The 50 MHz Clock is Not 50 MHz 1.0

J. Orrell, J. Wilkerson

June 21, 2001

1 Introduction

1.1 Purpose of the 50 MHz

The 50 MHz clock was originally intended to serve as the relative time basis for event-to-event timing. That is, detector global triggers are latched on the 20 nanosecond ticks of the 50 MHz clock. (Please see the *The SNO Trigger System* document for more information. SNO-STR-97-035) Due to the 43 bits allocated for the 50 MHz clock's record, the 50 MHz clock "rolls over" approximately every 2 days. The intention was to allow this to happen since the 10 MHz clock was to keep "absolute time" over the duration of the experiment. The 10 MHz clock is our absolute time because it is periodically synchronized to the Global Positioning System (GPS) time standard. For more information on the 10 MHz clock and GPS time please see the *GPS Verification of 10 MHz Time Stamps* document.

1.2 Potential of the 50 MHz

Certainly the 50 MHz clock can do more and there is ample reason to do so. Simple calculations tell us exactly how to account for 50 MHz clock's roll-over. All that is needed is the ability to correctly identify the point of roll-over. An initial comparison of the 50 MHz clock to the 10 MHz clock is provided by the *50 MHz Time* document. This document shows that we can hope to use the 50 MHz as a secondary time standard. A secondary time standard is extremely useful in situations where the 10 MHz clock has failed, the GPS communication is down or unavailable, or there are other uncertainties in the validity of the time stamps supplied by the 10 MHz clock.

1.3 Calibrating the 50 MHz

The 50 MHz clock is not synchronized to GPS or any other time standard. For this reason we need to confirm that the 50 MHz clock does indeed "tick" at 50 MHz. If the 50 MHz clock does not tick at 50 MHz, there is still no problem so long as we can determine the rate at which it does tick. The majority of the text of this document addresses the calculation of the actual rate of the 50 MHz clock.

1.4 Results

This paper reports the actual rate at which the 50 MHz clock runs and explains how the calculation was made. The 50 MHz clock does not run at precisely 50 MHz. This is not a surprise, nor is it a problem. In the end, we find that the average rate of the 50 MHz clock is

$$R_{50} = 49999473.2 \pm 0.4\text{Hz}. \tag{1}$$

Knowing this rate we can use the 50 MHz clock to make more accurate checks on the validity of the 10 MHz clock time when GPS is not available.

2 Time Information in SNO

2.1 50 MHz Records

The 50 MHz clock tick is stored in a 43 bit word. This 43 bit word can thus store 8796093022207 ticks. The 43 bits of this word correspond to a clock which can record sequential times up to 175921 seconds, 860444 microseconds, and 140 nanoseconds. This is equivalent to 2 days, 52 minutes, 1 second, 860444 microseconds, and 140 nanoseconds. The 50 MHz clock is termed to "roll-over", but in reality this roll-over is simply a limitation enforced by the finite size of our 43 bit storage allocation.

2.2 Extracting the 50 Mhz Record

There are two different ways to obtain 50 MHz time information. We have access to both the "raw", bit-packed 50 MHz clock ticks as well as the time elapsed. The bit-packed 50 MHz information is the number of ticks since the last 50 MHz roll-over. Likewise, the time elapsed is the amount of time since the last 50 MHz roll-over.

2.2.1 Extracting the Global Trigger Time

The Event Bank (see *The SNOMAN Companion*) supplies the time elapsed (from the last 50 MHz roll-over) for the 50 MHz clock. The following lines can be inserted into a ntuple definition to command SNOMAN to write out the 50 MHz record for any given event.

```
#.
'  gtr_gtr    ntime           EV+$KEV_GTR,0;              '
'  gtr_sc     reserved        ;                           '
'  gtr_usc    reserved        ;                           '
'  gtr_nsc    reserved        ;                           '
#.
```

The above four lines will output the negative of the event number, seconds, microseconds, and nanoseconds, respectively. This is referred to as the Global Trigger Time in *The SNOMAN Companion*.

2.2.2 Extracting the Bit-Packed Time

The ZDAB Bank (see *The SNOMAN Companion*) supplies the number of clock ticks (from the last 50 MHz roll-over) for the 50 MHz clock in a bit-packed form. The following lines can be inserted into a ntuple definition to command SNOMAN to write out the bit-packed 50 MHz record for any given event.

```
#.
'  z_50hi_2   bits            ZDAB+8,16,16;               '
'  z_50hi_1   bits            ZDAB+8,0,16;                '
#.
'  z_50lo     bits            ZDAB+7,21,11;               '
#.
```

The above three lines will output the highest 16 bits of the 50 MHz high word, the lowest 16 bits of the 50 MHz high word, and the 11 bits of the 50 MHz low word. This information can be used to double check that SNOMAN is correctly producing the elapsed 50 MHz time supplied in the Event Bank. Note that $16 + 16 + 11 = 43$ total bits in the 50 MHz clock record.

2.2.3 Checking the 50 Mhz Record

The bit-packed information can be used to double check that SNOMAN is correctly producing the time stamps supplied in the Event Bank. We have tested that the number of nanoseconds derived from the number of 50 MHz ticks (supplied by the bit-packed ZDAB Bank) is equal to the number of nanoseconds reported in the Event Bank. The C++ code fragment of interest is:

```
// Use bit shifters
FiftyMHzTicks += (unsigned long) ( z_50hi_2 << 27 );
FiftyMHzTicks += (unsigned long) ( z_50hi_1 << 11 );
FiftyMHzTicks += (unsigned long) ( z_50lo   <<  0 );

FiftyMHzNanoSecs = (unsigned long) FiftyMHzTicks * NanoSecsPer50MHzTick;

if(  FiftyMHzNanoSecs == (NanoSecsPerSec * gtr_sc)
     + (NanoSecsPerMicroSec * gtr_usc) + gtr_nsc  )
  {
   .
   .
   .
```

This code calculated a 50 MHz time from the bit-packed (z_...) variables and then compared it to the Global Trigger Time (gtr_...) variables. This test was applied to the first selected event for each run 10000 - 15299 and no events failed.

2.3 Extracting the 10 MHz Clock Time

For completeness we include the method for extracting the 10 MHz clock information. We will use this 10 MHz time information as the standard which the 50 MHz will be compared to.

```
#.
'  jdy_smjd ntime          EV+$KEV_JDY;                  '
'  jdy_sc   reserved       ;                             '
'  jdy_msc  reserved       ;                             '
'  jdy_nsc  reserved       ;                             '
#.
```

The above four lines will output the SNO Modified Julian Day, seconds, microseconds, and nanoseconds, respectively. This is just another format of the Universal Date Stamp as stated by the *The SNOMAN Companion* in the Event Bank.

3 Extracting Time Information from Runs

3.1 Impetus

In this study we are focused on calculating the actual rate of the 50 MHz clock. We will use the 10 MHz clock (which is synchronized to GPS time) as a time standard which the 50 MHz clock can be compared to. The challenge to making this comparision is that the 50 MHz rolls over every 2 days. Thus we will have to be able to identify when the 50 MHz rolls over if we are to be able to compare the two clocks over long periods of time.

The obvious thing to do is to scan all (or nearly all) of the events in a given run. This allows a continual comparison between the two clocks and allows us to microscopically determine the moment of 50 MHz roll-over. This method was employed in the work which produced the *50 MHz Time* document. We refer the reader to that document for a better description and explaination of this microscopic proceedure.

The above method is by far the most trust worthy, however it requires a significat amount of computer processing time. As a first look at 50 MHz clock's time, the *50 MHz Time* document shows us the way for addressing this limitation. For our purposes here, the important result given in the *50 MHz Time* document is that the run durations calculated from the 10 and 50 MHz clocks agree very well. This agreement is not perfect, of course. The 50 MHz clock consistantly reports a *shorter* run duration than the 10 MHz clock. (Please look ahead to Figure 5.) The clocks "agree" because the 50 MHz clock reported, on average, a run duration one scond shorter than the 10 MHz clock. The *50 MHz Time* document only considered run durations to an accuracy level of one second, thus the "slowness" of the 50 MHz clock was within the uncertainty introduced by rounding off to the nearest second.

