高精度時刻同期技術をベースとしたトリガ・タグ情報配信システムの開発

DEVELOPMENT OF A TRIGGER AND TAG INFORMATION DISTRIBUTION SYSTEM BASED ON THE HIGH-PRECISION TIME SYNCHRONIZATION TECHNOLOGY

増田剛正
Takemasa Masuda #)
Japan Synchrotron Radiation Research Institute (JASRI)

Abstract

I have been developing a new-paradigm trigger and tag information distribution system based on a high precision time synchronization technology over a network. This system generates ~209kHz revolution frequency signal synchronized with the specified bunch address of the SPring-8 storage ring and can deliver any additional information such as a shot number and precise time. As the high precision time synchronization technology, White Rabbit, which provides sub-nanoseconds synchronization accuracy, is adopted. Since the system generates trigger signals based on the digital information of the precise time, the output timing can be easily adjusted by using software. The system can be easily and inexpensively expanded using single mode fibers and the White Rabbit switches. I have built the proof of concept system that consists of a minimum set of a master PC, a slave PC and two White Rabbit switches. The master PC detects the zero-address signal as a pre-trigger and stamps the absolute time with 2ps resolution. Then the master calculates the absolute output time of the fundamental revolution frequency signal and sends the time to the slave PC via White Rabbit network according to the given decimation rate. The slave PC generates the revolution frequency signal synchronized with the target bunch address by adding the offset time by software. At present, the measured one-sigma jitter of the output signals from the slave PC is less than 100ps.

1. 開発の目的

従来のアナログ信号の分配によるタイミンシステムでは、信号の分周や遅延のための各種高周波回路が必要であり、タイミングの調整にも手間が掛かる。分配されるのは“信号”であるため、データ取得のための計算機システムとの親和性も悪く、例えばショット番号などデータに付加したい情報は別途用意しなければならない。特にSPring-8 の実験ユーザにとって、アナログ式のタイミングシステムは導入と利用に対する敷居が高い。そこで汎用ネットワークを介した高精度時刻同期技術をベースとしたデジタル型のタイミング配信システムの開発を行うこととした。ネットワークを介して配信を行うため、扱いや拡張が容易で、計算機システムとの親和性も高い。またデジタル情報であるため、ソフトウェアによって簡便にタイミング調整が可能である。そしてショット番号などの情報も容易に付加できる。

デジタル型のタイミング配信システム構築の検証を行うため、最小構成のシステムの開発を行った。高精度時刻同期技術としては CERN を中心に研究開発が進められている White Rabbit[1]を採用した。White Rabbit は IEEE 1588 の改良型とも言える技術で、Synchronous Ethernet によるハードウェアレイヤでのクロック同期と Digital Dual Mixer Time Difference(DDMTD)による位相測定によりサブナノ秒以下の高精度時刻同期を実現している。White Rabbit の技術は Open Hardware Repository（OHR）[2]で公開され、開発のために必要なあらゆる情報が CERN Open Hardware License に従って自由に入手可能である。

masuda@spring8.or.jp

検証システムでは、SPring-8 蓄積リングの特定バンチに同期した周回周波数信号（~209kHz）の生成と、ショット番号などのタグ情報の分配の実現を目指した。

2. 検証システム（Version 1）の構築

新しいトリガ・タグ情報配信システムの検証を行うため、最小構成のシステム（Version 1）を構築した(Figure 1)。システムはマスターPC とスレーブ PC、2台の White Rabbit スイッチ（マスターPC 側を WR-A、スレーブ PC 側を WR-B とする）、GPS レシーバとそれらを結ぶシング

