修　士　論　文

高精度計測・高速収集機能を備える
無線型地震観測システムの実装と評価

Implementation and Evaluation of
Wireless Earthquake Monitoring System
with High Fidelity Sampling
and High Throughput Data Transfer

指導教員　　**森川 博之 教授**

東京大学大学院工学系研究科
電気系工学専攻

氏　名　　**37-106444　角田 仁**

提　出　日　　**2012年2月8日**

目次

図目次

表目次

第１章

序論

1.1 背景と目的

科学技術が発達した現在においても世の中には完全に解明したとは言いがたい現象は数多く存在する．地震もその中の 1 つであり，実際に阪神淡路大震災，新潟県中越地震，スマトラ島沖地震と直近でも地震は人類に甚大な被害を与えている．このような地震に対して日本では阪神大震災を機に地震モニタリングシステムの高密度が図られ，日本全島を 20 km 四方でメッシュ状に地震センサが配備された．しかしながら 20 km 四方に 1 つのセンサでは地震が建物に与える影響を検出することは不可能であり，地震モニタリングシステムのさらなる高密度化が求められている．

地震モニタリングシステムの高密度化は次の 2 つの理由により困難となっている．1 つ目は地震を計測する加速度センサが高価であることである．地震を科学的用途で計測するためには地震によって生じる加速度波形の振幅や位相を正確に捉える必要がある．このため，現在の地震モニタリングシステムで使用されている加速度センサは，地震観測専用に開発されたものであり，1 つあたり数十万円と非常に高価である．2 つ目はセンサの設置コストである．現在の地震モニタリングシステムはセンサからのデータを有線で収集しているため，センサの設置場所に制限があることに加え，既存の構造物への取り付けには改修工事が必要となり現実的でない．このような問題に対して筆者らは集積化された加速度センサを用いた無線センサネットワークを構築することで地震モニタリングの低コスト化・高密度化を目指している [1].

本論文では，実展開に必要な機能を有する無線型地震観測システムの構築を通じて，高精度な同期タスク実行機構を用いたアプリケーションについて，開発時の課題と解決のための設計を示す．実展開に必要な機能には，UTC 時刻と同期した高精度な同期タスク実行，センサデータの常時収集，センサノードの状態収集，ノードへの指令の拡散があげられる．これらの機能の各々については研究がなされているが，これらの機能ををを 1 つのシステムに集約したものはない．

無線センサネットワーク上での，高精度な同期タスク実行機構を用いたアプリケーションの実装は，CPU の処理能力の制約，タイマ資源の制約，無線通信の帯域の制約により困難となる．無線センサノード高密度・広範囲の実空間の情報を収集するため，コスト面から省資源性が求められているためである．本論文はこれらの制約から生じる課題を具体

的に明示し，それらの課題を解決するための設計手法を示す．また，評価において提案した設計手法にて高精度な同期タスク実行が実行できること，高精度な同期タスク実行中に複数のノードからのセンサデータの常時収集が可能であること，他の計算が可能であることを示す．

1.2　本論文の構成

本論文の構成は以下の通りである．まず，第 2 章において加速度サンプリングと時刻同期の関連研究について述べる．既存の高周波数の加速度サンプリングシステムの特徴を述べ，そこで，実展開に向けた地震モニタリングシステムの要件を全て満たすシステムの評価が行われていないことを明らかにする．また，時刻同期は地震モニタリングシステムの精度をに大きく影響するものであることから実装評価に主眼を置いた時刻同期の関連研究とそれらの誤差の議論について述べる．また，時刻同期は地震モニタリングシステムの精度をに大きく影響するものであることから同期方法とそれらの誤差の議論について述べる．第 3 章では，第 2 章で明らかとなった要件を満たす地震モニタリングシステムの設計及び実装を行う．この実装を通じて高精度な同期タスク実行機構を用いたアプリケーションについて，開発時の課題と解決のための設計を示す．第 4 章では，評価を行う．サンプリングジッタ，高精度な同期サンプリング実行時の通信のスループット，システムの CPU 使用率を明らかにし，システムが実用的かつ現実的であることを示す．最後に第 5 章においてまとめとする．本研究の主たる成果について述べ，今後の課題について言及する．

第 2 章

関連研究

2.1 はじめに

本章では，加速度サンプリングと時刻同期の関連研究について述べる．既存の高周波数の加速度サンプリングシステムの特徴を述べ，そこで，実展開に向けた地震モニタリングシステムの要件を全て満たすシステムの評価が行われていないことを明らかにする．また，時刻同期は地震モニタリングシステムの精度をに大きく影響するものであることから同期方法とそれらの誤差の議論について述べる．

2.2 加速度モニタリング

無線センサネットワークを用いて多点で振動観測を行う取り組みは，いくつか進められている [2][3][4]．以下では，時刻同期を行い，多点での波形レベルでの相関を求めることを目的としたものに議論の対象を絞る．そのため，Wisden [4] など，厳密な時刻同期を行わないものは議論の対象外とする．

