **Prove**
okay_Reading_defn_rl $\{p_1 \leftarrow p_1$@P1S, $q_1 \leftarrow q_1$@P1S$\}$ **from** okay_Reading

okay_Reading_defn_lr_pr: **Prove** okay_Reading_defn_lr **from**
okay_Reading $\{p_1 \leftarrow p_1$@CS, $q_1 \leftarrow q_1$@CS$\}$

okay_Readvars_defn_rl_pr: **Prove** okay_Readvars_defn_rl $\{p_3 \leftarrow p_3@\mathsf{P1S}\}$ **from**
okay_Readvars

okay_Readvars_defn_lr_pr: **Prove** okay_Readvars_defn_lr **from**
okay_Readvars $\{p_3 \leftarrow p_3@\mathsf{CS}\}$

precision_enhancement_pr: **Prove** precision_enhancement **from**
precision_enhancement_ax $\{\mathsf{ppred} \leftarrow (\lambda\, q : \mathsf{correct}(q, t_p^{i+1}))\}$,
okay_Readpred_Reading $\{t \leftarrow t_p^{i+1}\}$,
okay_Readpred_Reading $\{t \leftarrow t_p^{i+1},\ \gamma \leftarrow \theta\}$,
okay_pairs_Readvars $\{t \leftarrow t_p^{i+1}\}$,
correct_count $\{t \leftarrow t_p^{i+1}\}$

okay_Readpred_Reading_pr: **Prove** okay_Readpred_Reading **from**
okay_Readpred $\{\mathsf{ppred} \leftarrow (\lambda\, p : \mathsf{correct}(p, t))\}$,
okay_Reading $\{p_1 \leftarrow l@P1S,\ q_1 \leftarrow m@P1S\}$

okay_pairs_Readvars_pr: **Prove** okay_pairs_Readvars **from**
okay_pairs $\{\mathsf{ppred} \leftarrow (\lambda\, p : \mathsf{correct}(p, t))\}$, okay_Readvars $\{p_3 \leftarrow p_3@\mathsf{P1S}\}$

rts_0_proof: **Prove** rts_0 **from** rts0 $\{t \leftarrow t_p^{i+1}\}$

rts_1_proof: **Prove** rts_1 **from** rts1 $\{t \leftarrow t_p^{i+1}\}$

End clockassumptions

delay: **Module**

Using arith, clockassumptions

Exporting all with clockassumptions

Theory

p, q, p_1, q_1: **Var** process
i: **Var** event
X, S, T: **Var** Clocktime
s, t, t_1, t_2: **Var** time
γ: **Var** function[process $\rightarrow$ Clocktime]
β': number
R, Λ': Clocktime
ppred, ppred1: **Var** function[process $\rightarrow$ bool]
S^0: Clocktime
$S^{\star 1}$: function[event $\rightarrow$ Clocktime] $= (\lambda\, i : i * R + S^0)$
$pc_{\star 1}(\star 2)$: function[process, Clocktime $\rightarrow$ time]
$ic_{\star 1}^{\star 2}(\star 3)$: function[process, event, Clocktime $\rightarrow$ time] =
$(\lambda\, p, i, T : pc_p(T - adj_p^i))$
$s_{\star 1}^{\star 2}$: function[process, event $\rightarrow$ time] $= (\lambda\, p, i : ic_p^i(S^i))$
T^0: Clocktime
$T_{\star 1}^{\star 2}$: function[process, event $\rightarrow$ Clocktime]

synctime_defn: **Axiom** $t_p^{i+1} = ic_p^i(T_p^{i+1})$

synctime0_defn: **Axiom** $t_p^0 = pc_p(T^0)$

correct_during: function[process, time, time $\rightarrow$ bool] =
$(\lambda\, p, t, s : t \leq s \wedge (\forall\, t_1 : t \leq t_1 \wedge t_1 \leq s \supset \text{correct}(p, t_1)))$
wpred: function[event $\rightarrow$ function[process $\rightarrow$ bool]]
rpred: function[event $\rightarrow$ function[process $\rightarrow$ bool]]
wvr_pred: function[event $\rightarrow$ function[process $\rightarrow$ bool]] =
$(\lambda\, i : (\lambda\, p : \text{wpred}(i)(p) \vee \text{rpred}(i)(p)))$

wvr_defn: **Lemma** $\text{wvr\_pred}(i) = (\lambda\, p : \text{wpred}(i)(p) \vee \text{rpred}(i)(p))$

wpred_wvr: **Lemma** $\text{wpred}(i)(p) \supset \text{wvr\_pred}(i)(p)$

rpred_wvr: **Lemma** $\text{rpred}(i)(p) \supset \text{wvr\_pred}(i)(p)$

wpred_ax: **Axiom** $\text{count}(\text{wpred}(i), N) \geq N - F$

wvr_count: **Lemma** $\text{count}(\text{wvr\_pred}(i), N) \geq N - F$

wpred_correct: **Axiom** $\text{wpred}(i)(p) \supset \text{correct\_during}(p, t_p^i, t_p^{i+1})$

wpred_preceding: **Axiom** $\text{wpred}(i + 1)(p) \supset \text{wpred}(i)(p) \vee \text{rpred}(i)(p)$

wpred_rpred_disjoint: **Axiom** $\neg(\text{wpred}(i)(p) \wedge \text{rpred}(i)(p))$

wpred_bridge: **Axiom**
wvr_pred$(i)(p) \wedge$ correct_during$(p, t_p^{i+1}, t_p^{i+2}) \supset$ wpred$(i+1)(p)$

wpred_fixtime: **Lemma** wpred$(i)(p) \supset$ correct_during(p, s_p^i, t_p^{i+1})

wpred_fixtime_low: **Lemma** wpred$(i)(p) \supset$ correct_during(p, t_p^i, s_p^i)

correct_during_trans: **Lemma**
correct_during$(p, t, t_2) \wedge$ correct_during(p, t_2, s)
$\supset$ correct_during(p, t, s)

correct_during_sub_left: **Lemma**
correct_during$(p, t, s) \wedge t \leq t_2 \wedge t_2 \leq s \supset$ correct_during(p, t, t_2)

correct_during_sub_right: **Lemma**
correct_during$(p, t, s) \wedge t \leq t_2 \wedge t_2 \leq s \supset$ correct_during(p, t_2, s)

wpred_lo_lem: **Lemma** wpred$(i)(p) \supset$ correct(p, t_p^i)

wpred_hi_lem: **Lemma** wpred$(i)(p) \supset$ correct(p, t_p^{i+1})

correct_during_hi: **Lemma** correct_during$(p, t, s) \supset$ correct(p, s)

correct_during_lo: **Lemma** correct_during$(p, t, s) \supset$ correct(p, t)

clock_ax: **Axiom** $PC_p(pc_p(T)) = T$

iclock_defn: **Lemma** $ic_p^i(T) = pc_p(T - adj_p^i)$

iclock_lem: **Lemma** correct$(p, pc_p(T - adj_p^i)) \supset IC_p^i(ic_p^i(T)) = T$

$ADJ_{\star 1}^{\star 2}$: function[process, event $\to$ Clocktime] $= (\lambda\, p, i : adj_p^{i+1} - adj_p^i)$

IClock_ADJ_lem: **Lemma** correct$(p, t) \supset IC_p^{i+1}(t) = IC_p^i(t) + ADJ_p^i$

iclock_ADJ_lem: **Lemma** $ic_p^{i+1}(T) = ic_p^i(T - ADJ_p^i)$

rts_new_1: **Axiom** correct$(p, t_p^{i+1}) \supset S^i + \alpha(\beta' + 2 * \Lambda') < T_p^{i+1}$

rts_new_2: **Axiom** correct$(p, t_p^i) \supset T_p^i < S^i - \alpha(\beta' + 2 * \Lambda')$

FIXTIME_bound: **Lemma** correct$(p, t_p^{i+1}) \supset S^{i+1} > S^i + 2 * \alpha(\beta' + 2 * \Lambda')$

R_bound: **Lemma** correct$(p, t_p^{i+1}) \supset R > 2 * \alpha(\beta' + 2 * \Lambda')$

RATE_1: **Axiom** correct_during$(p, pc_p(T), pc_p(S)) \wedge S \geq T$
$\supset pc_p(S) - pc_p(T) \leq (S - T) \star (1 + \rho)$