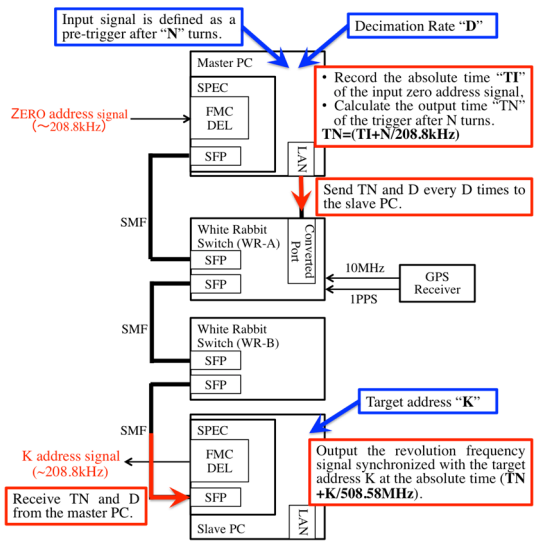
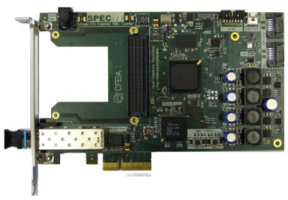
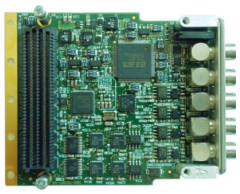


Figure 1: Block diagram of the proof of concept system (version 1).

ルモードファイバ(SMF)から構成される。WR-A には GPS レシーバからの 10MHz 信号と 1 PPS 信号を入力する。これによりマスターPC とスレーブ PC、WR-A、WR-B は SMF を介して White Rabbit ノードとして同期がなされる。マスターPC とスレーブ PC には、Xilinx Spartan6 を乗せた FPGA Mezzanine Card(FMC) キャリアボードである Simple PCI Express FMC Carrier(SPEC)[3](Figure 2(a))に、Fine Delay FMC (FMC DEL)[4](Figure 2(b))を組み合わせたボードを PCI Express 拡張スロットに実装している。これらの基板も OHR で公開されている。マスターPC、スレーブ PC ともオペレーティングシステムには Linux(Ubuntu 12.0.4 LTS)を用いた。



(a) SPEC board

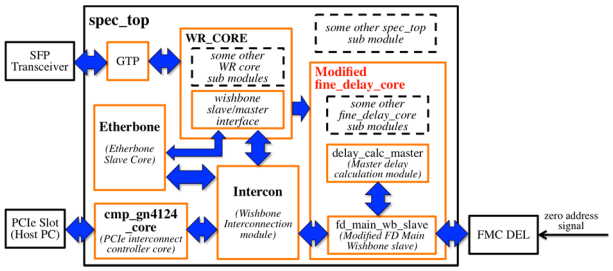


(b) Fine Delay FMC

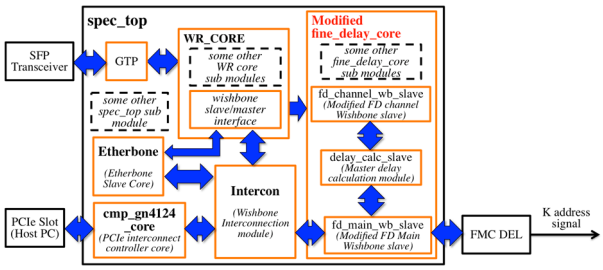
Figure 2: Pictures of the equipped hardware on a master and slave PCs.