The above result leads us to believe that a microscopic approach to identifying 50 MHz roll-overs is not necessary. If the 10 and 50 MHz show disagreement on the order of seconds from run to run then we should be able to use that knowledge to (if necessary) "add in" the appropreate number of roll-over periods (see Section 2.1 for the exact length of time added due to one roll-over) which will bring the clocks into agreement, even though we may only look at a few events in each run. Since SNO runs are in general contiguous, run duration is nearly equivalent to the time between the starts of two runs. Thus we will simply extract the 10 and 50 MHz time information from the first event of each run. The processing time for extraction of the first event of a run is ooddles of orders of magnitude shorter than the microscopic method originally used in the work presented in the *50 MHz Time* document. We will show that, indeed, only looking at the first event's time information from each run, is good enough for our purposes.

3.2 Extracting the First Event of Each Run

As already stated, we intend to extract and use both the 10 and 50 MHz clock's time information. It's unfortunately the case that we have to define what the "first event" of a run is. In this section we define what "first event" means through out the rest of this document.

When the data is initially processed in SNOMAN, a ntuple of the first 15 records is output for each run. This means most ntuples end up with 13 actual events that include a time stamp (since the first two records are actually run headers or something). These short ntuples are then processed further so that the first non-ORPHAN event, out of the first (approximately) 13 events is selected and then defined as the "first event" of the run. Thus, everywhere in this document, aside from this section, when we say "the first event of each run", we actually mean "the first non-ORPHAN event from the first (approximately) 13 events of each run". One should note that this allows for cases where ALL of the first (approximately) 13 events are ORPHANs. In this last case we are forced to record an ORPHAN as the defined "first event" of that run. Thus we will have to deal with ORPHANs again at a later stage of this analysis.

4 Defining Valid Time Records for 50 MHz Calibration

4.1 Taking Stock

At this point we have explained how we have produced a file which contains both the 10 and 50 MHz clock information of the first event of every single run (10000 - 15299). Henceforth, the first event's time will be synonomous with the run's start time. Furthermore, since each run has now been assigned a single starting time stamp for both the 10 and 50 MHz clocks, when we refer to a run we are implicity referring to the set of 10 and 50 MHz time stamps for that run.

There are a number of difficulties which we still face in using this data. There are runs which have no ZDABs and/or no events. Some runs begin with ORPHANs, and thus have no 10 MHz information which is useful. Other runs simply have 10 MHz times which are wrong. An example of this is a 10 MHz time which comes before the previous run's 10 MHz start time but is not an ORPHAN. These difficulties mean we will have to define a criterion for selecting valid runs (i.e. time stamps).

4.2 Defining Valid Runs for Clock Comparisons

In our analysis we imposed several requirements on the 10 MHz time and thus generated a valid set of times which the 50 MHz clock could be compared to. The cuts we applied are:

- Missing runs are noted and then simply ignored.
- The SNO Modified Julian Day must be ≥ 9072 (The start day run of 10000).
- The 10 MHz time must be later than the previous (valid) run's 10 MHz time.

As will become appearent later, we are actually concerned with finding and indentifying valid *pairs* of runs so that the time difference between the starts of runs can be calcualted. The criterion listed above have important ramifications on how we bring these valid pairs of runs together.

4.2.1 Missing Runs

When we say that we ignore missing runs we mean more than just, "We ignored it because it wasn't there." We also ignore the fact that it is a break in a consecutive series of runs and may be indicative of problems. Figure 1 shows how a missing run (e.g. 10672) is ignored and the time difference between the starts of the (valid) runs 10671 and 10673 will be calculated. The situation shown in Figure 1 seems inconsequential but the code allows for exactly the same sort of thing to happen when multiple consecutive runs are missing time records. One should be very concerned that we are allowing pairs of runs to be matched up and called a "valid pair" even when they are seperated by potentially tens of runs! We feel this is not a problem. It is possible for the 50 MHz clock to be independently ticking away while other things are conspiring to create numerous "empty" runs. If we can show that the 10 MHz run-to-run time difference for the valid pair is

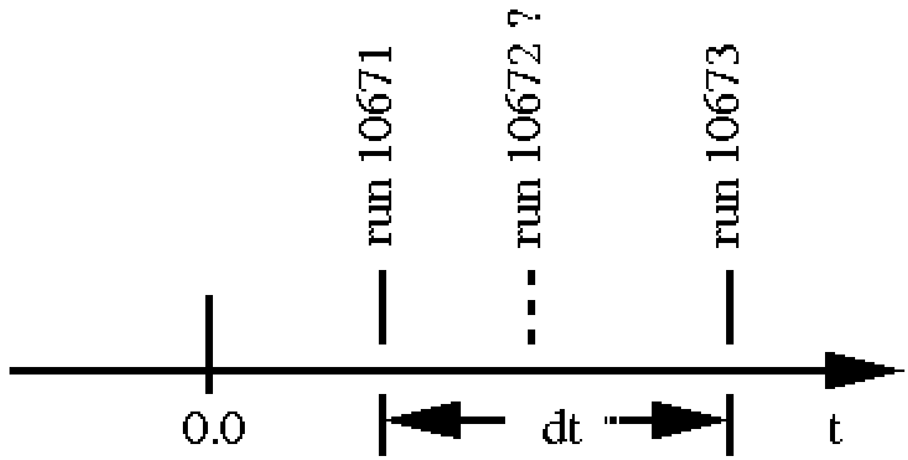

Figure 1: A missing run. If a run (e.g. 10672) is missing time records, or wholely absent, we ignore this fact and then calculate the run-to-run time difference between 10671 and 10673.

nearly equal to the time difference calculated from the 50 MHz, then we will claim we should use that valid pair of runs regardless of the fact there are missing runs "seperating" the valid pair.

4.2.2 SNO Modified Julian Days Less Than 9072

Requiring that the SNO Modified Julian Day of any run be greater than or equal to the SNO Modified Julian Day of the start of production data (run 10000) is a simple way to exclude runs which have erronious 10 MHz times. The major culprite in this class is runs that begin with ORPHANs. ORPHAN events have a SNO Modified Julian Day equal to 7671 and are missing all other 10 MHz time information. We could have used this date to selectively remove ORPHANs, but requiring *every* time to be after the start time makes perfect sense.

Figure 2 shows how an ORPHAN run is placed in a time line of events. In the graph "0.0" signifies the start of production data (i.e. not a true zero in any of SNO's time measures). Again, when we exclude ORPHAN events from the following analysis we end up allowing the creation of pairs of valid runs which the ORPHAN run should have come between. Our line of argumentation follows the same way as in Section 4.2.1.

4.2.3 "Negative" Time Runs

In a similar way and for similar reasons we exclude runs which show time records which come before the previous (valid) run. Figure 3 shows the situation schimatically. Once again, when we exclude these runs we allow valid pairs of runs to be formed around the offending run. Our justification is again that if we can match up the 10 MHz clock run-to-run time with the 50 MHz run-to-run time then there is no reason to throw out the pair simply because one intervening event has a garbled time stamp. See Section 4.2.1 for more argumentation along these lines.

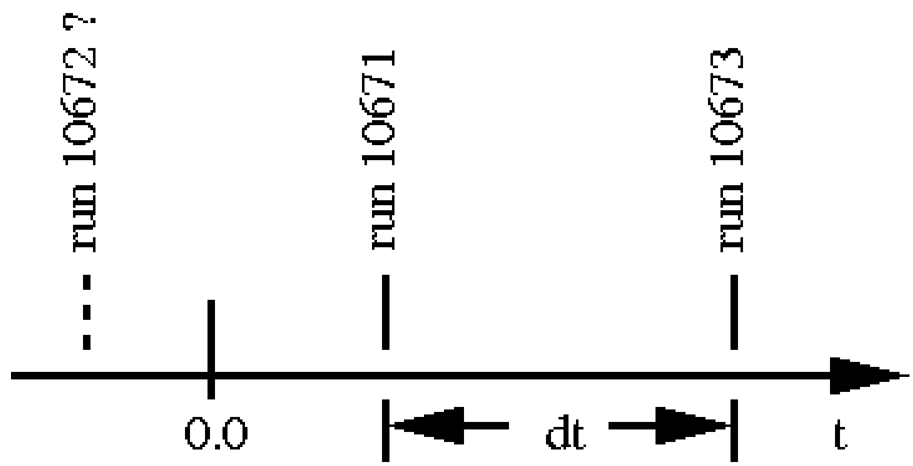

Figure 2: A "Pre-10000" run. In this graph 0.0 signifies the start of production data at run 10000 (i.e. SNO Modified Julian Day 9072).