Allen らは，無線センサネットワークを利用した科学用途の計測を目指し火山を対象に振動および音波のサンプリングを行っている [3]．[3] は，FTSP [5] を利用して時刻同期を行い，活火山による地震の発生から発火点の推定精度について検討を行っている．しかしながら，サンプリング精度に関する検討は行われていない．

Kim らは，Golden Gate Bridge に 64 台のセンサノードを設置し，常時微動の計測を実現している [2]．Kim らは，サンプリングジッタの問題に対して，計測前に無線通信等，計測に不要な割り込み要求を不許可とすることで，低ジッタでのサンプリングを実現している．しかしながら，この手法では図 2.1 のように無線通信に関わる割込みを禁止しているため，低ジッタサンプリングを行いながら無線通信を行うことはできない．

高周波サンプリングを行うセンサネットワークとして，CounterSniper のように音波を利用して測距や位置検出を行うアプリケーションが挙げられる．CounterSniper [6] は銃の発射音を検出することによって，スナイパーの存在位置を推定するアプリケーションである．CounterSniper では FPGA を利用して本稿での対象よりもはるかに周波数の高い 1 MHz というサンプリング周波数で音波のサンプリングを行いイベントの検出を行っている．しかしながら，イベント検出時の時刻を検出することが目的となっており，本稿のように波形レベルでの相関に関する検討は行っていない．

Aoun らは，TIA（Tick Interrupt Alignment）と呼ばれる，割込み同期機構を示している [7]．時刻同期を利用して，次回のサンプリングタイミングを決定し，割込みを同期させる手法は本機構と同様である．しかしながら，TIA では，他の割込みによる影響や，アトミックセクションなどの影響を考慮していないため，図 2.2 のようにタスク実行のジッタが発生してしまう．

[8] では，図 2.3 に示される細粒度にタスクを実行する時間スロットを用意し，タスク実行を時間的に分離することで，サンプリングジッタを除去しつつ無線通信を行うことができる．本手法を PAVENET OS [9] へ実装し，初期的な評価を示した．本手法は，一般的な CPU が具備する割込み許可フラグを，ソフトウェアの機能のみで制御することで実現可能なため，TinyOS[10] や IRIS mote などを利用した一般的なプラットフォームにも

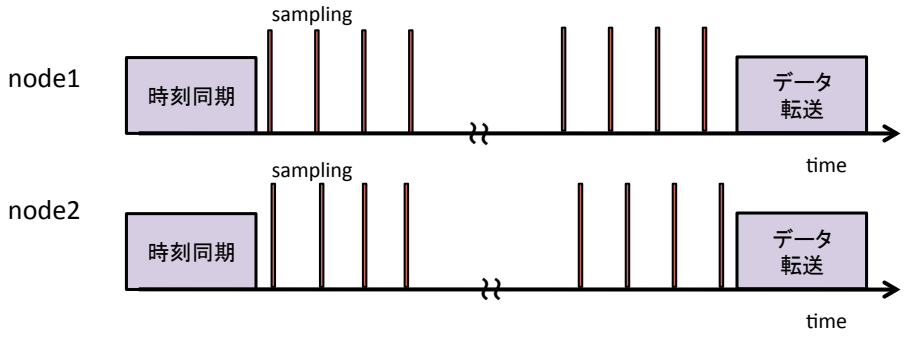

図 2.1　Kim らの加速度モニタリングシステムの動作

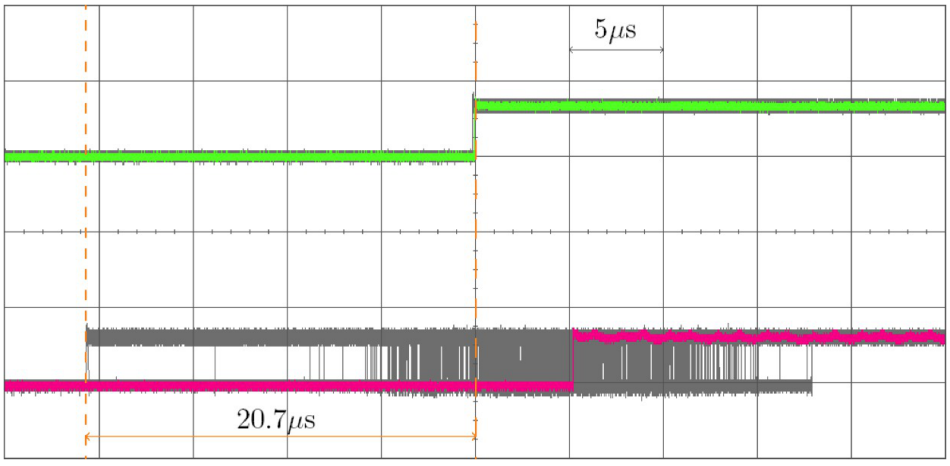

図 2.2　TIA のタスク実行のジッタ

適用可能である．本稿では本方式を TinyOS 上に実装するとともに，通信性能なども含めて評価を示す．

2.3 時刻同期

ここでは，無線センサネットワークの時刻同期に関する既存研究について，誤差推定と理論的解析の観点から議論し，本稿の位置付けを明確にする．時刻同期研究は，分散システムの理論研究においても広く検討されているが [11][12]，これらでは信頼性のある通信チャネルやトポロジの把握など，電波環境が動的に変動し，計算資源の制約も厳しい無線センサネットワークでは適用できない条件を前提としている．このため，これらの研究は議論の対象外とし，無線センサネットワークにおける実装評価に主眼を置いた研究を対象に論じる．