マスターPC には蓄積リングの基準周回周波数信号であるゼロ番地信号(~209kHz)を入力する。これをスレーブ PC 側の N 回前のプリトリガであると定義して、入力したゼロ番地信号の絶対時刻 TI から、N 周後のゼロ番地信号の絶対時刻 TN を計算する。別途マスターPC に与えられた間引きレート D に従って、D 回に1回の割合で TN と D をスレーブ PC に送出する。スレーブ PC は TN と D の値を受け取り、別途与えられたバンチアドレス K の周回周波数信号を絶対時刻 TK=TN+K/508.58MHz に出力する。間引きの補間は FMC DEL のパルス列出力機能を用いて行う。マスターPC およびスレーブ PC の一連の処理を時系列で示したのが Figure 3 である。この機能を実現するために Version 1 で開発した FPGA ロジックを Figure 4 に示す。この機能は、マスターPC 側 FPGA ロジックの modified fine_delay_core とスレーブ PC 側 FPGA ロジックの modified fine_delay_core に組み込まれており、今回、上記に述べる独自の手法を実現するために新たに開発を行ったのが主にこのロジック部分にあたる。Version 1 の開発を開始した時点では、マスターPC 側の FPGA ロジックの制約により、SPEC ボードの SFP モジュールから直接 TN や D の値を送ることができなかったため、PC の LAN ポートからソフトウェアによってこれらの情報をスレーブ PC に送るように実装した。具体的には OHR において Etherbone Master Core の実装が出来ていなかったことによる。なお、Version 1 ではロジック構築の複雑さから、D>N の場合だけを実装している。

FPGA の動作に必要な N や D、K といった値を設定するソフトウェアに加えて、スレーブ PC 側の出力信号の周波数や Fine delay 量 F、パルス出力幅などを変更するユーティリティソフトウェアも用意した。これらのソフトウェアを使用して、実際に非常に簡便に出力信号の調整ができる。



(a) FPGA logic for a master PC.



(b) FPGA logic for a slave PC.

Figure 4: Block diagrams of the FPGA logics.

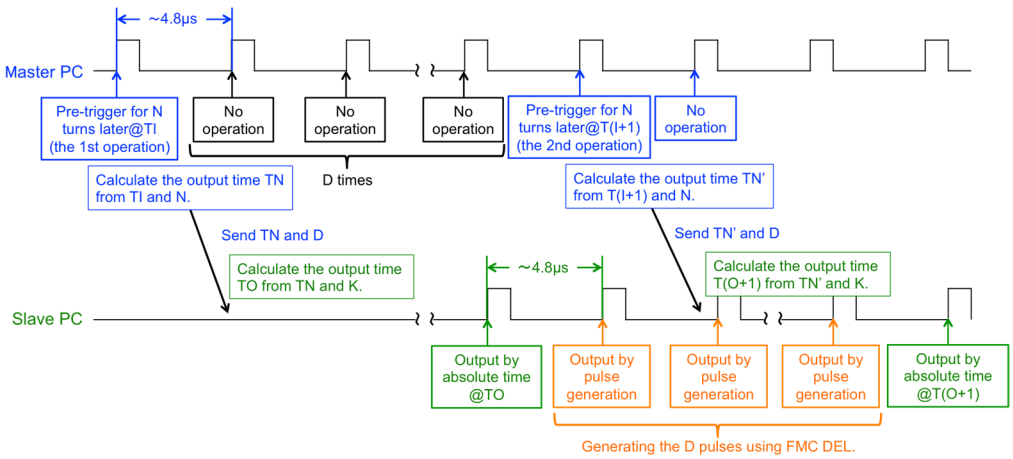


Figure 3: Timing chart of processes on the master PC and the slave PC.

3. 検証システム(Version 1)の評価試験

評価試験はテストベンチで行った(Figure 5)。ここではUTC(Coordinated Universal Time:世界協定時)と合わせる必要はないので、GPS Receiver の代わりにシンセサイザを用いた。シンセサイザからの 508.58MHz 出力を16bit カウンタモジュールに入力し、2436 分周して208kHz を作った。また、同じくシンセサイザからの10MHz 出力を WR-A に入力した。

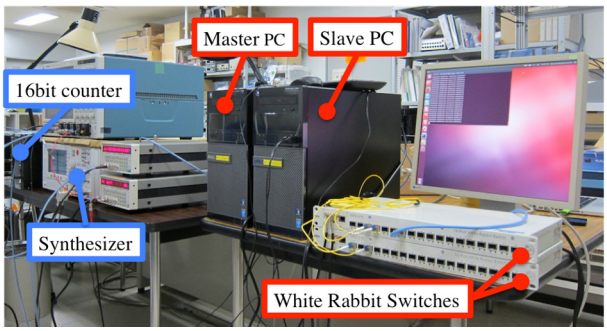


Figure 5: Evaluation test environment of the system.