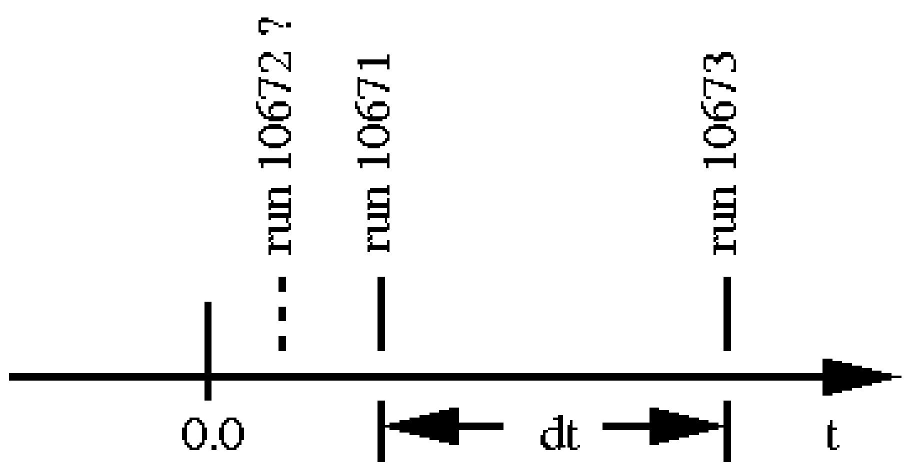

Figure 3: A "Negative Time" run. We exclude any runs which appear to run backwards in time since that just doesn't make any sense.

5 Calculating the Elapsed 50 MHz Time

5.1 The Idea

We wish to compare the 50 MHz clock's rate to a known standard: the 10 MHz clock. We need to collect a number of (simultaneous) 10 and 50 MHz time stamps and then see, after given periods of time, how much the 50 MHz clock's stated time has lagged behind the 10 MHz clock's stated time. This information is enough to allow us to calibrate the 50 MHz clock's actual rate.

The easiest way to accomplish this task to simply calculate the elapsed time between two (or more) runs for both the 10 and 50 MHz clocks. This has the advantage that we do not require that either clock be absolutely correct. That is to say, the 10 MHz could be off GPS time by a few seconds but still runnning at the correct rate. If this is the case, because we are calculating an elapsed time between two events, we can still make an accurate comparison.

The above example does *not* actually exist for any period long enough to make a comparison between the two clocks. However, our philosophy allows us to string together many runs and make longer elasped periods of comparison. This would otherwise be impossible if we required that every 10 MHz time stamp be GPS verified. This lack of GPS verification is not because the 10 MHz clock is wrong but because the GPS task is "out of commission" for whatever reason.

Given this strategy, we are left with two intertwined tasks. We need to accurately determine the number of 50 MHz roll-overs inbetween the start of any two given runs and calculate the elapsed time for both clocks from some given initial time. It should be obvious by now that we wish to extend these periods of comparison over as long of time intervals as possible. Thus we will not just compare the two clocks on a run-to-run basis but also over the combined time of many runs.

5.2 The 50 MHz Elapsed Time Algorythm

As explained in Section 2.1, the 50 MHz clock's record length is only long enough to hold approximately 2 days worth of 50 MHz ticks. Thus, the 50 MHz clock "rolls over" to zero approximately every 2 days. Using the knowledge of the 50 MHz record size we can develope an algorithm for determining the elapsed 50 MHz time from an initial 50 MHz time, taking into account the roll-overs.

The algorythm for calculating the elapsed 50 MHz time from an initial 50 MHz time is:

- $T^{\text{sec}}_{\text{Elapsed}} = (T^{\text{sec}}_{\text{Final}} + 175921 * N_{\text{roll-overs}}) - T^{\text{sec}}_{\text{Initial}}$
- $T^{\mu\text{s}}_{\text{Elapsed}} = (T^{\mu\text{s}}_{\text{Final}} + 860444 * N_{\text{roll-overs}}) - T^{\mu\text{s}}_{\text{Initial}}$
- $T^{\text{ns}}_{\text{Elapsed}} = (T^{\text{ns}}_{\text{Final}} + 140 * N_{\text{roll-overs}}) - T^{\text{ns}}_{\text{Initial}}$

In the above equations T is the 50 MHz time. The super-script gives the time units. The sub-script indentifies if the T refers to the 50 MHz time of the intial (i.e. earlier) run, the final (i.e. later) run, or the total elapsed time (i.e. the run-to-run time difference). The number of 50 MHz roll-overs which occure inbetween the initial and final 50 MHz records is identified by $N_{\text{roll-overs}}$. The constant numbers in each line come from a calculation of the maximum duration that the 50 MHz clock's 43 bits allow to be stored (see Section 2.1).

5.3 Looking for 50 MHz Roll-Overs

The algorythm of Section 5.2 implies that we already know the number of 50 MHz roll-overs between the two events we are considering (in our case the first events of runs). We can certainly re-arrange these equations so that we instead can determine the number of 50 MHz roll-overs. We would then need to know the elapsed 50 MHz time. We don't actually know the elapsed 50 MHz time, but we do know what values to guess and we can check our guess against the elapsed 10 MHz time.

We can simultaneously determine the both the number of 50 MHz roll-overs between two events and the elapsed 50 MHz time by a simple guess-and-check technique. Using the 10 MHz clock information of the first events of two runs, we can calculate the elapsed 10 MHz time. We then calculate the elapsed 50 MHz time *assuming* there have been zero roll-overs. We then compare these two elapsed times. If the elapsed 50 MHz time is in "good" agreement with the elapsed 10 MHz time, then we say we know that there have been zero 50 MHz rollovers and that we have a calculated value for the elapsed 50 MHz time. If on the other hand the elapsed 50 MHz time does not agree with the elapsed 10 MHz time, we recalcuate the elapsed 50 MHz time *assuming* there has been one roll-over of the 50 MHz clock record. We then make the comparison to the elapsed 10 MHz time again. If the addition of one roll-over period of 50 MHz time brings the clocks into "good" agreement, then we can say we know there was one 50 MHz roll-over and that we have calculated the elapsed 50 MHz time.

One should be able to see that this process can be exented to any number of 50 MHz roll-overs. However, there is an upper limit to the number of 50 MHz roll-overs which can occur within any single run. At one time, the longest a run was allowed to be was 96 hours. This is 96*3600 = 345600 seconds. Three 50 MHz roll-overs can occur within a time period of $2 \times 175921.8 + 1 = 351844.6$ seconds. One sees that 3 roll-overs requires slightly more time than SNO's longest run will allow. We choose to allow 3 roll-overs as the maxium number for which the 50 MHz is tested against the 10 MHz. It turns out that the largest number of 50 MHz roll-overs ever comfirmed by our method is 2.

5.3.1 What is "Good" Agreement?

In Section 5.3 we state that the 50 MHz run-to-run time needs to be in "good" agreement with the 10 MHz run-to-run time. From the *50 MHz Time* document we know that "good" agreement should be on the order of a few seconds. We will need to introduce a parameter which will quantify "good" agreement between the 10 and 50 MHz run-to-run times.

Based on some preliminary (i.e. by hand) calculations of the actual 50 MHz rate, we determined that the rate is around 49999461 Hz. Using this we can calculate the largest difference we might expect between the two clock's run-to-run times. Recalling the longest run time is 96 hours we calculate

$$[96 \times 3600]_{10\text{MHz}} - \left[96 \times 3600 \times \left(\frac{49999461}{50 \times 10^6}\right)\right]_{50\text{MHz}} = 3.72\text{sec} \tag{2}$$

is the largest difference we could expect to see between the two clock's run-to-run times. The value is positive because the 50 MHz is slow. This value is, of course, just an estimate. We should also be aware that there may be (50 MHz) jitter so that individual time stamps could fluctuate away from the average rate. Conscientious of these things, we will begin by saying "good" agreement is defined as

> If the absolute difference between the 10 and 50 MHz clock's run-to-run times is less than 4 seconds, then the 10 and 50 MHz clock's are in "good" agreement.