FTSP [5] は，リファレンスポイントの，シングルホップで生じる誤差の削減に取り組んでいる．リファレンスポイントとは，時刻同期パケットの送受信タイミングでの，ネットワーク時刻と自ノードの時刻とのペアを指す．また，複数のリファレンスポイントに対して線形回帰を行って時刻を推定することで同期精度を向上させている．さらに，推定した時刻をフラッディングベースの簡略な一方向の通信でネットワーク全体に転送することで，ネットワーク全体の時刻同期を実現している．FTSP は，[3][2][6][13] などの実際

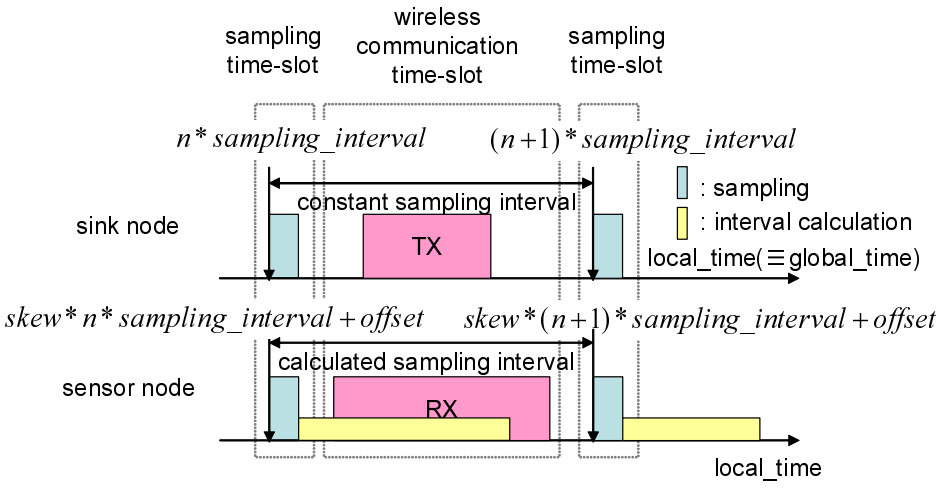

図 2.3 鈴木らの手法による同期サンプリング機構の動作

のアプリケーションでの利用例も報告されており，標準的なプロトコルとして知られている．

FTSP の提案以降，時刻同期プロトコルは，RITS [14]，KFMP [15]，PulseSync [16]，VHT [17]，TIRP [18]，Glossy [19] など，FTSP の拡張を主として進められている．

これらの先行研究では，精度や消費電力の観点から様々な改良が試みられている．しかしながら，「運用環境での時刻同期誤差推定」について論じた先行研究は，筆者らの知る限り存在しない．

KFMP，PulseSync，TIRP などでは，理論的解析も行われており，ホップ数やパケットロスを考慮した誤差モデルが示されている．しかしながら，KFMP，および TIRP は，誤差を表す確率変数の依存関係を正しく扱えていない．また，PulseSync は確率変数の依存関係を正しく扱っているものの，オーダーでの議論に止まっており，定量的な推定は不可能である．

RBS [20]，TPSN [21]，FTSP [5] など，比較的初期の研究では，時刻同期パケットを転送するための，タイムスタンプの精度を，いかにして向上するかについて検討されている．VHT [17] では，細粒度な消費電力の高いクロックと，粗粒度で消費電力の低いクロックとを併用することで，省電力性を維持しつつ，高精度なクロックを利用する方法が示されている．TCTS [22] では，ソフトウェア的に温度補償を行うことで，時刻同期誤差を低減する手法について示されている．

PulseSync [16] は，本稿と同様に，転送のアルゴリズムを分離することで，ホップ数に対する誤差の増大を低減している．しかしながら，誤差のモデル化は，オーダを示すに止まっている．また，LPL（Low Power Listening）型の MAC の利用などを想定しておらず，省電力な MAC プロトコルを利用した場合に，精度にどの程度影響があるのかは示されていない．さらに，温度依存性の低減についても有効であると示唆されているものの，理論的解析や実験的評価は行われていない．

TIRP [18] は，FTSP のように非同期に同期パケットを転送する場合に，FEV（Frequency Error Variance）と呼ばれる時刻同期パケットのためのルーティングメトリックを規定し，これに基づいて時刻同期パケットの転送を行うことで，温度依存性の伝播の影響を低減可能であることを示している．本稿では，分離して転送する場合には，この誤差を CPU の 1 クロック程度に低減可能であり，このようなルーティングを行わずとも，周波数変動の影響の伝播を大きく除去できることを示す．

ODS [23] では，誤差推定を行い，適応的に時刻同期間隔を調整することで，必要な精度を満たしつつ，可能な限り省電力にする方式が示されている．これに対して，本稿では，一定間隔で時刻同期パケットを転送することで，ノードの省電力を簡単に実現できることを示す．