RATE_2: **Axiom** correct_during$(p, pc_p(T), pc_p(S)) \wedge S \geq T$
$\supset pc_p(S) - pc_p(T) \geq (S - T)/(1 + \rho)$

RATE_1_iclock: **Lemma**

$$\text{correct\_during}(p, ic_p^i(T), ic_p^i(S)) \wedge S \geq T$$
$$\supset ic_p^i(S) - ic_p^i(T) \leq (S - T) \star (1 + \rho)$$

RATE_2_iclock: **Lemma**

$$\text{correct\_during}(p, ic_p^i(T), ic_p^i(S)) \wedge S \geq T$$
$$\supset ic_p^i(S) - ic_p^i(T) \geq (S - T)/(1 + \rho)$$

rate_simplify: **Lemma** $S \geq T \supset (S - T)/(1 + \rho) \geq (S - T) \star (1 - \rho)$

rate_simplify_step: **Lemma** $S \geq T \supset (1 + \rho) \star (S - T) \star (1 - \rho) \leq S - T$

RATE_2_simplify: **Lemma**

$$\text{correct\_during}(p, pc_p(T), pc_p(S)) \wedge S \geq T$$
$$\supset pc_p(S) - pc_p(T) \geq (S - T) \star (1 - \rho)$$

RATE_2_simplify_iclock: **Lemma**

$$\text{correct\_during}(p, ic_p^i(T), ic_p^i(S)) \wedge S \geq T$$
$$\supset ic_p^i(S) - ic_p^i(T) \geq (S - T) \star (1 - \rho)$$

RATE_lemma1: **Lemma**

$$\text{correct\_during}(p, pc_p(T), pc_p(S))$$
$$\wedge\ \text{correct\_during}(q, pc_q(T), pc_q(S)) \wedge S \geq T$$
$$\supset |pc_p(S) - pc_q(S)| \leq |pc_p(T) - pc_q(T)| + 2 * \rho \star (S - T)$$

RATE_lemma1_iclock: **Lemma**

$$\text{correct\_during}(p, ic_p^i(T), ic_p^i(S))$$
$$\wedge\ \text{correct\_during}(q, ic_q^i(T), ic_q^i(S)) \wedge S \geq T$$
$$\supset |ic_p^i(S) - ic_q^i(S)| \leq |ic_p^i(T) - ic_q^i(T)| + 2 * \rho \star (S - T)$$

RATE_lemma2: **Lemma**

$$\text{correct\_during}(p, pc_p(T), pc_p(S)) \wedge S \geq T$$
$$\supset |(pc_p(S) - S) - (pc_p(T) - T)| \leq \rho \star (|S - T|)$$

RATE_lemma2_iclock: **Lemma**

$$\text{correct\_during}(p, ic_p^i(T), ic_p^i(S)) \wedge S \geq T$$
$$\supset |(ic_p^i(S) - S) - (ic_p^i(T) - T)| \leq \rho \star (|S - T|)$$

bnd_delay_init: **Axiom** $\text{wpred}(0)(p) \wedge \text{wpred}(0)(q) \supset |s_p^0 - s_q^0| \leq \beta'$

reading_error3: **Axiom**

$$\text{correct\_during}(p, s_p^i, t_p^{i+1})$$
$$\wedge\ \text{correct\_during}(q, s_q^i, t_q^{i+1}) \wedge |s_p^i - s_q^i| \leq \beta'$$
$$\supset |(\Theta_p^{i+1}(q) - IC_p^i(t_p^{i+1})) - (s_p^i - s_q^i)| \leq \Lambda'$$

ADJ_lem1: **Lemma** $\text{correct\_during}(p, s_p^i, t_p^{i+1})$

$$\supset (ADJ_p^i = cfn(p, (\lambda p_1 : \Theta_p^{i+1}(p_1) - IC_p^i(t_p^{i+1}))))$$

ADJ_lem2: **Lemma** $\text{correct\_during}(p, s_p^i, t_p^{i+1})$

$$\supset (ADJ_p^i = cfn(p, \Theta_p^{i+1}) - IC_p^i(t_p^{i+1}))$$

read_self: **Axiom** wpred$(i)(p) \supset \Theta_p^{i+1}(p) = IC_p^i(t_p^{i+1})$

fix_between_sync: **Axiom**
correct_during$(p, t_p^i, t_p^{i+1}) \supset t_p^i < s_p^i \wedge s_p^i < t_p^{i+1}$

Proof

FIXTIME_bound_pr: **Prove** FIXTIME_bound **from** rts_new_1, rts_new_2 $\{i \leftarrow i + 1\}$

R_bound_pr: **Prove** R_bound **from** FIXTIME_bound, $S^{\star 1}$, $S^{\star 1}$ $\{i \leftarrow i + 1\}$

iclock_defn_pr: **Prove** iclock_defn **from** $ic_{\star 1}^{\star 2}(\star 3)$

wpred_fixtime_pr: **Prove** wpred_fixtime **from**
fix_between_sync,
wpred_correct,
correct_during_sub_right $\{s \leftarrow t_p^{i+1},\ t \leftarrow t_p^i,\ t_2 \leftarrow s_p^i\}$

wpred_fixtime_low_pr: **Prove** wpred_fixtime_low **from**
fix_between_sync,
wpred_correct,
correct_during_sub_left $\{s \leftarrow t_p^{i+1},\ t \leftarrow t_p^i,\ t_2 \leftarrow s_p^i\}$

correct_during_sub_left_pr: **Prove** correct_during_sub_left **from**
correct_during $\{s \leftarrow t_2\}$, correct_during $\{t_1 \leftarrow t_1 @ p1\}$

correct_during_sub_right_pr: **Prove** correct_during_sub_right **from**
correct_during $\{t \leftarrow t_2\}$, correct_during $\{t_1 \leftarrow t_1 @ p1\}$

correct_during_trans_pr: **Prove** correct_during_trans **from**
correct_during,
correct_during $\{s \leftarrow t_2,\ t_1 \leftarrow t_1 @ p1\}$,
correct_during $\{t \leftarrow t_2,\ t_1 \leftarrow t_1 @ p1\}$

wpred_wvr_pr: **Prove** wpred_wvr **from** wvr_defn

rpred_wvr_pr: **Prove** rpred_wvr **from** wvr_defn

wvr_defn_hack: **Lemma**
$(\forall\, p :$ wvr_pred$(i)(p) = ((\lambda\, p :$ wpred$(i)(p) \vee$ rpred$(i)(p))p))$

wvr_defn_hack_pr: **Prove** wvr_defn_hack **from** wvr_pred $\{p \leftarrow p@c\}$

wvr_defn_pr: **Prove** wvr_defn **from**
pred_extensionality
{pred1 ← wvr_pred(i),
pred2 ← $(\lambda\, p :$ wpred$(i)(p) \vee$ rpred$(i)(p))\}$,
wvr_defn_hack $\{p \leftarrow p@p1\}$

wvr_count_pr: **Prove** wvr_count **from**
  wpred_ax,
  count_imp
    {ppred1 ← wpred(i),
     ppred2 ← ($\lambda\, p$: wpred(i)(p) $\vee$ rpred(i)(p)),
     $n \leftarrow N$},
  wvr_defn,
  imp_pred_or {ppred1 ← wpred(i), ppred2 ← rpred(i)}

w, x, y, z: **Var** number

mult_abs_hack: **Lemma** $x \star (1 - \rho) \leq y \wedge y \leq x \star (1 + \rho) \supset |y - x| \leq \rho \star x$

mult_abs_hack_pr: **Prove** mult_abs_hack **from**
  mult_ldistrib $\{y \leftarrow 1,\ z \leftarrow \rho\}$,
  mult_ldistrib_minus $\{y \leftarrow 1,\ z \leftarrow \rho\}$,
  mult_rident,
  abs_3_bnd $\{x \leftarrow y,\ y \leftarrow x,\ z \leftarrow \rho \star x\}$,
  mult_com $\{y \leftarrow \rho\}$

RATE_1_iclock_pr: **Prove** RATE_1_iclock **from**
  RATE_1 $\{S \leftarrow S - adj_p^i,\ T \leftarrow T - adj_p^i\}$,
  iclock_defn,
  iclock_defn $\{T \leftarrow S\}$