We need to show that this is a reasonable definition.

Figure 4 shows an enlarged range of run-to-run time differences. Let's recall what the run-to-run time difference means. We defined the run-to-run time to be the amount of time which has elapsed between the first events of two runs. The difference referred to comes from subtracting the run-to-run time as determined by the 50 MHz clock from the run-to-run time as determined by the 10 MHz clock. Also, make sure to have in mind the fact that we are adding in the appropreate number of 50 MHz roll-overs, based upon our guess-and-check method. Figure 4 shows that there is just a smattering of run-to-run time differences away from the 0.0. Figure 5 shows the run-to-run time differences which would be considered to be within "good" agreement based on our definition above.

One concern that crops up is whether we are preferentially taking "BAD" run-to-run time matches for pairs of events that include a run of short duration and eliminating "OK" run-to-run time matches for pairs of events that include a run of long duration. One can see how this would be possible by considering the expected deviation of the 50 MHz clock from the 10 MHz clock for short periods of time verses long periods of time. Periods (only consider ones with correct time stamps) which include a short run should have a smaller run-to-run time difference than those which include a long run. However, our definition of "good" agreement is a fixed time window. This situation inherently allows for more "slop" when the run-to-run time is short rather than long.

Figures 6, 7, & 8 show run-to-run time differences segregated into either less than 1 hour or greater than 50 hours. The net result seems to be that our definition of "good" catches all the run-to-run time differences which are greater than 50 hours. This is a good sign. A further "warm fuzzy" is that it appears that the distribution tightens up around 0.0 for run-to-run time differences which are less than 1 hour. This

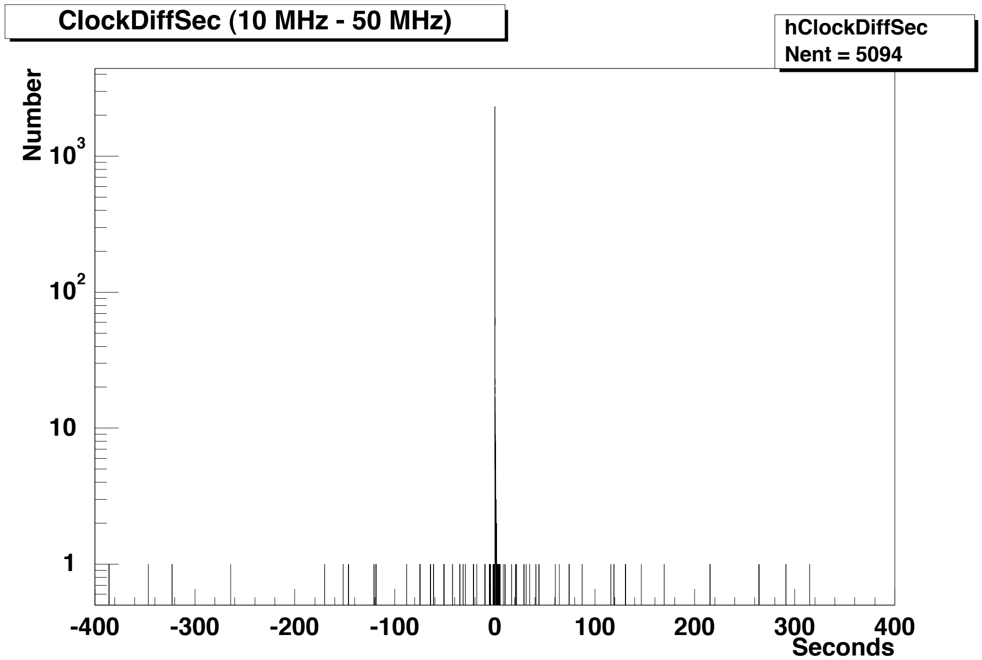

Figure 4: Run-to-run time differences of the 10 MHz and 50 MHz clocks.

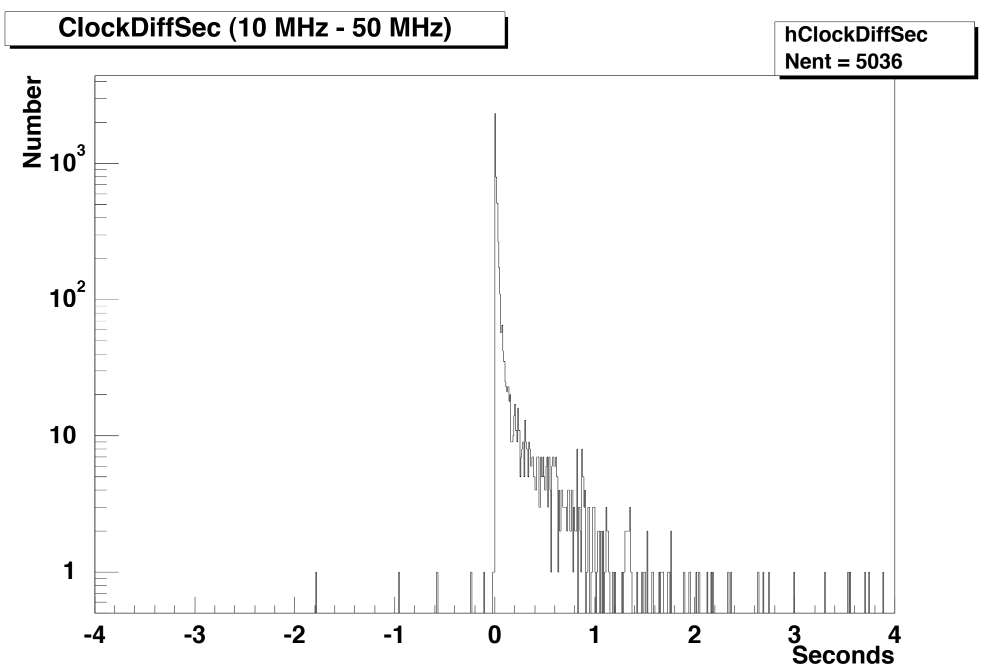

Figure 5: Run-to-run time differences of the 10 MHz and 50 MHz clocks.

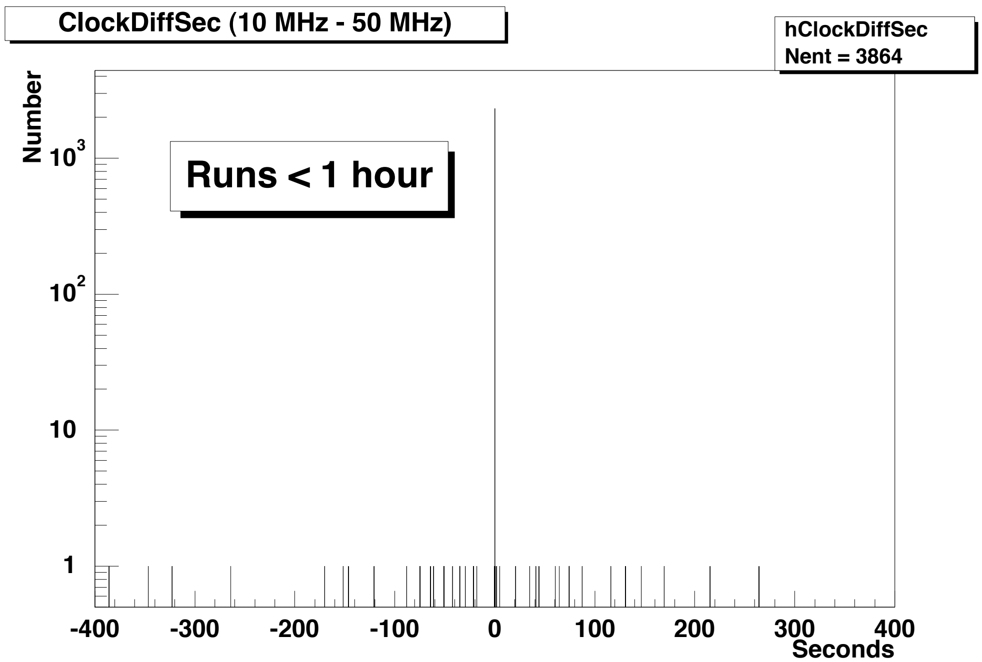

Figure 6: Run-to-run time differences of the 10 MHz and 50 MHz clocks.