3.1 出力信号のジッタ計測

スレーブ PC からの 209kHz 出力信号のジッタが最小となるように出力周波数を微調整して、ジッタ計測を行った(D=1000,N=900)。その結果を Figure 6 に示す。

この図から明らかなように、出力には不自然な構造が見え、±5ns を超える大きなジッタが観測された。ジッタ全体の標準偏差は約 250ps であるが、中心付近のみを切り出した場合の標準偏差は約 170ps であった。

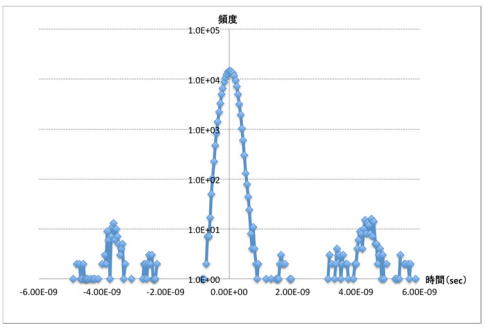


Figure 6: Result of the jitter measurement.

3.2 ジッタの間引きレート D 依存性測定

D の値を変えてジッタを計測した(Version1 では D>N という制約あり)。結果を Table 1に示す。

Table 1: Decimation Rate D Dependence on the Jitters

Measurement Conditions	Std. Dev. (Whole)	Std. Dev. (Central Part)	Num. of samples
D=1000, N=900	254 (ps)	171 (ps)	163,233
D=600, N=500	221 (ps)	139 (ps)	68,321
D=300, N=200	201 (ps)	131 (ps)	71,249

予想通り D が小さくなるほどジッタが低減している。ジッタを小さくするためにはできるだけ D を小さくしたいが、D を小さくするとネットワーク・トラフィックが増えることになる。また Version 1 の FPGA ロジックでは、小さな値を指定するとシステムがハングアップしてしまうため、D=600～1000 が現実的であると考えられる。これは当初想定していた値よりも大きな値であった。

3.3 スレーブ PC 側の遅延量変更機能の確認

スレーブ側に与えるターゲットバンチアドレス K、Fine delay 量 F の変更機能が問題なく働いていることを確認するために、これらを変えながら出力信号の平均値を測定した。結果を Table 2 に示す。オシロスコープのトリガはマスターPC に入力しているゼロ番地信号とした。

K を 0 から 2435 に変更すると1バンチ分＝約-1.9ns 出力が移動するはずだが、期待通り 1.9ns 早く出力されていることがわかる。また Fine Delay 量を+4.0ns としたところ期待通り 4ns 遅くなった。スレーブ PC 側の遅延量変更機能は問題なく働いていることを確認した。

Table 2: Delay Control Functions of the Slave PC

Measurement Conditions	Ave. (ns)	Num. of samples
D=600, N=500, K=0, F=0	12.28	176.7k
D=600, N=500, K=2435(-1.9ns), F=0	10.37	110.5k
D=600, N=500, K=0, F=2047(4.0ns)	16.27	131.8k

3.4 White Rabbit Switch 間 SMF の長さ依存性測定

White Rabbit はノード間のリンク遅延を動的に補正するため、出力時刻は SMF の長さに依存しない。そのことを White Rabbit Switch 間の SMF の長さを変えて確認した。具体的には、Switch 間の SMF の長さを 1m と 100m に変え、遅延量や出力周波数などの設定は全く同じにして、基準となるマスターPC 側に入力しているゼロ番地信号からの遅れの平均値を計測した。併せてジッタの最小・最大値や標準偏差も測定した。結果を Table 3 に示す。期待通り、SMF の長さには全く依存せずに同じ時刻に出力していることが確認出来た。