RATE_2_iclock_pr: **Prove** RATE_2_iclock **from**
  RATE_2 $\{S \leftarrow S - adj_p^i,\ T \leftarrow T - adj_p^i\}$,
  iclock_defn,
  iclock_defn $\{T \leftarrow S\}$

RATE_2_simplify_iclock_pr: **Prove** RATE_2_simplify_iclock **from**
  RATE_2_simplify $\{S \leftarrow S - adj_p^i,\ T \leftarrow T - adj_p^i\}$,
  iclock_defn,
  iclock_defn $\{T \leftarrow S\}$

RATE_lemma1_sym: **Lemma**
  correct_during($p, pc_p(T), pc_p(S)$)
    $\wedge$ correct_during($q, pc_q(T), pc_q(S)$) $\wedge\ S \geq T \wedge pc_p(S) \geq pc_q(S)$
   $\supset |pc_p(S) - pc_q(S)| \leq |pc_p(T) - pc_q(T)| + 2 * \rho \star (S - T)$

Rl1hack: **Lemma** $w \leq x \wedge y \leq z \wedge y \geq x \supset |y - x| \leq |z - w|$

Rl1hack_pr: **Prove** Rl1hack **from** $|\star 1|$ $\{x \leftarrow y - x\}$, $|\star 1|$ $\{x \leftarrow z - w\}$

RATE_lemma1_sym_pr: **Prove** RATE_lemma1_sym **from**
RATE_1,
RATE_2_simplify $\{p \leftarrow q\}$,
Rl1hack
$\{x \leftarrow pc_q(S)$,
$y \leftarrow pc_p(S)$,
$w \leftarrow pc_q(T) + (S - T) \star (1 - \rho)$,
$z \leftarrow pc_p(T) + (S - T) \star (1 + \rho)\}$,
mult_ldistrib $\{x \leftarrow S - T,\ y \leftarrow 1,\ z \leftarrow \rho\}$,
mult_ldistrib_minus $\{x \leftarrow S - T,\ y \leftarrow 1,\ z \leftarrow \rho\}$,
abs_plus $\{x \leftarrow pc_p(T) - pc_q(T),\ y \leftarrow 2 * \rho \star (S - T)\}$,
mult_com $\{x \leftarrow \rho,\ y \leftarrow S - T\}$,
abs_ge0 $\{x \leftarrow 2 * \rho \star (S - T)\}$,
mult_non_neg $\{x \leftarrow \rho,\ y \leftarrow S - T\}$,
rho_0

RATE_lemma1_pr: **Prove** RATE_lemma1 **from**
RATE_lemma1_sym,
RATE_lemma1_sym $\{p \leftarrow q,\ q \leftarrow p\}$,
abs_com $\{x \leftarrow pc_p(S),\ y \leftarrow pc_q(S)\}$,
abs_com $\{x \leftarrow pc_p(T),\ y \leftarrow pc_q(T)\}$

RATE_lemma1_iclock_sym: **Lemma**
correct_during$(p, ic_p^i(T), ic_p^i(S))$
$\wedge$ correct_during$(q, ic_q^i(T), ic_q^i(S)) \wedge S \geq T \wedge ic_p^i(S) \geq ic_q^i(S)$
$\supset |ic_p^i(S) - ic_q^i(S)| \leq |ic_p^i(T) - ic_q^i(T)| + 2 * \rho \star (S - T)$

RATE_lemma1_iclock_sym_pr: **Prove** RATE_lemma1_iclock_sym **from**
RATE_1_iclock,
RATE_2_simplify_iclock $\{p \leftarrow q\}$,
Rl1hack
$\{x \leftarrow ic_q^i(S)$,
$y \leftarrow ic_p^i(S)$,
$w \leftarrow ic_q^i(T) + (S - T) \star (1 - \rho)$,
$z \leftarrow ic_p^i(T) + (S - T) \star (1 + \rho)\}$,
mult_ldistrib $\{x \leftarrow S - T,\ y \leftarrow 1,\ z \leftarrow \rho\}$,
mult_ldistrib_minus $\{x \leftarrow S - T,\ y \leftarrow 1,\ z \leftarrow \rho\}$,
abs_plus $\{x \leftarrow ic_p^i(T) - ic_q^i(T),\ y \leftarrow 2 * \rho \star (S - T)\}$,
mult_com $\{x \leftarrow \rho,\ y \leftarrow S - T\}$,
abs_ge0 $\{x \leftarrow 2 * \rho \star (S - T)\}$,
mult_non_neg $\{x \leftarrow \rho,\ y \leftarrow S - T\}$,
rho_0

RATE_lemma1_iclock_pr: **Prove** RATE_lemma1_iclock **from**
RATE_lemma1_iclock_sym,
RATE_lemma1_iclock_sym $\{p \leftarrow q,\ q \leftarrow p\}$,
abs_com $\{x \leftarrow ic_p^i(S),\ y \leftarrow ic_q^i(S)\}$,
abs_com $\{x \leftarrow ic_p^i(T),\ y \leftarrow ic_q^i(T)\}$

RATE_lemma2_pr: **Prove** RATE_lemma2 **from**
RATE_1,
RATE_2_simplify,
mult_abs_hack $\{x \leftarrow S - T,\ y \leftarrow pc_p(S) - pc_p(T)\}$,
abs_ge0 $\{x \leftarrow S - T\}$

RATE_lemma2_iclock_pr: **Prove** RATE_lemma2_iclock **from**
RATE_lemma2 $\{S \leftarrow S - adj_p^i,\ T \leftarrow T - adj_p^i\}$,
iclock_defn $\{T \leftarrow S\}$,
iclock_defn

wpred_lo_lem_pr: **Prove** wpred_lo_lem **from**
wpred_correct,
correct_during $\{s \leftarrow t_p^{i+1},\ t \leftarrow t_p^i,\ t_1 \leftarrow t_p^i\}$

wpred_hi_lem_pr: **Prove** wpred_hi_lem **from**
wpred_correct,
correct_during $\{s \leftarrow t_p^{i+1},\ t \leftarrow t_p^i,\ t_1 \leftarrow t_p^{i+1}\}$

correct_during_hi_pr: **Prove** correct_during_hi **from** correct_during $\{t_1 \leftarrow s\}$

correct_during_lo_pr: **Prove** correct_during_lo **from** correct_during $\{t_1 \leftarrow t\}$

mult_assoc: **Lemma** $x \star (y \star z) = (x \star y) \star z$

mult_assoc_pr: **Prove** mult_assoc **from**
$\star 1 \star \star 2$ $\{y \leftarrow y \star z\}$,
$\star 1 \star \star 2$,
$\star 1 \star \star 2$ $\{x \leftarrow y,\ y \leftarrow z\}$,
$\star 1 \star \star 2$ $\{x \leftarrow x \star y,\ y \leftarrow z\}$

diff_squares: **Lemma** $(1 + \rho) \star (1 - \rho) = 1 - \rho \star \rho$

diff_squares_pr: **Prove** diff_squares **from**
distrib $\{x \leftarrow 1,\ y \leftarrow \rho,\ z \leftarrow 1 - \rho\}$,
mult_lident $\{x \leftarrow 1 - \rho\}$,
mult_ldistrib_minus $\{x \leftarrow \rho,\ y \leftarrow 1,\ z \leftarrow \rho\}$,
mult_rident $\{x \leftarrow \rho\}$

rate_simplify_step_pr: **Prove** rate_simplify_step **from**
mult_com $\{x \leftarrow (S - T),\ y \leftarrow (1 - \rho)\}$,
mult_assoc $\{x \leftarrow 1 + \rho,\ y \leftarrow 1 - \rho,\ z \leftarrow S - T\}$,
diff_squares,
distrib_minus $\{x \leftarrow 1,\ y \leftarrow \rho \star \rho,\ z \leftarrow S - T\}$,
mult_lident $\{x \leftarrow S - T\}$,
pos_product $\{x \leftarrow \rho \star \rho,\ y \leftarrow S - T\}$,
pos_product $\{x \leftarrow \rho,\ y \leftarrow \rho\}$,
rho_0

rate_simplify_pr: **Prove** rate_simplify **from**
  div_ineq
    $\{z \leftarrow (1+\rho),$
     $y \leftarrow (S-T),$
     $x \leftarrow (1+\rho)\star(S-T)\star(1-\rho)\},$
  div_cancel $\{x \leftarrow (1+\rho),\ y \leftarrow (S-T)\star(1-\rho)\},$
  rho_0,
  rate_simplify_step