[19] では，IEEE 802.15.4 に準拠した無線通信においては，同一のデータが 0.5 us 以内の時間差で複数のノードから同時に送信される場合には，干渉が建設的になることに着目し，省電力に高速なフラッディング方式を可能とする Glossy と呼ばれる手法を示している．Glossy は，CC2420 を備えるセンサノードでこれを示しているが，他のノードへの移植可能性については示していない．特に，CC2420 と同様に広く利用されている無線通信モジュールである RF230 は，デジタルドメインが 1 MHz で駆動されており，1 us の量子化誤差が存在するため，利用が可能かは不明瞭である．本稿では，このような特殊な無線通信を行わずとも，時刻同期精度に影響を与えない程度に十分に高精度で，十分に省電力なフラッディングが可能なことを示す．

KFMP [24]，ACES [25] では，カルマンフィルタを時刻同期に適用する手法について示している．しかしながら，ACES では，マルチホップネットワークへの適用について示されていない．また，KFMP では，カルマンフィルタを，時刻同期パケットの転送にも利用しているため，誤差が指数関数的に増大するという問題がある．

2.4 おわりに

本章では，加速度サンプリングと実装評価に主眼を置いた時刻同期の関連研究について述べた．実展開に向けた地震モニタリングシステムの要件を全て満たすシステムの評価が行われていないことを明らかにした．

第 3 章

地震モニタリングシステムの設計と実装

3.1 はじめに

本章では，実展開に向けた無線型地震観測システムの設計と実装を示す．本システムは機能として， UTC 時刻と同期した高精度な同期サンプリング中に，センサデータの常時収集，センサノードの状態収集，センサノードへの指令拡散を行うことを特徴とする．UTC 時刻との同期は他ネットワークの地震モニタリングシステムや，他システムの波形データと比較のための機能である．センサデータの常時収集は，偽陽性，偽陰性の少ない建物の揺れの検知のための機能である．センサノードの状態収集は，通信の品質やリアルタイムにセンサの故障を診断のための機能である．センサノードへの柔軟な指令拡散は，各々のノードの常時収集や状態収集の ON/OFF などの動作管理や，システム全体の動作モードの変更やリセットのための機能である．本章ではこの中でも実装と設計の要点となる，高精度な同期サンプリング，GPS 受信器の出力する UTC 時刻との同期，常時収集について詳細を述べる．

本章の構成は以下の通りである．3.2 では実展開に向けた無線型地震観測システムの全体像を述べる．3.3 では高精度サンプリング，3.4 では GPS との同期，3.5 では常時収集の設計と実装を述べる．

3.2 全体像

地震モニタリングは，構造物に設置された加速度センサによって地震によって生じる構造物の加速度を計測するための技術である．図 3.1 に示すように，本システムは 1 つの部屋に設置された 1 台のシンクノードと 7 台のセンサノードから構成され，地震による空間の歪みを計測する．現在の構造物は，図 3.1 に示される層間変形角（story drift）が 1/200 を超えない範囲で安全性を保証しなければならない，という指針のもと構造設計が行われている．本システムでは，地震中の加速度を計測し，加速度から時刻歴における変位を求め，ノード間の変位の相関から層間変形角を求めることによって構造物の健全性を判断する．

筆者は，図 3.2 に示される，UTC 時間と同期した高精度な同期サンプリングを行いながら，センサデータの常時収集とセンサノードの状態収集，センサノードへの指令拡散を行うシステムをアプリケーションを想定している．コストを考慮して，MICA mote の

ような処理性能，通信速度は低いものの低コストで実現できる無線センサノードを利用する．

動作の概要

具体的な処理フローは以下のようである．シンクノードは定期的に時刻同期パケットをブロードキャストし，センサノードは時刻同期パケットを受信し時刻同期を行う．また，地震モニタリングにおいては，発生から終了までのすべての地震波をセンシングする必要があるため，地震検出後にセンサノードをウエイクアップさせるような省電力制御は適用できない．このため，すべてのノードは常に加速度センシングを行った状態で地震を待機する．サンプリング周波数は，従来型の有線の地震モニタリングシステムに倣い 100 Hz とした．

本システムではセンサデータの常時収集を行う．センサデータの常時収集を行うことで，揺れの検出についての偽陽性（False positive），偽陰性（False negative）を抑えることができる．前者は，建造物内での人の動きなどによる振動の発生や，センサのノイズによって発生する．後者は，地震の揺れが軽微であるときに発生する．そこで本システムでは，100Hz でサンプリングしたセンサデータの常時収集を行う．前述の通り，全てのセン