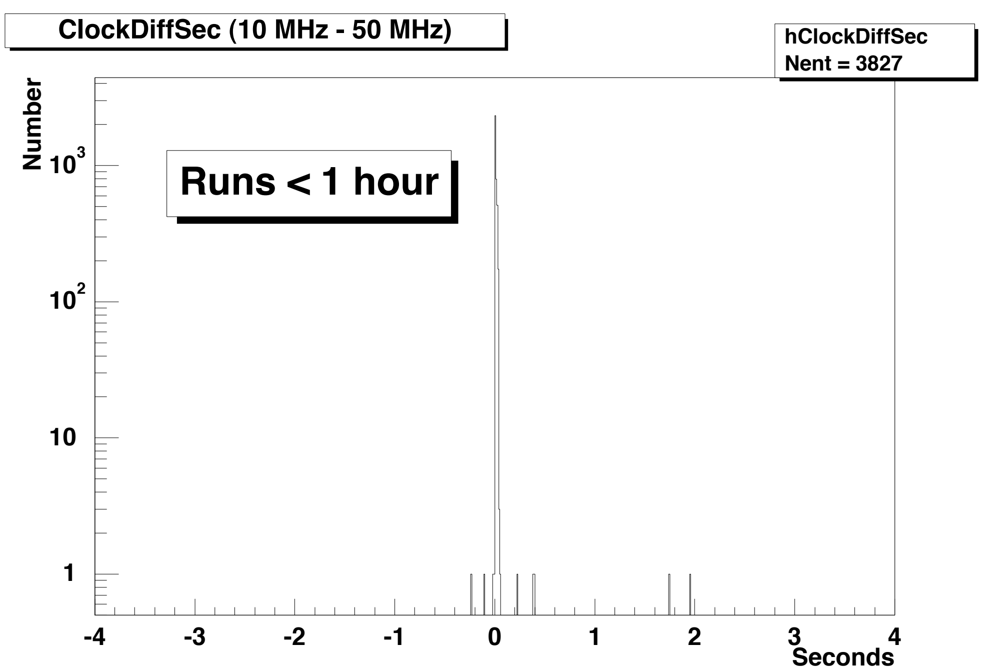

Figure 7: Run-to-run time differences of the 10 MHz and 50 MHz clocks.

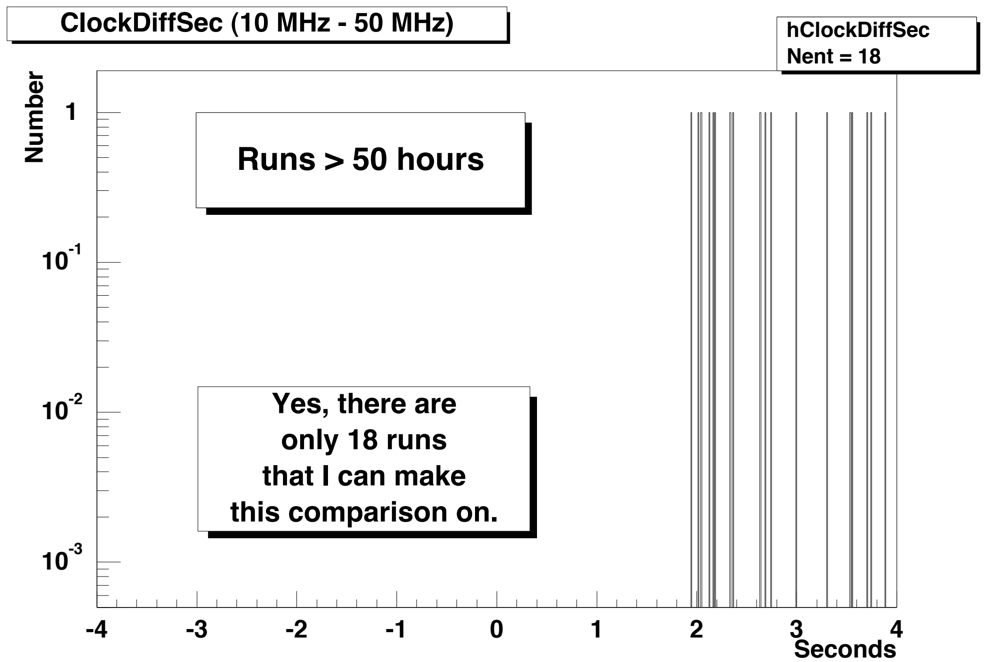

Figure 8: Run-to-run time differences of the 10 MHz and 50 MHz clocks.

is at least what one would expect, though it really isn't any sort of a guarantee that everything is fine. So in the end we feel that we have demonstrated that we can confidently use our method of catching 50 MHz roll-overs.

5.4 Making a Continuous Elapsed Time

In Section 5.3 we focused on determining how many 50 MHz roll-overs occurred between any two given events. In doing this test we were actually calculating the run-to-run times which can be used to compare the 50 MHz clock's rate to that of the 10 MHz clock. As already stated, we wish to compare these two clocks over long periods of time and many measurements. It should be obvious that we can successively calculate the run-to-run times for several contiguous[1] runs and then add the run-to-run time together to get a longer elapsed time. One can imagine doing this for several runs and then ploting the 50 MHz elapsed time against the 10 MHz elapsed time. This plot can then be fitted with a straight line and the slope will be indicative of the 50 MHz clock's actual rate. This is exactly what we will do.

Figure 9 shows the total elapsed time of the 50 MHz clock verses the total elapsed time of the 10 MHz clock for runs 10000 to 15299. The "kinks" in this line are due to the 50 MHz being reset. The 50 MHz is reset when the timing rack (or wherever the 50 MHz scalar is located) loses power. Figure 10 shows the same sort of information as Figure 9, except we have set the elapsed 50 MHz time to zero whenever we detect that the 50 MHz has been reset. Actually, this is a bit of a lie since we are never absolutely certain that the 50 MHz has been reset. Recall from Section 5.3 how we tested for the number of 50 MHz roll-overs. We never mentioned what we did when we could not find agreement between the two clocks. Figure 10 shows what we do. We consider any disagreement between the two clocks to be a sign that the 50 MHz has been reset. A disagreement between the run-to-run times *could* come from the 10 MHz clock or 50 MHz jitter. However, Section 4 shows how we have attempted to eliminate incorrect 10 MHz times. In the end, these details are not important because what we really want is long periods of continous elapsed time where there is no question about the each of the clocks' state. One should by now be convinced by looking at Figure 10 that we *can* string together long periods of continous 50 MHz elapsed time that can be compared to the 10

[1]Sometimes semi-contiguous as shown in Section 4.2.

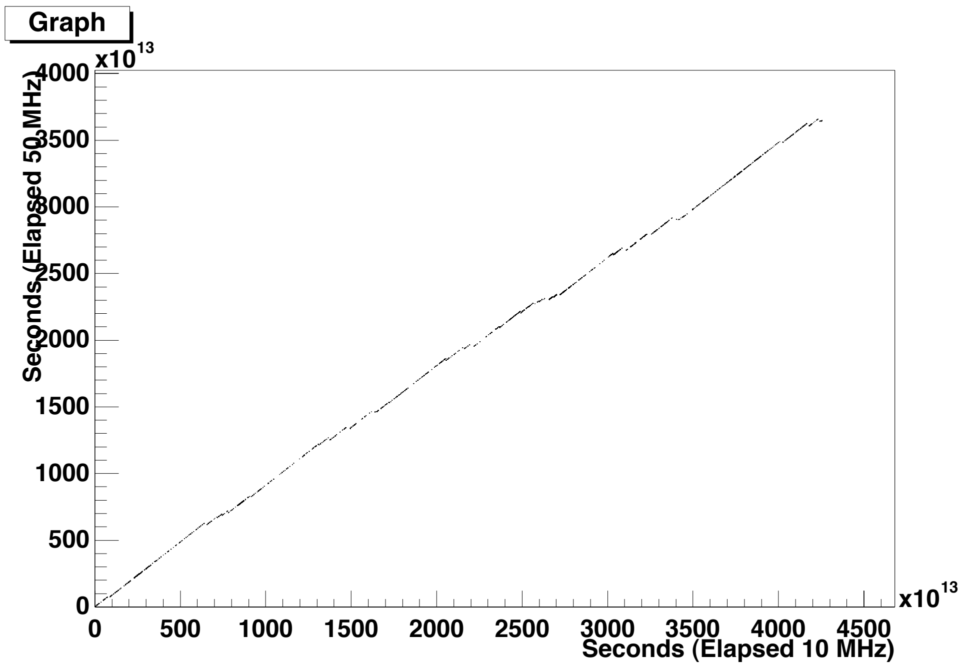

Figure 9: The elapsed 50 MHz time verses the elapsed 10 MHz time. The "kinks" are due to the 50 MHz clock record getting reset.