RATE_2_simplify_pr: **Prove** RATE_2_simplify **from** RATE_2, rate_simplify

iclock_lem_pr: **Prove** iclock_lem **from**
  iclock_defn, IClock_defn $\{t \leftarrow ic_p^i(T)\}$, clock_ax $\{T \leftarrow T - adj_p^i\}$

IClock_ADJ_lem_pr: **Prove** IClock_ADJ_lem **from**
  IClock_defn, IClock_defn $\{i \leftarrow i+1\}$, $ADJ_{\star 1}^{\star 2}$

iclock_ADJ_lem_pr: **Prove** iclock_ADJ_lem **from**
  iclock_defn $\{T \leftarrow T - ADJ_p^i\}$, iclock_defn $\{i \leftarrow i+1\}$, $ADJ_{\star 1}^{\star 2}$

ADJ_lem1_pr: **Prove** ADJ_lem1 **from**
  ADJ_lem2,
  translation_invariance $\{X \leftarrow -IC_p^i(t_p^{i+1}),\ \gamma \leftarrow \Theta_p^{i+1}\}$

ADJ_lem2_pr: **Prove** ADJ_lem2 **from**
  $ADJ_{\star 1}^{\star 2}$,
  $adj_{\star 1}^{\star 2}$ $\{i \leftarrow i+1\},$
  IClock_defn $\{t \leftarrow t_p^{i+1},\ i \leftarrow i\},$
  correct_during_hi $\{t \leftarrow s_p^i,\ s \leftarrow t_p^{i+1}\}$

End delay

delay2: **Module**

Using arith, clockassumptions, delay

Exporting all with clockassumptions, delay

Theory

p, q, p_1, q_1: **Var** process
i: **Var** event
delay_pred: function[event $\rightarrow$ bool] =
$(\lambda\, i : (\forall\, p, q : \text{wpred}(i)(p) \wedge \text{wpred}(i)(q) \supset |s_p^i - s_q^i| \leq \beta'))$
ADJ_pred: function[event $\rightarrow$ bool] =
$(\lambda\, i : (\forall\, p : i \geq 1 \wedge \text{wpred}(i-1)(p) \supset |ADJ_p^{i-1}| \leq \alpha(\beta' + 2 * \Lambda')))$

delay_pred_lr: **Lemma**
delay_pred$(i) \supset (\text{wpred}(i)(p) \wedge \text{wpred}(i)(q) \supset |s_p^i - s_q^i| \leq \beta')$

bnd_delay_offset: **Theorem** ADJ_pred$(i) \wedge$ delay_pred(i)

bnd_delay_offset_0: **Lemma** ADJ_pred$(0) \wedge$ delay_pred(0)

bnd_delay_offset_ind: **Lemma**
ADJ_pred$(i) \wedge$ delay_pred$(i) \supset$ ADJ_pred$(i+1) \wedge$ delay_pred$(i+1)$

bnd_delay_offset_ind_a: **Lemma** delay_pred$(i) \supset$ ADJ_pred$(i+1)$

bnd_delay_offset_ind_b: **Lemma**
delay_pred$(i) \wedge$ ADJ_pred$(i+1) \supset$ delay_pred$(i+1)$

good_ReadClock: **Lemma**
delay_pred$(i) \wedge \text{wpred}(i)(p) \supset$ okay_Readpred$(\Theta_p^{i+1}, \beta' + 2 * \Lambda', \text{wpred}(i))$

delay_prec_enh: **Lemma**
delay_pred$(i) \wedge \text{wpred}(i)(p) \wedge \text{wpred}(i)(q)$
$\supset |(s_p^i - s_q^i) - (ADJ_p^i - ADJ_q^i)| \leq \pi(2 * \Lambda' + 2, \beta' + 2 * \Lambda')$

delay_prec_enh_step1: **Lemma**
delay_pred$(i) \wedge \text{wpred}(i)(p) \wedge \text{wpred}(i)(q)$
$\supset |cfn(p, (\lambda\, p_1 : \Theta_p^{i+1}(p_1) - IC_p^i(t_p^{i+1}) - \lfloor s_p^i \rfloor))$
$- cfn(q, (\lambda\, p_1 : \Theta_q^{i+1}(p_1) - IC_q^i(t_q^{i+1}) - \lceil s_q^i \rceil))|$
$\leq \pi(2 * \Lambda' + 2, \beta' + 2 * \Lambda')$

delay_prec_enh_step1_sym: **Lemma**
delay_pred$(i) \wedge \text{wpred}(i)(p) \wedge \text{wpred}(i)(q) \wedge (ADJ_p^i - s_p^i \geq ADJ_q^i - s_q^i)$
$\supset |(ADJ_p^i - s_p^i) - (ADJ_q^i - s_q^i)|$
$\leq |cfn(p, (\lambda\, p_1 : \Theta_p^{i+1}(p_1) - IC_p^i(t_p^{i+1}) - \lfloor s_p^i \rfloor))$
$- cfn(q, (\lambda\, p_1 : \Theta_q^{i+1}(p_1) - IC_q^i(t_q^{i+1}) - \lceil s_q^i \rceil))|$

prec_enh_hyp1: **Lemma**

$$\begin{aligned}&\mathsf{delay\_pred}(i) \wedge \mathsf{wpred}(i)(p) \wedge \mathsf{wpred}(i)(q)\\ &\quad\supset \mathsf{okay\_pairs}((\,\lambda\, p_1 : \Theta_p^{i+1}(p_1) - IC_p^i(t_p^{i+1}) - \lfloor s_p^i \rfloor),\\ &\qquad (\,\lambda\, p_1 : \Theta_q^{i+1}(p_1) - IC_q^i(t_q^{i+1}) - \lceil s_q^i \rceil),\\ &\qquad 2 * \Lambda' + 2,\\ &\qquad \mathsf{wpred}(i))\end{aligned}$$

prec_enh_hyp_2: **Lemma**

$$\begin{aligned}&\mathsf{delay\_pred}(i) \wedge \mathsf{wpred}(i)(p)\\ &\quad\supset \mathsf{okay\_Readpred}((\,\lambda\, p_1 : \Theta_p^{i+1}(p_1) - IC_p^i(t_p^{i+1}) - \lfloor s_p^i \rfloor),\\ &\qquad \beta' + 2 * \Lambda',\\ &\qquad \mathsf{wpred}(i))\end{aligned}$$

prec_enh_hyp_3: **Lemma**

$$\begin{aligned}&\mathsf{delay\_pred}(i) \wedge \mathsf{wpred}(i)(q)\\ &\quad\supset \mathsf{okay\_Readpred}((\,\lambda\, p_1 : \Theta_q^{i+1}(p_1) - IC_q^i(t_q^{i+1}) - \lceil s_q^i \rceil),\\ &\qquad \beta' + 2 * \Lambda',\\ &\qquad \mathsf{wpred}(i))\end{aligned}$$