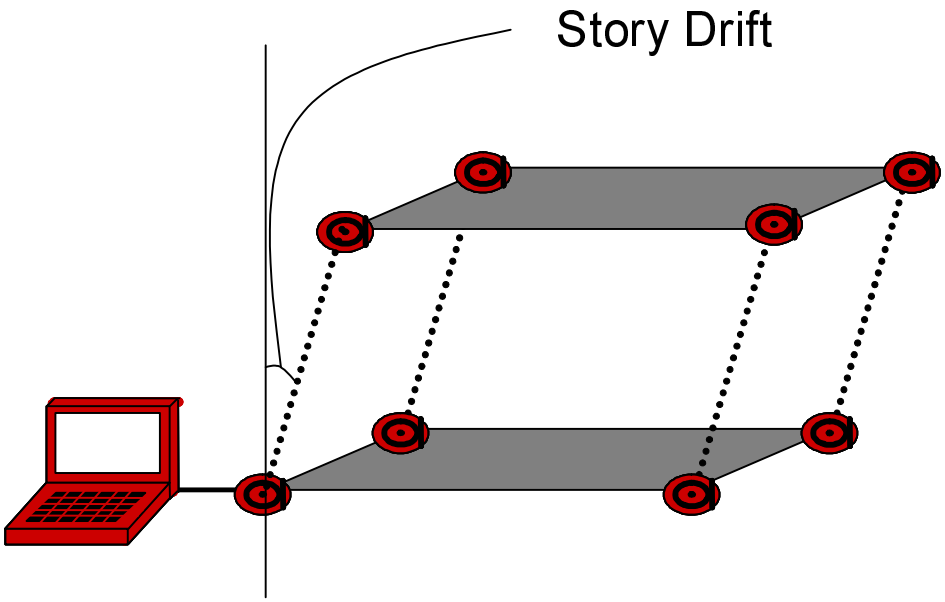

図 3.1　地震モニタリングシステム概要

サデータの信頼性のある収集は現実的でないことから，指定したノードから，end-to-end の信頼性のある収集を実行する．また，常時収集の収容台数を増やすために，必要に応じてリサンプリングを行い，収集するセンサデータ量を低減させる．

常時収集に加え本システムではセンサノードの状態収集を行う．センサノードののネットワークのトポロジ情報や，センサボードのオフセットやセンサデータの二乗和の収集を行う．トポロジ情報の収集は無線センサネットワークで問題となるリンク品質の確保に，用いられる．また，センサボードのオフセットやセンサデータの二乗和はセンサノードの故障検知や，定常的なノイズのパワーを推定することに用いられる．

実展開に向けたシステムにおいて，ノードの情報を収集するのみではネットワークを管理することはできない．そこで，本システムでは拡散プロトコルを用いて各センサノードに対して指令を拡散する．これを利用して，各ノードについて常時収集や状態収集の ON/OFF の設定や，ネットワーク全体の一括収集モードへの動作の変更やリセットを行うことができる．

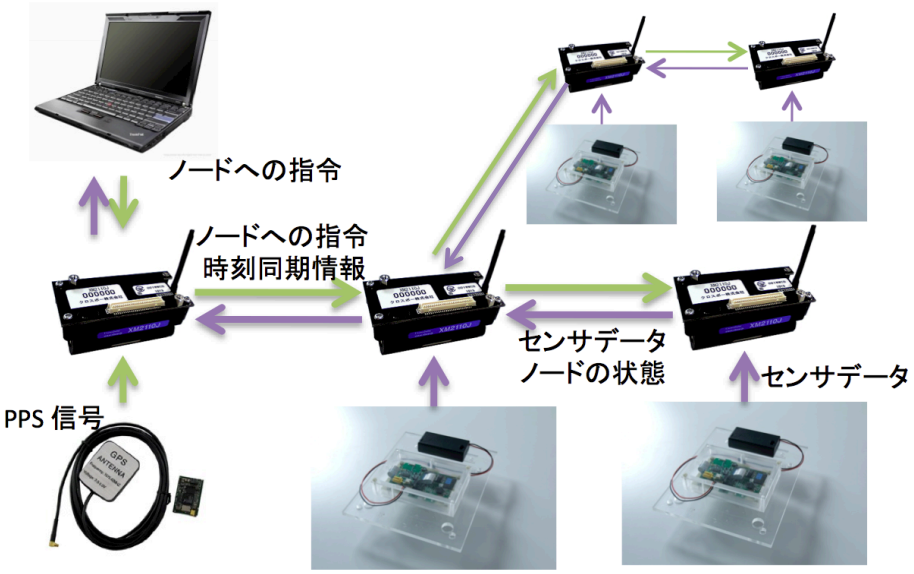

図 3.2

3.3　高精度サンプリング

無線センサネットワークにおけるサンプリングジッタ

地震モニタリングでは，加速度波のサンプリングを行わなくてはならない．この際，取得したサンプルの精度を確保することが重要となる．例えば記録したサンプル列を用いて健全性判断を行う場合，サンプルの精度を確保できなければ偽陽性や偽陰性の確率が高くなってしまう．

離散信号システムでは，サンプリング誤差，センサノイズ，量子化誤差の 3 つの誤差要因が存在する．サンプリング誤差は，図 3.3 に示す通り，想定したサンプリングタイミングと実際にサンプリングが行われたタイミングのズレによって発生する誤差であり，信号の周波数成分とサンプリングジッタに依存する．センサノイズはセンサの出力に含まれるホワイトノイズであり，センサの種類に依存する．量子化誤差は AD 変換によって生じる誤差であり，AD コンバータのビット長に依存する．