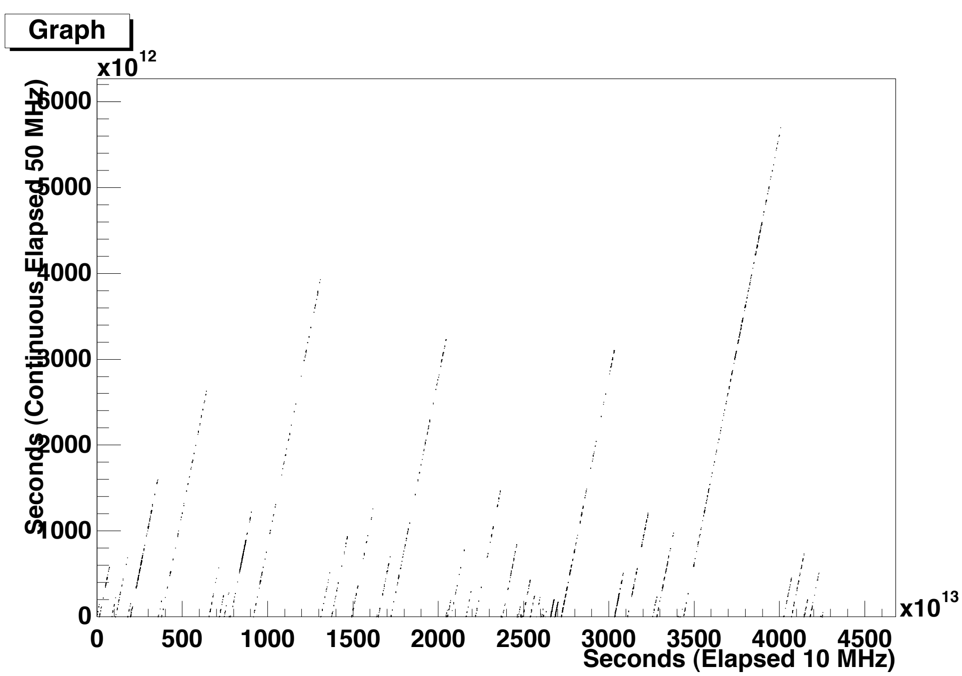

Figure 10: The continuous elapsed 50 MHz time verses the elapsed 10 MHz time. The elapsed 50 MHz time is set to zero whenever the 50 MHz clock record is reset.

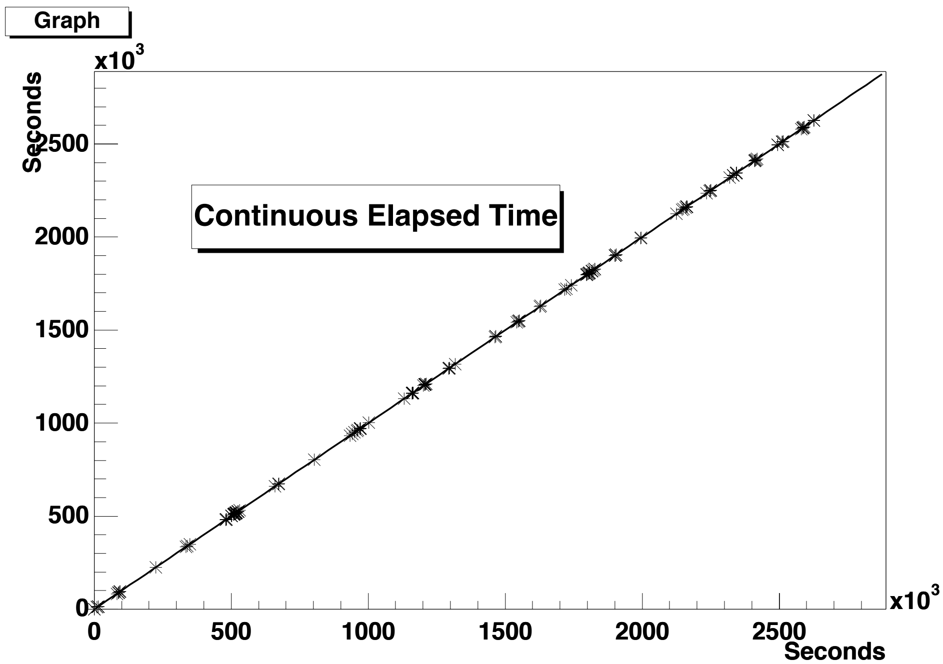

Figure 11: The continuous elapsed 50 MHz time verses the continuous elapsed 10 MHz time.

MHz clock.

6 Comparing the Clocks

Having used a variety of tests, we have collected information about the 10 and 50 MHz elapsed run-to-run times. We have shown that these elapsed run-to-run times can be concatonated to form continuous periods of 10 and 50 MHz clock monitoring. We will require in any further analysis that the continuous elapsed 10 MHz time be greater than 1.5 days for a given period of continuous clock monitoring (i.e. no 50 MHz resets). This 1.5 days is an arbitrary number which we will justify after our analysis is complete.

6.1 Fitting the Data

For any given period of continuous clock monitoring there will be a number of records. The data consist of pairs of elapsed 10 and 50 MHz times. One pair from each run-to-run calculation that is made. Figure 11 is an example of one period of continuous elapsed time. This data is then fit with a line which is forced to go through the origin since we calculate both the 10 and 50 MHz continuous elapsed times from the same starting event. The formula of the fit is:

$$y = 0.0 + x\left(1 + \frac{\Delta}{50 \times 10^6}\right) \qquad (3)$$

The fit to the data determines the parameter Δ which is the difference between the actual rate of the 50 MHz clock and 50 MHz. It is then trivial to determine the 50 MHz clock's actual rate. Applying all these techniques to runs 10000 - 15299 we arrive at the following results which are listed in Table 1.

6.2 Residuals

We should show some residuals from our analysis. We define the residual as the difference between the continuous elapsed 10 MHz time and the continuous elapsed 50 MHz time *after* the actual rate of the 50