Proof

delay_pred_lr_pr: **Prove** delay_pred_lr **from** delay_pred

delay_prec_enh_step1_pr: **Prove** delay_prec_enh_step1 **from**

precision_enhancement_ax

$$\begin{aligned}&\{\mathsf{ppred} \leftarrow \mathsf{wpred}(i),\\ &\ y \leftarrow \beta' + 2 * \Lambda',\\ &\ x \leftarrow 2 * \Lambda' + 2,\\ &\ \gamma \leftarrow (\,\lambda\, p_1 : \Theta_p^{i+1}(p_1) - IC_p^i(t_p^{i+1}) - \lfloor s_p^i \rfloor),\\ &\ \theta \leftarrow (\,\lambda\, p_1 : \Theta_q^{i+1}(p_1) - IC_q^i(t_q^{i+1}) - \lceil s_q^i \rceil)\},\end{aligned}$$

prec_enh_hyp1,
prec_enh_hyp_2,
prec_enh_hyp_3,
wpred_ax

prec_enh_hyp_2_pr: **Prove** prec_enh_hyp_2 **from**

good_ReadClock,
okay_Readpred

$$\begin{aligned}&\{\gamma \leftarrow (\,\lambda\, p_1 : \Theta_p^{i+1}(p_1) - IC_p^i(t_p^{i+1}) - \lfloor s_p^i \rfloor),\\ &\ y \leftarrow \beta' + 2 * \Lambda',\\ &\ \mathsf{ppred} \leftarrow \mathsf{wpred}(i)\},\end{aligned}$$

okay_Readpred

$$\begin{aligned}&\{\gamma \leftarrow \Theta_p^{i+1},\\ &\ y \leftarrow \beta' + 2 * \Lambda',\\ &\ \mathsf{ppred} \leftarrow \mathsf{wpred}(i),\\ &\ l \leftarrow l@p2,\\ &\ m \leftarrow m@p2\}\end{aligned}$$

prec_enh_hyp_3_pr: **Prove** prec_enh_hyp_3 **from**
good_ReadClock $\{p \leftarrow q\}$,
okay_Readpred
$\{\gamma \leftarrow (\lambda\, p_1 : \Theta_q^{i+1}(p_1) - IC_q^i(t_q^{i+1}) - \lceil s_q^i \rceil)$,
$y \leftarrow \beta' + 2 * \Lambda'$,
ppred $\leftarrow$ wpred$(i)\}$,
okay_Readpred
$\{\gamma \leftarrow \Theta_q^{i+1}$,
$y \leftarrow \beta' + 2 * \Lambda'$,
ppred $\leftarrow$ wpred(i),
$l \leftarrow l@p2$,
$m \leftarrow m@p2\}$

bnd_del_off_0_pr: **Prove** bnd_delay_offset_0 **from**
ADJ_pred $\{i \leftarrow 0\}$,
delay_pred $\{i \leftarrow 0\}$,
bnd_delay_init $\{p \leftarrow p@p2,\ q \leftarrow q@p2\}$

bnd_delay_offset_ind_pr: **Prove** bnd_delay_offset_ind **from**
bnd_delay_offset_ind_a, bnd_delay_offset_ind_b

bnd_delay_offset_pr: **Prove** bnd_delay_offset **from**
induction {prop $\leftarrow$ ($\lambda\, i$: ADJ_pred$(i) \wedge$ delay_pred(i))},
bnd_delay_offset_0,
bnd_delay_offset_ind $\{i \leftarrow j@p1\}$

a, b, c, d, e, f, g, h: **Var** number

abs_hack: **Lemma** $|a - b|$
$\leq |e - f| + |(a - c) - (d - e)| + |(b - c) - (d - f)|$

abs_hack_pr: **Prove** abs_hack **from**
abs_com $\{x \leftarrow f,\ y \leftarrow e\}$,
abs_com $\{x \leftarrow (d - f),\ y \leftarrow (b - c)\}$,
abs_plus
$\{x \leftarrow (f - e)$,
$y \leftarrow ((a - c) - (d - e)) + ((d - f) - (b - c))\}$,
abs_plus $\{x \leftarrow ((a - c) - (d - e)),\ y \leftarrow ((d - f) - (b - c))\}$

abshack2: **Lemma** $|a| \leq b \wedge |c| \leq d \wedge |e| \leq d \supset |a| + |c| + |e| \leq b + 2 * d$

abshack2_pr: **Prove** abshack2

good_ReadClock_pr: **Prove** good_ReadClock **from**

okay_Readpred

$\{\gamma \leftarrow \Theta_p^{i+1},$

$y \leftarrow \beta' + 2 * \Lambda',$

ppred $\leftarrow$ wpred$(i)\}$,

delay_pred $\{p \leftarrow l@p1,\ q \leftarrow m@p1\}$,

delay_pred $\{q \leftarrow l@p1\}$,

delay_pred $\{q \leftarrow m@p1\}$,

reading_error3 $\{q \leftarrow l@p1\}$,

reading_error3 $\{q \leftarrow m@p1\}$,

abs_hack

$\{a \leftarrow \Theta_p^{i+1}(l@p1),$

$b \leftarrow \Theta_p^{i+1}(m@p1),$

$c \leftarrow IC_p^i(t_p^{i+1}),$

$d \leftarrow s_p^i,$

$e \leftarrow s_{l@p1}^i,$

$f \leftarrow s_{m@p1}^i\},$

abshack2

$\{a \leftarrow e@p7 - f@p7,$

$b \leftarrow \beta',$

$c \leftarrow ((a@p7 - c@p7) - (d@p7 - e@p7)),$

$d \leftarrow \Lambda',$

$e \leftarrow ((b@p7 - c@p7) - (d@p7 - f@p7))\},$

wpred_fixtime,

wpred_fixtime $\{p \leftarrow l@p1\}$,

wpred_fixtime $\{p \leftarrow m@p1\}$

bnd_del_off_ind_a_pr: **Prove** bnd_delay_offset_ind_a **from**

ADJ_pred $\{i \leftarrow i + 1\}$,

ADJ_lem2 $\{p \leftarrow p@p1\}$,

accuracy_preservation_ax

$\{$ppred $\leftarrow$ wpred(i),

$\gamma \leftarrow \Theta_{p@p1}^{i+1},$

$p \leftarrow p@p1,$

$q \leftarrow p@p1,$

$x \leftarrow \beta' + 2 * \Lambda'\},$

wpred_ax,

read_self $\{p \leftarrow p@p1\}$,

good_ReadClock $\{p \leftarrow p@p1\}$,

wpred_fixtime $\{p \leftarrow p@p1\}$

abshack4: **Lemma** $a - b \geq c - d$

$\supset |(a - b) - (c - d)| \leq |(a - \lfloor b \rfloor) - (c - \lceil d \rceil)|$

floor_hack: **Lemma** $a - \lfloor b \rfloor \geq a - b$

floor_hack_pr: **Prove** floor_hack **from** floor_defn $\{x \leftarrow b\}$

ceil_hack: **Lemma** $c - d \geq c - \lceil d \rceil$

ceil_hack_pr: **Prove** ceil_hack **from** ceil_defn $\{x \leftarrow d\}$

abshack4_pr: **Prove** abshack4 **from**
abs_ge0 $\{x \leftarrow (a - b) - (c - d)\}$,
abs_ge0 $\{x \leftarrow (a - \lfloor b \rfloor) - (c - \lceil d \rceil)\}$,
floor_hack,
ceil_hack

X: **Var** Clocktime

ADJ_hack: **Lemma** wpred$(i)(p)$
$\supset ADJ_p^i - X = cfn(p, (\lambda p_1 : \Theta_p^{i+1}(p_1) - IC_p^i(t_p^{i+1}) - X))$

ADJ_hack_pr: **Prove** ADJ_hack **from**
ADJ_lem1,
translation_invariance
$\{\gamma \leftarrow (\lambda p_1 \rightarrow \text{Clocktime} : \Theta_p^{i+1}(p_1) - IC_p^i(t_p^{i+1})),$
$X \leftarrow -X\}$,
wpred_fixtime

delay_prec_enh_step1_sym_pr: **Prove** delay_prec_enh_step1_sym **from**
ADJ_hack $\{X \leftarrow \lfloor s_p^i \rfloor\}$,
ADJ_hack $\{p \leftarrow q,\ X \leftarrow \lceil s_q^i \rceil\}$,
abshack4 $\{a \leftarrow ADJ_p^i,\ b \leftarrow s_p^i,\ c \leftarrow ADJ_q^i,\ d \leftarrow s_q^i\}$

abshack5: **Lemma** $|((a - b) - (\lfloor c \rfloor - d)) - ((e - f) - (\lceil g \rceil - d))|$
$\leq |(a - b) - (\lfloor c \rfloor - d)| + |(e - f) - (\lceil g \rceil - d)|$

abshack5_pr: **Prove** abshack5 **from**
abs_com $\{x \leftarrow e - f,\ y \leftarrow \lceil g \rceil - d\}$,
abs_plus $\{x \leftarrow (a - b) - (\lfloor c \rfloor - d),\ y \leftarrow (\lceil g \rceil - d) - (e - f)\}$