Table 3: SMF Length Dependence on the Output Signal

Measurement Conditions D=600, N=500, K=0, F=0	Ave. (ns)	Min. (ns)	Max. (ns)	Std Dev (ns)	Num. of samples
SMF 1m	4.75	-2.34	10.23	0.45	118.8k
SMF 100m	4.68	-2.11	10.86	0.47	600.8k

4. 課題の調査と対策

3 章の調査の結果、Version 1 の検証システムには以下に示す課題があることが分かった。

1. スレーブ PC からの出力信号が時々抜ける
2. 最大±5ns 程度の大きなジッタが発生している

3. システムの設定で D>N という制約がある

1 と 2 は実機に適用する場合にシビアな課題であるため、原因の調査と解決を図ることにした。

4.1 スレーブ PC からの出力抜け

詳細な調査の結果、スレーブ PC からの出力抜けは、マスターPC からの出力時刻 TN と間引きレート D の送出がソフトウェアによって行われていることに起因していることが分かった。それは、出力抜けが起こっている時間が間引きレート D と周回時間 4.8μs の積の倍数となっていること、D の値が大きくなると起こりにくくなること、マスターPC の CPU に負荷を掛けると容易に出力抜けの事象が起こることから確実であると考えられる。

この事象を解決するには、当初目指していた通り、マスターPC からのデータ送出を FPGA ロジックによって行う必要がある。幸いこのタイミングで Etherbone Master Core が（サンプル版という条件付きではあったが）OHR から提供されることになったため、Version 2 にてこの改修を行うこととなった。

4.2 最大±5ns 程度の大きなジッタの発生

この事象の原因は、FMC DEL が発熱により非常に高温になることだと考えられる。3.1 節の計測の後、使用していた FMC DEL が1枚故障したため新しい FMC DEL を用いてジッタ計測を行ったところ、Table 4 および Figure 7 のようになった。Figure 6 のような大きなジッタが発生していないことがわかる。

Table 4: Result of the Jitter Measurements with a New FMC DEL

Measurement conditions	Ave. (ps)	Min. (ps)	Max. (ps)	Std. Dev. (ps)	Num. of Samples
D=300,N=200	20.4	-522	595	127.7	121.2k

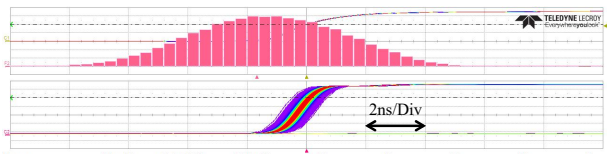


Figure 7: Hardcopy of the oscilloscope display at the jitter measurement with a new FMC DEL.

その後暫くの間諸々の計測を行っていると、再び大きなジッタが発生するようになった。その時 FMC DEL は非常に高温になっていて、コア温度は 70 度を超えていることが分かった。OHR のサイトにおいて、FMC DEL が非常に高温になるため SPEC ボードに冷却用のファンを取り付けるという話が出ていたため、対処療法的に PC のカバーを外して扇風機で冷やしたところ、大きなジッタが出なくなっていた。FMC DEL を確実に冷却できるよう、OHR で公開されている設計書に従って SPEC ボードにファンを取り付けることとした。

5. システムの改良（Version 2）

4 章に示した課題を解決するため、以下のようなシステムの改良を行った。

5.1 FPGA ロジックによるデータ自動送信の実現

TN および D をソフトウェアで送信していたことによって生じていたスレーブ PC からの出力抜けを防ぐため、マスターPC 側の FPGA ロジックに Etherbone Master Core を組み込んだ（Figure 8）。赤線で囲まれた部分が Figure 4(a)から変更になった箇所で、Etherbone Master Core が組み込まれている。当初この部分は Etherbone Master Core だけで実装できると考えていたが、実際には Slave Core も組み込まないと実装が出来なかったため、両方を組み込んでいる。ここに Etherbone Master Core が組み込まれたことで、ソフトウェアのアシストなしで直接 SPEC ボードの SFP モジュールからスレーブ PC 側の Etherbone Slave Core へ自動的にデータ送信が出来るようになった（Figure 9）。