Days	Run Range	Δ (Hz)	Error (Hz)	Rate (Hz)
1.6	10000 - 10016	-519.2	142.3	49999480.7
8.4	10018 - 10105	-510.0	13.7	49999489.9
2.5	10111 - 10134	-82.4	100.7	49999917.5
7.9	10140 - 10178	-525.6	27.1	49999474.3
1.8	10182 - 10199	-515.0	155.4	49999484.9
18.4	10204 - 10679	-516.3	2.9	49999483.6
2.1	10681 - 10694	-526.6	103.5	49999473.3
32.1	10696 - 10838	-516.7	2.7	49999483.2
7.0	10839 - 10888	-514.4	42.6	49999485.5
3.2	10889 - 10935	-514.7	58.7	49999485.2
3.2	10936 - 10950	-521.7	110.9	49999478.2
15.4	10957 - 11291	-518.0	3.7	49999481.9
45.6	11292 - 11513	-533.9	1.5	49999466.0
7.1	11519 - 11541	-581.6	37.6	49999418.3
13.6	11542 - 11621	-548.0	8.2	49999451.9
16.9	11622 - 11706	-559.2	10.4	49999440.7
8.5	11717 - 11804	-525.2	12.6	49999474.7
37.3	11805 - 12094	-553.1	1.4	49999446.8
2.3	12133 - 12166	-549.6	100.2	49999450.3
10.2	12167 - 12192	-528.6	20.7	49999471.3
6.3	12193 - 12211	-505.1	69.2	49999494.8
17.1	12212 - 12310	-558.3	4.8	49999441.6
9.8	12330 - 12412	-524.2	9.7	49999475.7
5.0	12466 - 12533	-537.2	24.6	49999462.7
3.9	12534 - 12576	-483.5	66.0	49999516.4
2.9	12577 - 12585	-565.2	100.5	49999434.7
3.0	12619 - 12790	-543.6	31.4	49999456.3
4.2	12791 - 12975	-551.7	35.5	49999448.2
36.1	12976 - 13505	-550.9	1.6	49999449.0
8.7	13506 - 13735	-527.2	13.2	49999472.7
17.0	13736 - 13982	-533.3	4.2	49999466.6
2.7	14006 - 14037	-522.4	62.8	49999477.5
14.2	14040 - 14196	-533.4	8.5	49999466.5
67.9	14228 - 15035	-520.4	0.5	49999479.5
5.2	15036 - 15079	-568.5	26.0	49999431.4
8.4	15081 - 15182	-524.3	12.0	49999475.6
4.0	15185 - 15229	-530.5	62.4	49999469.4
7.2	15230 - 15281	-521.5	31.6	49999478.4

Table 1: Fits to periods of continuous elapsed time for the 10 and 50 MHz clock.

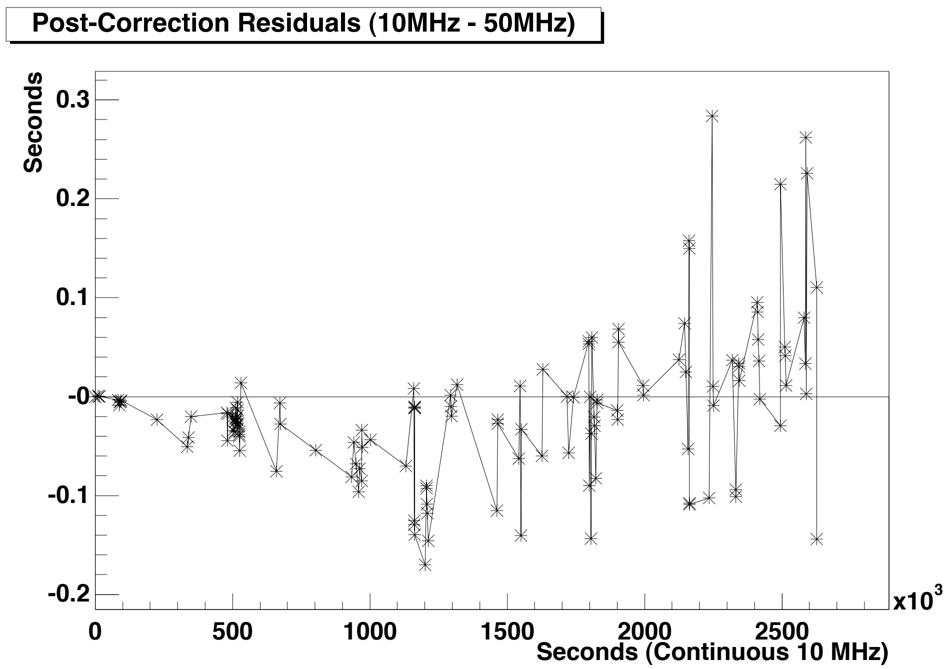

Figure 12: Residuals for the period between runs 10696 and 10838. This is the period shown in Fig. 11.

MHz clock has been accounted for.

$$y_i^{residual} = x_i - y_i \left(\frac{50 \times 10^6}{50 \times 10^6 + \Delta} \right) \tag{4}$$

Before showing the residual plots we should mention what these plots would like before the 50 MHz rate is corrected. Since the 50 MHz clock is slow compared to the 10 MHz clock, a plot of the difference of the continuous elapsed times of the 10 and 50 MHz (i.e. 10 MHz - 50 MHz) will show a line with positive slope if the residuals are plotted as a function of the elapsed 10 MHz time.

When the residuals are calculated after the 50 MHz clock has been corrected for, one would expect the residuals to be equally distributed above and below zero. One would also expect the absolute difference from zero to grow as a function of the 10 MHz elapsed time. Figures 12, 13, and 14 are examples of the residuals for 3 different periods of clock comparison.

6.3 Combined Fit

Taking the results shown in Table 1 we can histogram the actual rate of the 50 MHz clock as done in Figure 15. The weighted mean (i.e. using the errors list in Table 1) of this distribution is

$$R_{50} = 49999473.2 \pm 0.4\text{Hz}. \tag{5}$$

Thus, as has already been stated, the 50 MHz clock does not run at 50 MHz.

6.4 Sanity Checks

There are two choices of parameters we made during this analysis. It is worth altering the values chosen for those parameters to see if our results are stable. By far the most crucial parameter choice was the amount of time we allowed the 50 MHz run-to-run time to deviate from the 10 MHz run-to-run time. This was our so called "good" agreement between the 2 clocks. Table 2 shows how R_{50} is a function of our "good" parameter. Upon close inspection of Table 2 one sees that there is a local minimum in the number of periods

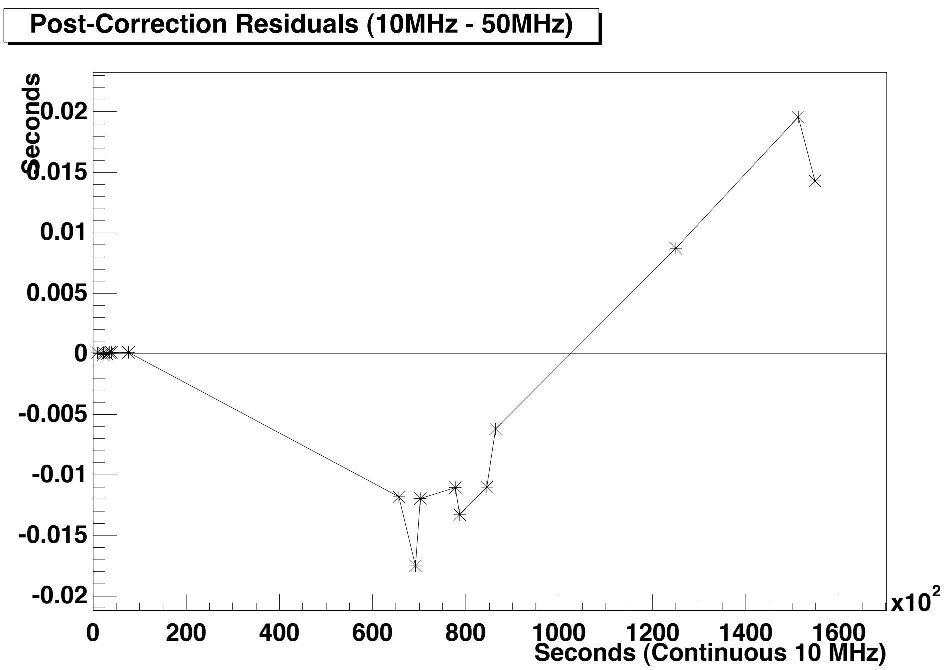

Figure 13: Residuals for the period between runs 10182 and 10199. This is the period with the poorest fit.