absfloor: **Lemma** $|a - \lfloor b \rfloor| \leq |a - b| + 1$

absceil: **Lemma** $|a - \lceil b \rceil| \leq |a - b| + 1$

absfloor_pr: **Prove** absfloor **from**
floor_defn $\{x \leftarrow b\}$, $|\star 1|$ $\{x \leftarrow a - \lfloor b \rfloor\}$, $|\star 1|$ $\{x \leftarrow a - b\}$

absceil_pr: **Prove** absceil **from**
ceil_defn $\{x \leftarrow b\}$, $|\star 1|$ $\{x \leftarrow a - \lceil b \rceil\}$, $|\star 1|$ $\{x \leftarrow a - b\}$

abshack6a: **Lemma** $|(a - b) - (\lfloor c \rfloor - d)| \leq |(a - b) - (c - d)| + 1$

abshack6b: **Lemma** $|(e - f) - (\lceil g \rceil - d)| \leq |(e - f) - (g - d)| + 1$

abshack6a_pr: **Prove** abshack6a **from**
absfloor $\{a \leftarrow (a-b)+d,\ b \leftarrow c\}$,
abs_plus $\{x \leftarrow (a-b)-(c-d),\ y \leftarrow 1\}$,
abs_ge0 $\{x \leftarrow 1\}$

abshack6b_pr: **Prove** abshack6b **from**
absceil $\{a \leftarrow (e-f)+d,\ b \leftarrow g\}$,
abs_plus $\{x \leftarrow (e-f)-(g-d),\ y \leftarrow 1\}$,
abs_ge0 $\{x \leftarrow 1\}$

abshack7: **Lemma** $|(a-b)-(c-d)| \le h \wedge |(e-f)-(g-d)| \le h$
$\supset |((a-b)-(\lfloor c \rfloor - d)) - ((e-f)-(\lceil g \rceil - d))| \le 2*(h+1)$

abshack7_pr: **Prove** abshack7 **from** abshack5, abshack6a, abshack6b

prec_enh_hyp1_pr: **Prove** prec_enh_hyp1 **from**
okay_pairs
$\{\gamma \leftarrow (\lambda\, p_1 : \Theta_p^{i+1}(p_1) - IC_p^i(t_p^{i+1}) - \lfloor s_p^i \rfloor)$,
$\theta \leftarrow (\lambda\, p_1 : \Theta_q^{i+1}(p_1) - IC_q^i(t_q^{i+1}) - \lceil s_q^i \rceil)$,
$x \leftarrow 2*(\Lambda'+1)$,
ppred $\leftarrow$ wpred(i)},
delay_pred $\{q \leftarrow p_3$@p1$\}$,
delay_pred $\{p \leftarrow q,\ q \leftarrow p_3$@p1$\}$,
reading_error3 $\{q \leftarrow p_3$@p1$\}$,
reading_error3 $\{p \leftarrow q,\ q \leftarrow p_3$@p1$\}$,
abshack7
$\{a \leftarrow \Theta_p^{i+1}(p_3$@p1$)$,
$b \leftarrow IC_p^i(t_p^{i+1})$,
$c \leftarrow s_p^i$,
$d \leftarrow s^i_{p_3@p1}$,
$e \leftarrow \Theta_q^{i+1}(p_3$@p1$)$,
$f \leftarrow IC_q^i(t_q^{i+1})$,
$g \leftarrow s_q^i$,
$h \leftarrow \Lambda'\}$,
wpred_fixtime,
wpred_fixtime $\{p \leftarrow q\}$,
wpred_fixtime $\{p \leftarrow p_3$@p1$\}$

abshack3: **Lemma** $|(a-b)-(c-d)| = |(c-a)-(d-b)|$

abshack3_pr: **Prove** abshack3 **from** abs_com $\{x \leftarrow a-b,\ y \leftarrow c-d\}$

delay_prec_enh_pr: **Prove** delay_prec_enh **from**
delay_prec_enh_step1,
delay_prec_enh_step1 $\{p \leftarrow q, \; q \leftarrow p\}$,
delay_prec_enh_step1_sym,
delay_prec_enh_step1_sym $\{p \leftarrow q, \; q \leftarrow p\}$,
abs_com $\{x \leftarrow ADJ_p^i - s_p^i, \; y \leftarrow ADJ_q^i - s_q^i\}$,
abshack3 $\{a \leftarrow s_p^i, \; b \leftarrow s_q^i, \; c \leftarrow ADJ_p^i, \; d \leftarrow ADJ_q^i\}$

End delay2

delay3: **Module**

Using arith, clockassumptions, delay2

Exporting all with clockassumptions, delay2

Theory

p, q, p_1, q_1: **Var** process
i: **Var** event
T: **Var** Clocktime
good_interval: function[process, event, Clocktime → bool] =
$(\lambda\, p, i, T : (\text{correct\_during}(p, s_p^i, ic_p^{i+1}(T)) \wedge T - ADJ_p^i \geq S^i)$
$\vee\, (\text{correct\_during}(p, ic_p^{i+1}(T), s_p^i) \wedge S^i \geq T - ADJ_p^i))$

recovery_lemma: **Axiom**
delay_pred$(i) \wedge$ ADJ_pred$(i+1)$
$\wedge$ rpred$(i)(p) \wedge$ correct_during$(p, t_p^{i+1}, t_p^{i+2}) \wedge$ wpred$(i+1)(q)$
$\supset |s_p^{i+1} - s_q^{i+1}| \leq \beta'$

good_interval_lem: **Lemma**
wpred$(i)(p) \wedge$ wpred$(i+1)(p) \wedge$ ADJ_pred$(i+1) \supset$ good_interval(p, i, S^{i+1})

betaprime_ax: **Axiom**
$2 * \rho \star (R + \alpha(\beta' + 2 * \Lambda')) + \pi(2 * (\Lambda' + 1), \beta' + 2 * \Lambda') \leq \beta'$

R_0_lem: **Lemma** wpred$(i)(p) \wedge$ ADJ_pred$(i+1) \supset R > 0$

bound_future: **Lemma**
delay_pred$(i) \wedge$ ADJ_pred$(i+1)$
$\wedge$ wpred$(i)(p)$
$\wedge$ wpred$(i)(q) \wedge$ good_interval$(p, i, T) \wedge$ good_interval(q, i, T)
$\supset |ic_p^{i+1}(T) - ic_q^{i+1}(T)|$
$\leq 2 * \rho \star (|T - S^i| + \alpha(\beta' + 2 * \Lambda')) + \pi(2 * (\Lambda' + 1), \beta' + 2 * \Lambda')$

bound_future1: **Lemma**
delay_pred$(i) \wedge$ ADJ_pred$(i+1) \wedge$ wpred$(i)(p) \wedge$ good_interval(p, i, T)
$\supset |(ic_p^i(T - ADJ_p^i) - s_p^i) - (T - ADJ_p^i - S^i)|$
$\leq \rho \star (|T - S^i| + \alpha(\beta' + 2 * \Lambda'))$

bound_future1_step: **Lemma**
delay_pred$(i) \wedge$ ADJ_pred$(i+1) \wedge$ wpred$(i)(p) \wedge$ good_interval(p, i, T)
$\supset |(ic_p^i(T - ADJ_p^i) - s_p^i) - (T - ADJ_p^i - S^i)| \leq \rho \star (|T - ADJ_p^i - S^i|)$

bound_FIXTIME: **Lemma**
delay_pred$(i) \wedge$ ADJ_pred$(i+1)$
$\wedge$ wpred$(i)(p)$
$\wedge$ wpred$(i)(q)$
$\wedge$ good_interval$(p, i, S^{i+1}) \wedge$ good_interval(q, i, S^{i+1})
$\supset |s_p^{i+1} - s_q^{i+1}| \leq \beta'$

bound_FIXTIME2: **Lemma**
delay_pred(i) $\wedge$ ADJ_pred($i+1$) $\wedge$ wpred(i)(p) $\wedge$ wpred(i)(q)
$\supset$ (wpred($i+1$)(p) $\wedge$ wpred($i+1$)(q) $\supset |s_p^{i+1} - s_q^{i+1}| \leq \beta'$)

delay_offset: **Lemma** wpred(i)(p) $\wedge$ wpred(i)(q) $\supset |s_p^i - s_q^i| \leq \beta'$