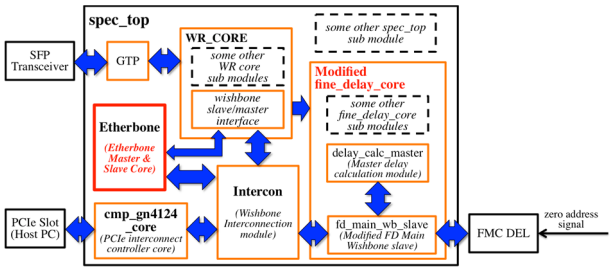


Figure 8: Block diagrams of the updated FPGA logic for a master PC.

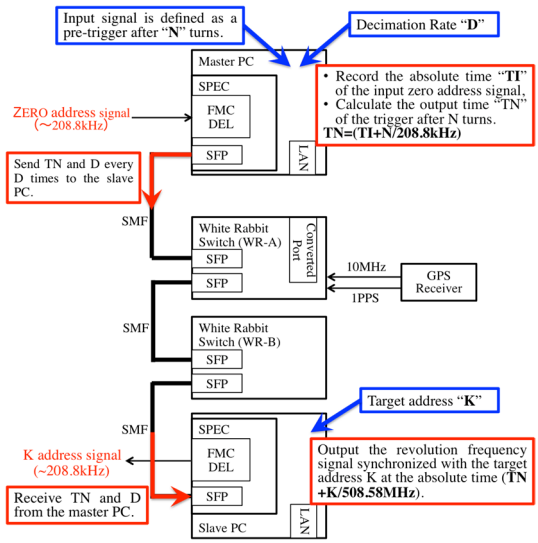


Figure 9: Block diagram of the updated system.

5.2 D<N の場合の FPGA ロジックの実装

Version 2 では、D<N の場合のロジックを組み込むことに成功した。D<N の場合、スレーブ側が最初のパルス出力を行う前に次のデータが送られてきてしまうため、D>N の場合に比べると実装が難しく、Version 1 では実装を見送っていた機能である。これにより D の値を小さくできるため、性能が非常に向上することが期待できる。

5.3 SEPC ボードに冷却用ファンを実装

FMC DEL の温度上昇によりジッタ性能の悪化が見られたことから、OHR の設計書に従って SPEC ボードに FMC 冷却用のファンを実装する改造を行った。

6. 改良システム（version 2）の評価試験

Figure 5 と同様のセットアップで Version2 の評価試験を行った。ただし現時点では、スレーブ PC 側の遅延量の設定値として K=0、F=0 以外の値を設定するとスレーブ PC からの出力が行われないか、開始から数十 ms 秒程度の極く短時間で出力が止まってしまう問題が生じていている。

6.1 出力信号のジッタ計測

スレーブ PC 側の出力タイミングの調整ができない状況であるため、行える評価試験は限られてしまうのであるが、幸い出力側の周波数の微調は可能な状況であったので、マスターPC 側の設定パラメータを変えてジッタの計測を行った。スレーブ PC 側のパラメータは、K=0、F=0、出力周波数微調整（-31.2ps）で固定として、マスター側の間引きパラメータ D を変えて測定を行った。結果を Table 5 に示す。また D=50、N=1000 の設定時のオシロスコープの画面のハードコピーを Figure 10 に示す。

Table 5: Results of the Jitter Measurements with the Updated System.

Measurement conditions K=0, F=0, Δf=-31.2ps	Ave. (ps)	Min. (ps)	Max. (ps)	Std. Dev. (ps)	Num. of samples
D=1000, N=1000	4.89	4.24	5.53	155.2	108.2k
D=50, N=1000	5.06	4.55	5.62	97.33	458.3k

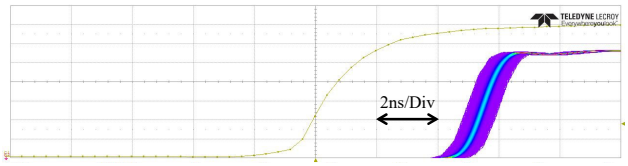


Figure 10: Hardcopy of the oscilloscope display at the jitter measurement with the updated system.