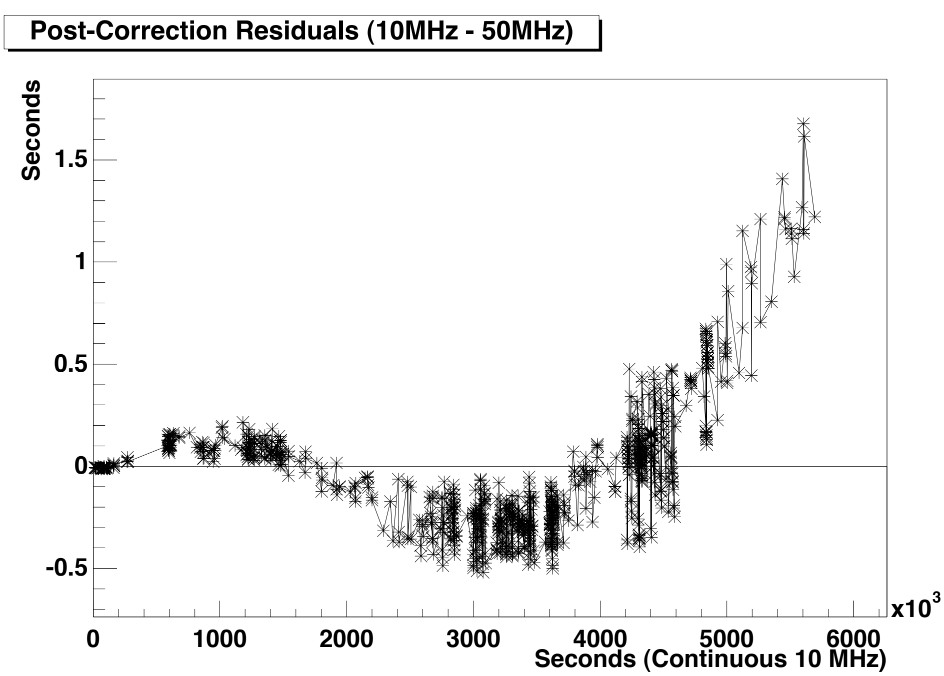

Figure 14: Residuals for the period between runs 14228 and 15035. This is the longest, best fit period.

Seconds	Periods	Rate (Hz)	Error (Hz)
0.25	7	49999483.5160	7.72375
0.50	30	49999471.8232	5.16228
0.75	64	49999469.1249	3.77885
1.0	59	49999480.1105	0.59738
1.5	54	49999480.0625	0.56982
2.0	52	49999479.3880	0.56224
2.5	48	49999478.4359	0.54625
3.0	44	49999475.9282	0.51949
3.5	43	49999475.9186	0.45740
3.75	39	49999473.3995	0.42520
4.0	38	49999473.2067	0.42390
4.5	38	49999473.2067	0.42390
5.0	39	49999473.2295	0.42387
5.5	39	49999473.2237	0.42382
6.0	39	49999473.2237	0.42382
6.5	39	49999473.2237	0.42382
7.0	39	49999473.2237	0.42382
7.5	39	49999473.2237	0.42382
8.0	39	49999473.2237	0.42382
8.5	40	49999473.1928	0.42382
8.75	40	49999473.1928	0.42382
9.0	40	49999473.1928	0.42382
9.25	40	49999473.1928	0.42382
9.5	39	49999473.9281	0.42353
10	39	49999473.9281	0.42353
11	39	49999473.9174	0.42352
12	39	49999473.9174	0.42352
15	39	49999473.9174	0.42352
20	39	49999473.9174	0.42352
30	38	49999473.8092	0.42228
40	38	49999473.7553	0.42227
100	37	49999472.6099	0.42148
200	34	49999475.0633	0.41637
400	32	49999077.9134	0.41539

Table 2: The fit results for the 50 MHz rate as a fucntion of our “good” agreement parameter. This parameter is the allowed time difference between the run-to-run times from the 10 and 50 MHz clocks. See Section 5.3.1 for details.

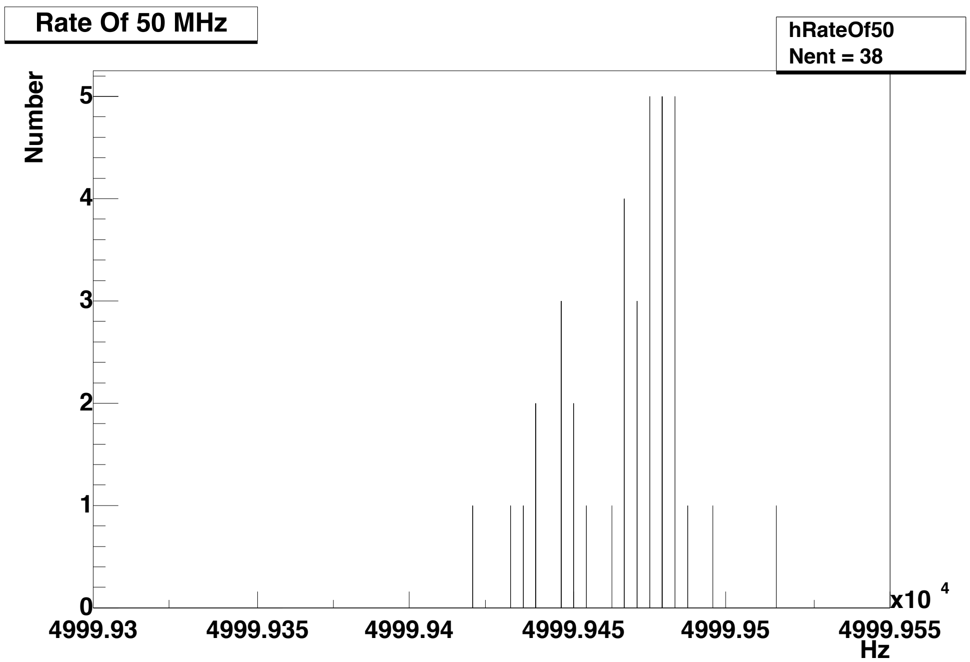

Figure 15: Histogram of the results of fits to 38 different periods of 50 MHz clock comparisons to the 10 MHz clock.

found/used at the 4 second allowed difference. This is the value we chose to used based upon an earlier estimate of the 50 MHz rate. One will also notice that there is a local maximum in the number of periods found/used at a 9 second allowed difference.

There is a way to think about this situation in an idealized world. Assume that all runs are the same length. This means all the run-ro-run times are the same. Furthermore, assume that the 50 MHz clock is slower than the 10 MHz clock in such away that the difference between the two clocks is Gaussian distributed. In this idealized case, one expects the number of periods found/used to increase(decrease) as the allowed time difference decreases(increases). This idealized case is useful because it explains the general trend shown in Table 2.

Upon looking at the 2 additional periods which cause the local maximum we find that both are periods of less than 1.9 days. In other words, increasing the allowed time difference let two more periods pick enough time to become part of the fit. We notice that one of these periods has a calculated 50 MHz rate of 49995812.1 - a significant outlier. So in the end we stick with our "good" agreement parameter set to 4.0 seconds difference between the clocks. It makes sense to stick with the smallest value possible so long as it doesn't bias your final answer. We feel the overall consistency shown in the 50 MHz rates of Table 2 is an indication of our freedom from bias.

The next parameter we should reconsider is the minimum length which we require before a period of continuous elapsed time is added to the analysis. Our results have been shown with a minimum period length of 1.5 days. Table 3 shows the results as a function of the chosen minimum length (in hours). Table 3 shows excellent consistency regardless of the minimum length required. This is really only a statement about the nature of a weighted average - the very longest periods (one's with small errors) dominate the average and thus the minimum length chosen is almost irrelevant.

7 Conclusion

The 50 MHz clock does not run at 50 MHz. We have shown that on average the 50 Mhz ticks at a rate

$$R_{50} = 49999473.2 \pm 0.4\text{Hz}. \tag{6}$$

Hours	Periods	Rate (Hz)	Error (Hz)
6	49	49999473.20661	0.423907
12	44	49999473.20690	0.423908
18	42	49999473.20692	0.423908
24	41	49999473.20695	0.423908
26	39	49999473.20690	0.423908
30	39	49999473.20690	0.423908
36	38	49999473.20675	0.423909
42	37	49999473.20668	0.423911
48	35	49999473.21016	0.423943
54	33	49999473.21057	0.423950
60	31	49999473.21379	0.423999

Table 3: The fit results for the 50 MHz rate as a fucntion of the minimum period length allowed for the fit. This analysis uses 36 hours (1.5 days) as the minimum length required for a period of 10 and 50 MHz monitoring to be included in the analysis. See Section 6 for details.

Anyone who wishes to use the 50 MHz clock for anything other than event-to-event, relative timing, should heed this notice and make good use of it.