これらの誤差要因の中で，センサネットワークによる計測では，分散システムであること，複数のタスクを並列に実行するという 2 つの特徴に起因して，サンプリング誤差が顕在化する．前述のように，ナイキストのサンプリング定理に従ってサンプリングを行えば，取得したサンプルから元の波形を完全に復元可能である．しかしながら，サンプリン

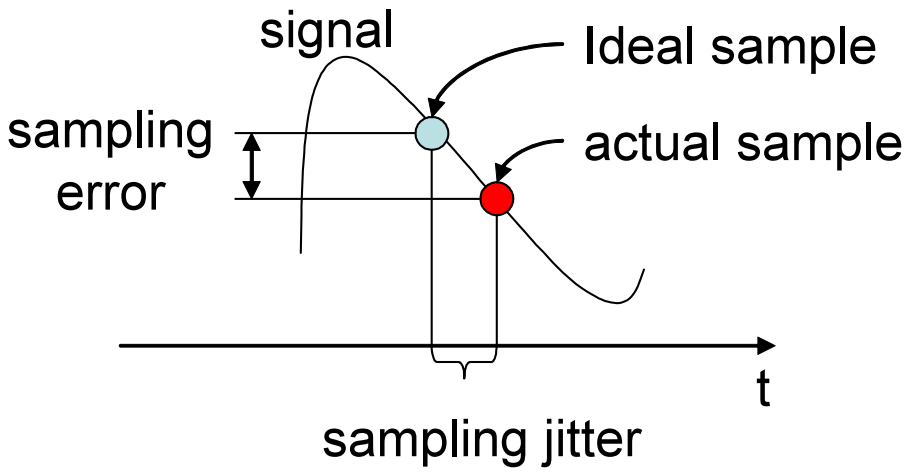

図 3.3　サンプリング誤差

グ定理には，「一定間隔のサンプリング」という重要な前提があり，元のアナログ波形を復元するためには，この前提を満たさなければならない．

有線接続の計測システムではセンサのみが分散して配置されているため，サンプリングのタイミングはすべての観測点で同一である．さらに，地震観測に特化したハードウェアで構成されるため，スケジューリングによる不確定さは発生せず，サンプリングジッタは水晶振動子の温度特性等による変動のみである．地震波は有意な周波数成分が高々 20 Hz 程度であるため，水晶振動子の温度特性のように ppm のオーダでの変動は問題とならず，一定間隔かつ同一タイミングでのサンプリングと見なすことができる．

これに対して，センサネットワークはそれぞれのノードが固有のシステムクロックを有する分散システムであり，サンプリングタイミングは各センサノードが独立して指定しなくてはならない．時刻同期を行うことで，それぞれのセンサノードの時刻軸を 1 つの時刻軸に変換することは可能であるものの，時刻同期には誤差を伴う．これに加えて，センサノードでは複数のタスク要求が 1 つの CPU に対して同時に発生しうるために，一定間隔でサンプリングを行おうとしても，スケジューリングの不確定性によってサンプリングジッタが生じてしまう．

無線センサノードの並列実行モデル

センサノードはタイマ，AD コンバータ等のデバイス制御，無線通信処理，計算処理など複数のタスクを並列に実行しなければならない．センサノードは，これらの複数のタスクを，CPU への割り込み要求に対応する形で CPU 時間をスケジュールすることによって並列性を実現している．

地震モニタリングでは，サンプリングジッタの削減のために，サンプリングタスクと時刻同期のためのタイムスタンプを行うタスクに関して不確定な遅延を除去する必要がある．サンプリングタスクの遅延は直接サンプリングジッタとなり，また，タイムスタンプを行うタスクの遅延は時刻同期の誤差を生むことから，同様にサンプリングジッタを発生させる．

同時に複数の割り込み要求が存在する場合には，以下の 2 つの要因によりタスクの実行開始時間に不確定さが生じてしまう．1 つ目の要因は，割り込み要求が同時に発生しうることである．割り込み要求が同時に発生してしまうと，他のタスクに CPU 時間が奪われ

表 3.1 TinyOS を利用した場合のサンプリング間隔

Task	Default Timer			Micro Timer		
	Max Cycle (ms)	Min Cycle (ms)	Jitter (us)	Max Cycle (ms)	Min Cycle (ms)	Jitter (us)
Sampling + Listening	11.735	7.794	3,941	10.032	10.000	32
Sampling + RX	11.770	7.448	4,322	10.062	10.000	62
Sampling + TX	12.342	7.187	5,155	10.671	10.000	671

てしまい実行時間を保証することができない．2 つ目の要因は，割り込みを不許可とするクリティカルセクションが存在することである．クリティカルセクションはデータの一貫性を維持するために存在し，CPU の割り込み許可／不許可フラグを制御することで実現される．他のタスクがクリティカルセクションを実行していた場合には，割り込み要求が発生しても，対応するタスクへと即座に制御を移すことができない．