ADJ_bound: **Lemma** wpred(i)(p) $\supset |ADJ_p^i| \leq \alpha(\beta' + 2 * \Lambda')$

Alpha_0: **Lemma** wpred(i)(p) $\supset \alpha(\beta' + 2 * \Lambda') \geq 0$

Proof

delay_offset_pr: **Prove** delay_offset **from** bnd_delay_offset, delay_pred

ADJ_bound_pr: **Prove** ADJ_bound **from**
bnd_delay_offset $\{i \leftarrow i+1\}$, ADJ_pred $\{i \leftarrow i+1\}$

a_1, b_1, c_1, d_1: **Var** number

abs_0: **Lemma** $|a_1| \leq b_1 \supset b_1 \geq 0$

abs_0_pr: **Prove** abs_0 **from** $|\star 1|$ $\{x \leftarrow a_1\}$

Alpha_0_pr: **Prove** Alpha_0 **from**
ADJ_bound, abs_0 $\{a_1 \leftarrow ADJ_p^i,\ b_1 \leftarrow \alpha(\beta' + 2 * \Lambda')\}$

R_0_hack: **Lemma** wpred(i)(p) $\wedge$ ADJ_pred($i+1$) $\supset S^{i+1} - S^i > 0$

R_0_hack_pr: **Prove** R_0_hack **from**
ADJ_pred $\{i \leftarrow i+1\}$,
FIXTIME_bound,
wpred_hi_lem,
abs_0 $\{a_1 \leftarrow ADJ_p^i,\ b_1 \leftarrow \alpha(\beta' + 2 * \Lambda')\}$

R_0_lem_pr: **Prove** R_0_lem **from** R_0_hack, $S^{\star 1}$, $S^{\star 1}$ $\{i \leftarrow i+1\}$

abshack_future: **Lemma** $|(a_1 - b_1) - (c_1 - d_1)| = |(a_1 - c_1) - (b_1 - d_1)|$

abshack_future_pr: **Prove** abshack_future

abs_minus: **Lemma** $|a_1 - b_1| \leq |a_1| + |b_1|$

abs_minus_pr: **Prove** abs_minus **from**
$|\star 1|$ $\{x \leftarrow a_1 - b_1\}$, $|\star 1|$ $\{x \leftarrow a_1\}$, $|\star 1|$ $\{x \leftarrow b_1\}$

bound_future1_pr: **Prove** bound_future1 **from**
  bound_future1_step,
  abs_minus $\{a_1 \leftarrow T - S^i,\ b_1 \leftarrow ADJ_p^i\}$,
  ADJ_pred $\{i \leftarrow i + 1\}$,
  mult_leq_2
    $\{z \leftarrow \rho,$
    $y \leftarrow |T - ADJ_p^i - S^i|,$
    $x \leftarrow |T - S^i| + \alpha(\beta' + 2 * \Lambda')\}$,
  rho_0

bound_future1_step_a: **Lemma**
  correct_during$(p, ic_p^i(T - ADJ_p^i), s_p^i) \wedge S^i \geq T - ADJ_p^i$
    $\supset |(ic_p^i(T - ADJ_p^i) - s_p^i) - (T - ADJ_p^i - S^i)| \leq \rho \star (|T - ADJ_p^i - S^i|)$

bound_future1_step_b: **Lemma**
  correct_during$(p, s_p^i, ic_p^i(T - ADJ_p^i)) \wedge T - ADJ_p^i \geq S^i$
    $\supset |(ic_p^i(T - ADJ_p^i) - s_p^i) - (T - ADJ_p^i - S^i)| \leq \rho \star (|T - ADJ_p^i - S^i|)$

bound_future1_step_a_pr: **Prove** bound_future1_step_a **from**
  RATE_lemma2_iclock $\{T \leftarrow T - ADJ_p^i,\ S \leftarrow S^i\}$,
  $s_{\star 1}^{\star 2}$,
  abshack_future
    $\{a_1 \leftarrow ic_p^i(T - ADJ_p^i),$
    $b_1 \leftarrow s_p^i,$
    $c_1 \leftarrow T - ADJ_p^i,$
    $d_1 \leftarrow S^i\}$,
  abs_com $\{x \leftarrow a_1@p3 - c_1@p3,\ y \leftarrow b_1@p3 - d_1@p3\}$,
  abs_com $\{x \leftarrow T@p1,\ y \leftarrow S@p1\}$

bound_future1_step_b_pr: **Prove** bound_future1_step_b **from**
  RATE_lemma2_iclock $\{S \leftarrow T - ADJ_p^i,\ T \leftarrow S^i\}$,
  $s_{\star 1}^{\star 2}$,
  abshack_future
    $\{a_1 \leftarrow ic_p^i(T - ADJ_p^i),$
    $b_1 \leftarrow s_p^i,$
    $c_1 \leftarrow T - ADJ_p^i,$
    $d_1 \leftarrow S^i\}$

bound_future1_step_pr: **Prove** bound_future1_step **from**
  good_interval, bound_future1_step_a, bound_future1_step_b, iclock_ADJ_lem

good_interval_lem_pr: **Prove** good_interval_lem **from**
good_interval $\{T \leftarrow S^{i+1}\}$,
$s_{\star 1}^{\star 2}$ $\{i \leftarrow i+1\}$,
wpred_fixtime,
wpred_fixtime_low $\{i \leftarrow i+1\}$,
correct_during_trans $\{t \leftarrow s_p^i,\ t_2 \leftarrow t_p^{i+1},\ s \leftarrow s_p^{i+1}\}$,
wpred_hi_lem,
FIXTIME_bound,
ADJ_pred $\{i \leftarrow i+1\}$,
$|\star 1|$ $\{x \leftarrow ADJ_p^i\}$

bound_FIXTIME2_pr: **Prove** bound_FIXTIME2 **from**
bound_FIXTIME, good_interval_lem, good_interval_lem $\{p \leftarrow q\}$

bound_FIXTIME_pr: **Prove** bound_FIXTIME **from**
bound_future $\{T \leftarrow S^{i+1}\}$,
$S^{\star 1}$,
$S^{\star 1}$ $\{i \leftarrow i+1\}$,
abs_ge0 $\{x \leftarrow R\}$,
R_0_lem,
$s_{\star 1}^{\star 2}$ $\{p \leftarrow p@p1,\ i \leftarrow i+1\}$,
$s_{\star 1}^{\star 2}$ $\{p \leftarrow q@p1,\ i \leftarrow i+1\}$,
betaprime_ax

bnd_delay_offset_ind_b_pr: **Prove** bnd_delay_offset_ind_b **from**
bound_FIXTIME2 $\{p \leftarrow p@p2,\ q \leftarrow q@p2\}$,
delay_pred $\{i \leftarrow i+1\}$,
delay_pred $\{p \leftarrow p@p2,\ q \leftarrow q@p2\}$,
recovery_lemma $\{p \leftarrow p@p2,\ q \leftarrow q@p2\}$,
recovery_lemma $\{p \leftarrow q@p2,\ q \leftarrow p@p2\}$,
abs_com $\{x \leftarrow s_{p@p2}^{i+1},\ y \leftarrow s_{q@p2}^{i+1}\}$,
wpred_preceding $\{p \leftarrow p@p2\}$,
wpred_preceding $\{p \leftarrow q@p2\}$,
wpred_correct $\{i \leftarrow i+1,\ p \leftarrow p@p2\}$,
wpred_correct $\{i \leftarrow i+1,\ p \leftarrow q@p2\}$

a, b, c, d, e, f, g, h, aa, bb: **Var** number

abshack: **Lemma** $|a-b|$
$\leq |(a-e)-(c-f-d)| + |(b-g)-(c-h-d)|$
$+ |(e-g)-(f-h)|$

abshack2: **Lemma** $|(a-e)-(c-f-d)| \leq$ aa
$\wedge\ |(b-g)-(c-h-d)| \leq$ aa $\wedge\ |(e-g)-(f-h)| \leq$ bb
$\supset |a-b| \leq 2 *$ aa + bb