Version2 での改良で可能となった D=50、N=1000 というマスター側の設定によって、ジッタの標準偏差が 100ps を切ることに成功した。ジッタ性能の向上は、FMC DEL による補助的なパルス列出力機能の使用をできる限り控え、時刻ベースでの出力を増やすことで達成できることを示した結果であると言える。

6.2 改良システム（Version2）の安定性について

6 章の冒頭で述べた問題と関連があるかもしれないが、システム全体の安定性に課題があるようである。例えば現時点で唯一スレーブ側からの出力が可能な設定パラメータ K=0, F=0 の設定をした場合でも、D=1000、N=1000 の設定で 18 時間以上止まらずに動作を継続していた場合もある一方で、同じ設定でも1時間もしないうちに止まってしまうこともある。試験の途中でスレーブ PC がハングアップすることもある。一方で、マスター側がハングアップを起こす事象は現時点までで発生していない。2.5 の冒頭で述べた問題と併せて原因調査などを継続する予定である。

7. まとめ

本研究では、高精度時刻同期技術をベースとしたトリガ・タグ情報配信システム構築に向けて、SPring-8 蓄積リングの特定バンチに同期した 209kHz の周回周波数信号を生成するための検証システムを開発した。

初期システム（Version1）では、ソフトウェアのアシストによってマスター側からの時刻データの送出を行っていたためスレーブ側での出力に歯抜けを起こすことがあるが、D=300 での出力ジッタは標準偏差で 127ps を実現した。現実的な間引きレートの設定（D=600~1000）においては、出力ジッタは 140~170ps(1σ)程度であった。

改良システム（Version2）では、ハードウェアによる時刻データの送出に成功した。これにより速い繰り返し周期でも問題なく動作できるようになり、D=50 の設定で出力ジッタは 97ps(1σ)程度と 100ps を切る値を実現出来た。ただし Version2 は実際に利用するためには課題があり、現在デバッグを継続中である。

開発にあたっては OHR で提供されるハードウェア、IP コアを活用した。本開発で新たに Modified Fine Delay Core を開発し、マスター側では 209kHz 入力信号のタイムスタンピング、出力時刻の計算、間引きレートに従った出力時刻データの送信機能を、スレーブ側ではマスター側から送られた出力時刻と間引きレートに従って FMC DEL の制御を行う機能を実装した。また改良システム（Version2）では、新たに提供された Etherbone Master Core を実装して、FPGA 内部から時刻データや間引きレートなどの情報を送出する機能を実現した。

構築した検証システムでは、時刻と間引きレートというデジタル情報を配信することで実際に 209kHz のトリガ信号を出力することに成功した。タイミングの調整は、時刻情報を加工することで行えるので、ソフトウェアを用いて非常に容易に行える。マスターとスレーブの間はネットワークケーブル1本で接続することができるため敷設も非常に容易であるうえ、ネットワークスイッチによる信号出力ノードの増設も容易に行うことができる。遅延量の調整などを行う際には、ノード間を接続する光ファイバの長さやスイッチの存在、温度等による光ファイバ経路長の変化などは考慮する必要がない。

Version 2 システムのデバッグ終了後、実際にある放射光利用実験に試験的に適用する予定である。また、機会を得ることが出来れば、さらなる低ジッタ化、入力信号の周波数変化への自動追従機能、高繰り返し出力への対応情報のブロードキャスト／マルチキャスト送信、UTC を用いた運用等の高度化を行いたいと考えている。

参考文献

[1] http://www.ohwr.org/projects/white-rabbit
[2] http://www.ohwr.org
[3] http://www.ohwr.org/projects/spec
[4] http://www.ohwr.org/projects/fmc-delay-1ns-8cha