スケジューリング遅延は，無線センサノードの処理性能が低いことから，不確定性が ms のオーダとなり測定精度に大きな影響を与える．筆者らは，一般的な状況におけるサンプリングジッタがどの程度であるのかを評価するために TinyOS [10] を利用して簡単な実験評価を行った．表 3.1 は TinyOS の Timer モジュール [26] と MicroTimer モジュール [27] を利用して 10 ms ごとにサンプリングを行うプログラムを実行した時のサンプリング間隔である．Timer を利用した場合は対応するタスクはスケジューラから実行され，MicroTimer を利用した場合は ISR（Interrupt Servie Routine）で直接実行される．表 3.1 に示すとおり，複数の割り込み要求が同時に発生する場合には，MicroTimer を利用した場合でさえも 1 ms に近い不確定な遅延が発生している．

以上のことから，無線センサネットワークで地震モニタリングを実現するためには，時刻同期を行いつつも一定間隔でのサンプリングを可能とするスケジューリング手法を考えなくてはならない．

高精度な同期サンプリング機構

全体像書かないといけない図 3.4 に○○をを示す

時刻同期プロトコル

時刻同期プロトコルに同期サンプリングの初期化を行う機能を追加した．時刻同期プロトコルには TinyOS の FTSP を用い，図 3.5 時刻同期パケットに送信時のサンプリング番号と前のサンプリングから経過した時間を追加する変更を行った．最初の時刻同期パケットを受信した際に，時刻同期の計算をするとともに，パケットの内容と送信時刻と現在時刻を用いてサンプリング番号と次サンプリングタイミングを計算する．

サンプリングタイミング決定アルゴリズム

すべてのノードで同期してサンプリングを行うためには，時刻同期を行ったうえで，その情報に基づいてサンプリングタイミングを動的に計算する必要がある．センサノードが具備する水晶振動子には数 10 ppm 程度の固有差は存在するものの，短期間では安定した周波数で発振する．本機構では，FTSP [5] と同様に，この固有差を線形回帰によって補正することで時刻同期を行う．それぞれのノードは localtime と呼ばれる固有の時刻を直接補正することなく保持し，時刻同期によって与えられる情報からネットワーク上で 1 つの globaltime（シンクノードの localtime）を推定する．

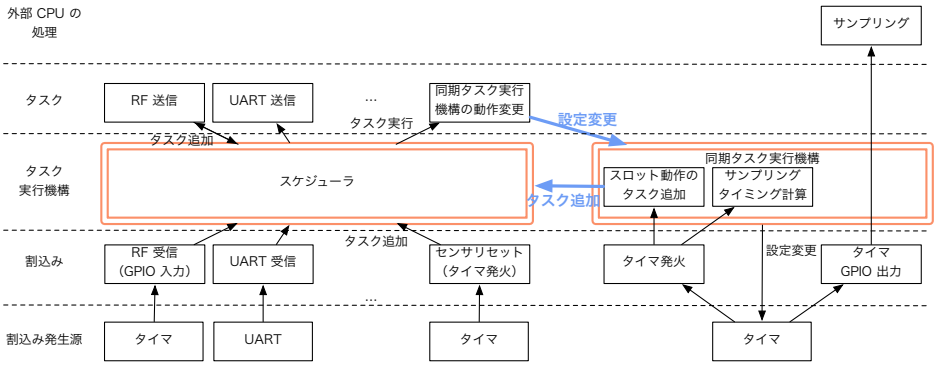

図 3.4　OS のスケジューラと高精度な同期サンプリング機構の全体像

rootID (2)	nodeID (2)	seqNum (1)	globalTime (4)	slotID (4)	slotTime (2)

図 3.5　時刻同期パケットのデータ

具体的な同期手順は以下の通りである．シンクノードは定期的に同期パケットを送信する．同期パケットには，n（4 B），globaltime(6 B) の情報が含まれている．globaltime は同期パケットが送信開始された直後のシンクノードの localtime であり，n は次にサンプリングされるデータに紐付けられるインデックスである．同期パケットを受信したセンサノードは，受信時のタイマ値とともに，$(localtime, globaltime)$ という形で保存する．次いで，これらのペアをあらかじめ定めた個数蓄えた後に線形回帰を行うことで，

$$localtime = skew_{gl} \times globaltime + offset_{gl}$$

という変換式の $skew_{gl}$ および $offset_{gl}$ を得る．$skew_{gl}$ は，それぞれの水晶振動子の発信周波数の比であり，$offset_{gl}$ はタイマ間のオフセットである．

ここで得た時刻同期情報に基づき，以下の計算を行ってサンプリングタイミングを決定する．割り込み要求を発生できるタイマは localtime だけであるため，globaltime から localtime に変換しなければならない．具体的には，n 番目のサンプリングを行うべき globaltime を GS_n，対応する localtime を LS_n，globaltime でのサンプリング間隔を GT (定数)，$n+1$ 番目のサンプリングと n 番目のサンプリングの時間間隔を LT_n として，

$$\begin{aligned} GS_n &= GT \times n \\ LS_n &= skew_{gl} \times GS_n + offset_{gl} \\ LT_n &= LS_{n+1} - LS_n \end{aligned}$$