abshack2_pr: **Prove** abshack2 **from** abshack

abshack_pr: **Prove** abshack **from**

abs_com $\{x \leftarrow b - g,\ y \leftarrow c - h - d\}$,

abs_plus $\{x \leftarrow (a - e) - (c - f - d),\ y \leftarrow (c - h - d) - (b - g)\}$,

abs_plus $\{x \leftarrow x@p2 + y@p2,\ y \leftarrow (e - g) - (f - h)\}$

bound_future_pr: **Prove** bound_future **from**

bound_future1,

bound_future1 $\{p \leftarrow q\}$,

delay_prec_enh,

iclock_ADJ_lem,

iclock_ADJ_lem $\{p \leftarrow q\}$,

abshack2

$\{a \leftarrow ic_p^i(T - ADJ_p^i)$,

$b \leftarrow ic_q^i(T - ADJ_q^i)$,

$c \leftarrow T$,

$d \leftarrow S^i$,

$e \leftarrow s_p^i$,

$f \leftarrow ADJ_p^i$,

$g \leftarrow s_q^i$,

$h \leftarrow ADJ_q^i$,

aa $\leftarrow \rho \star (|T - S^i| + \alpha(\beta' + 2 * \Lambda'))$,

bb $\leftarrow \pi(2 * (\Lambda' + 1), \beta' + 2 * \Lambda')\}$

End delay3

delay4: **Module**

Using arith, clockassumptions, delay3

Exporting all with clockassumptions, delay3

Theory

p, q, p_1, q_1: **Var** process
i: **Var** event
X, S, T: **Var** Clocktime
s, t, t_1, t_2: **Var** time
γ: **Var** function[process $\rightarrow$ Clocktime]
ppred, ppred1: **Var** function[process $\rightarrow$ bool]
option1, option2: bool

option1_alg: **Axiom** option1 $\supset T_p^{i+1} = (i+1) * R + T^0$

option2_alg: **Axiom** option2 $\supset T_p^{i+1} = (i+1) * R + T^0 - ADJ_p^i$

options_disjoint: **Axiom** $\neg$(option1 $\wedge$ option2)

option1_bounded_delay: **Lemma**
option1 $\wedge$ $(\beta = 2 * \rho \star (R - (S^0 - T^0)) + \beta')$ $\wedge$ wpred(i)(p) $\wedge$ wpred(i)(q)
$\supset |t_p^{i+1} - t_q^{i+1}| \leq \beta$

option2_bounded_delay: **Lemma**
option2 $\wedge$ $(\beta = \beta' - 2 * \rho \star (S^0 - T^0))$ $\wedge$ wpred(i)(p) $\wedge$ wpred(i)(q)
$\supset |t_p^{i+1} - t_q^{i+1}| \leq \beta$

option2_convert_lemma: **Lemma**
$(\beta = \beta' - 2 * \rho \star (S^0 - T^0))$
$\supset 2 * \rho \star ((R - (S^0 - T^0)) + \alpha(\beta' + 2 * \Lambda'))$
$+ \pi(2 * (\Lambda' + 1), \beta' + 2 * \Lambda')$
$\leq \beta$

option2_good_interval: **Lemma**
option2 $\wedge$ wpred(i)(p) $\supset$ good_interval($p, i, (i+1) * R + T^0$)

R_FIX_SYNC_0: **Axiom** $R - (S^0 - T^0) > 0$

Proof

option1_bounded_delay_pr: **Prove** option1_bounded_delay **from**
RATE_lemma1_iclock $\{S \leftarrow (i+1) * R + T^0,\ T \leftarrow S^i\}$,
$S^{\star 1}$,
delay_offset,
wpred_fixtime,
wpred_fixtime $\{p \leftarrow q\}$,
synctime_defn,
synctime_defn $\{p \leftarrow q\}$,
$s^{\star 2}_{\star 1}$,
$s^{\star 2}_{\star 1}$ $\{p \leftarrow q\}$,
option1_alg,
option1_alg $\{p \leftarrow q\}$,
R_FIX_SYNC_0

option2_good_interval_pr: **Prove** option2_good_interval **from**
good_interval $\{T \leftarrow T_p^{i+1} + ADJ_p^i\}$,
wpred_fixtime,
wpred_hi_lem,
rts_new_1,
iclock_ADJ_lem $\{T \leftarrow T@p1\}$,
synctime_defn,
Alpha_0,
option2_alg

option2_convert_lemma_pr: **Prove** option2_convert_lemma **from**
betaprime_ax,
mult_ldistrib_minus
$\{x \leftarrow \rho,$
$y \leftarrow R + \alpha(\beta' + 2 * \Lambda'),$
$z \leftarrow (S^0 - T^0)\}$

option2_bounded_delay_pr: **Prove** option2_bounded_delay **from**
option2_convert_lemma,
option2_good_interval,
option2_good_interval $\{p \leftarrow q\}$,
bound_future $\{T \leftarrow (i+1) * R + T^0\}$,
option2_alg,
option2_alg $\{p \leftarrow q\}$,
iclock_ADJ_lem $\{T \leftarrow T@p4\}$,
iclock_ADJ_lem $\{T \leftarrow T@p4,\ p \leftarrow q\}$,
synctime_defn,
synctime_defn $\{p \leftarrow q\}$,
$S^{\star 1}$,
R_0_lem,
bnd_delay_offset,
bnd_delay_offset $\{i \leftarrow i + 1\}$,
abs_ge0 $\{x \leftarrow (R - (S^0 - T^0))\}$,
R_FIX_SYNC_0

End delay4

References

[1] Schneider, Fred B.: *Understanding Protocols for Byzantine Clock Synchronization.* Department of Computer Science, Cornell University, Technical Report 87-859, Ithaca, NY, Aug. 1987.

[2] Shankar, Natarajan: *Mechanical Verification of a Schematic Byzantine Clock Synchronization Algorithm.* NASA, Contractor Report 4386, July 1991.

[3] Welch, J. Lundelius; and Lynch, N.: A New Fault-Tolerant Algorithm for Clock Synchronization. *Information and Computation*, vol. 77, no. 1, Apr. 1988, pp. 1–36.

[4] Rushby, John; and von Henke, Friedrich: *Formal Verification of a Fault Tolerant Clock Synchronization Algorithm.* NASA, Contractor Report 4239, June 1989.

[5] Lamport, Leslie; and Melliar-Smith, P.M.: Synchronizing Clocks in the Presence of Faults. *Journal of the ACM*, vol. 21, Jan. 1985, pp. 52–78.

[6] Miner, Paul S.: *A Verified Design of a Fault-Tolerant Clock Synchronization Circuit: Preliminary Investigations.* NASA, Technical Memorandum 107568, Langley Research Center, Hampton, VA, Mar. 1992.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*

2. REPORT DATE
June 1992

3. REPORT TYPE AND DATES COVERED
Technical Memorandum

4. TITLE AND SUBTITLE
An Extension to Schneider's General Paradigm for Fault-Tolerant Clock Synchronization

5. FUNDING NUMBERS
WU 505-64-10-05

6. AUTHOR(S)
Paul S. Miner

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
NASA Langley Research Center
Hampton, VA 23665-5225

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
National Aeronautics and Space Administration
Washington, DC 20546-0001

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
NASA TM-107634

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Unclassified - Unlimited

Subject Category 62

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

In 1987, Schneider presented a general paradigm that provides a single proof of a number of fault-tolerant clock synchronization algorithms. His proof was subsequently subjected to the rigor of mechanical verification by Shankar. However, both Schneider and Shankar assumed a condition Shankar refers to as bounded delay. This condition states that the elapsed time between synchronization events (i.e. the time that the local process applies an adjustment to its logical clock) is bounded. This property is really a result of the algorithm and should not be assumed in a proof of correctness. This paper remedies this by providing a proof of this property in the context of the general paradigm proposed by Schneider. The argument given is a generalization of Welch and Lynch's proof of a related property for their algorithm.

14. SUBJECT TERMS
Fault Tolerance, Clock Synchronization, Formal Verification

15. NUMBER OF PAGES
55

16. PRICE CODE
A04

17. SECURITY CLASSIFICATION OF REPORT
Unclassified

18. SECURITY CLASSIFICATION OF THIS PAGE
Unclassified

19. SECURITY CLASSIFICATION OF ABSTRACT

20. LIMITATION OF ABSTRACT

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std Z39-18
298-102