という計算式に従ってサンプリング間隔 LT_n を算出し，LT_n を CPU に内蔵されるタイマコンペアレジスタに代入する．

マルチコアによる高精度サンプリング

センサボード

カットオフ周波数が 50 Hz のローパスフィルタ，16 bit の AD コンバータ，2 MB の SRAM および，CPU として PIC18LF8527 を具備した加速度センサボードを用いた．筆者らの開発した地震モニタリング用加速度センサボードを図 3.6 に，仕様を表 3.2 に示す．LIS3L02AQ のレンジを 2 G と設定し，16 bit の AD コンバータを選定した．これによって，震度 7 の巨大地震にも対応しつつ，震度 1 や震度 2 の微弱な地震も計測することができる．

地震モニタリング用加速度センサボードはセンシングおよび地震検知を主なタスクとしている．PAVENET モジュールからセンシング要求が入力されると，即座にセンシングを行い，下に示す地震検出アルゴリズムを実行する．取得した加速度値はセンサボードが具備する SRAM に蓄積され，地震検出時にシンクノードからの要求に応じてセンサ値を転送する．

実験を進めるなかで，図 3.7 に示すようにセンサ値が温度に依存して特性が変化することが明らかになった．これによって，時間の経過とともに，振動のない状態でも地震と誤

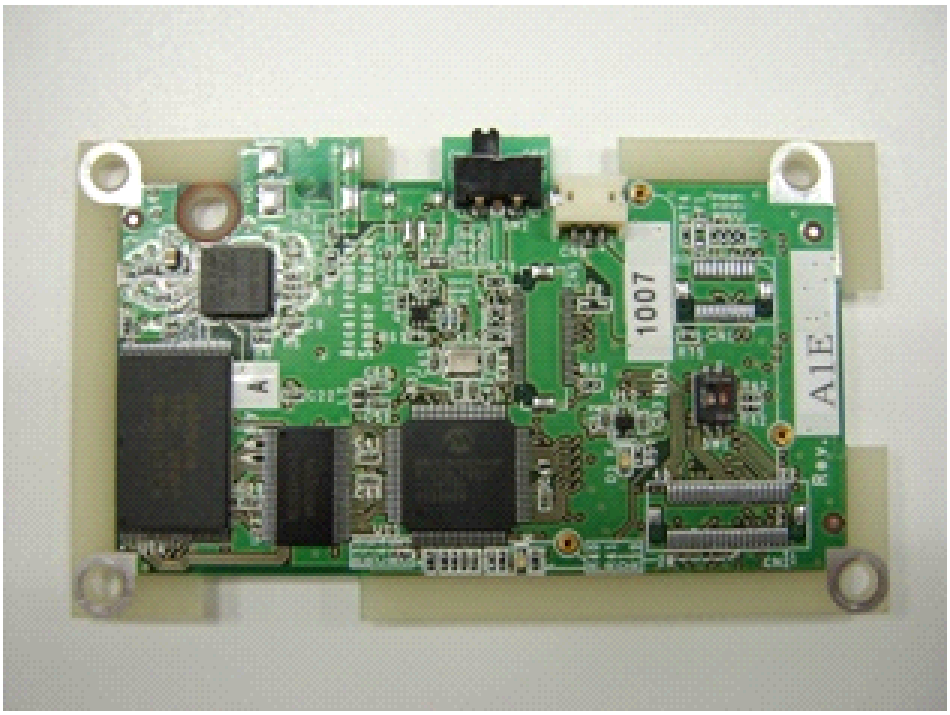

図 3.6　地震モニタリング用加速度センサボード

表 3.2　地震モニタリング用加速度センサボード仕様

Sensor	ST Microelectronics LIS3L02AQ
Filter	Low Pass Filter (50 Hz)
AD Converter	Texas Instruments ADS8341E (16 bit)
Storage	Renesas R1LV1616HSA (2 MB SRAM)
CPU	Microchip PIC18LF8527

検知してしまうという問題が生じた．この問題に対して，温度変化によるセンサ値の変動は地震の長さの間では無視できるとして，サンプリングごとにゼロ点補正を行うことで温度変化に伴う地震の誤検出を解決した．具体的には，図 3.8 に示すように，t=–23.03 s から t=–12.80 s までの 1024 サンプルの平均値をゼロ点として，t=–2.55 s から t=0.00 s までの 256 サンプルの RMS 値を計算することとした．

センサボードとの通信

センサノードとセンサボード間の通信は GPIO とシリアル通信を用いる．センシング後に常時収集を行うことを前提することと，センサデータのアクセスは逐次的であることを利用して通信回数を削減している．

はじめに，常時収集時の動作について説明する．センサノードはセンサボードの初期化時にサンプリングのインデックスを送信する．サンプリングのインデックスを受信すると，センサボードは常時収集モードにモードを変更する．初期化が終わり，サンプリングをする際にはセンサノードは GPIO を用いてセンサボードにサンプリングタイミングを通知する．センサボードはサンプリング直後にセンサデータを図 3.9 のパケットで送信す

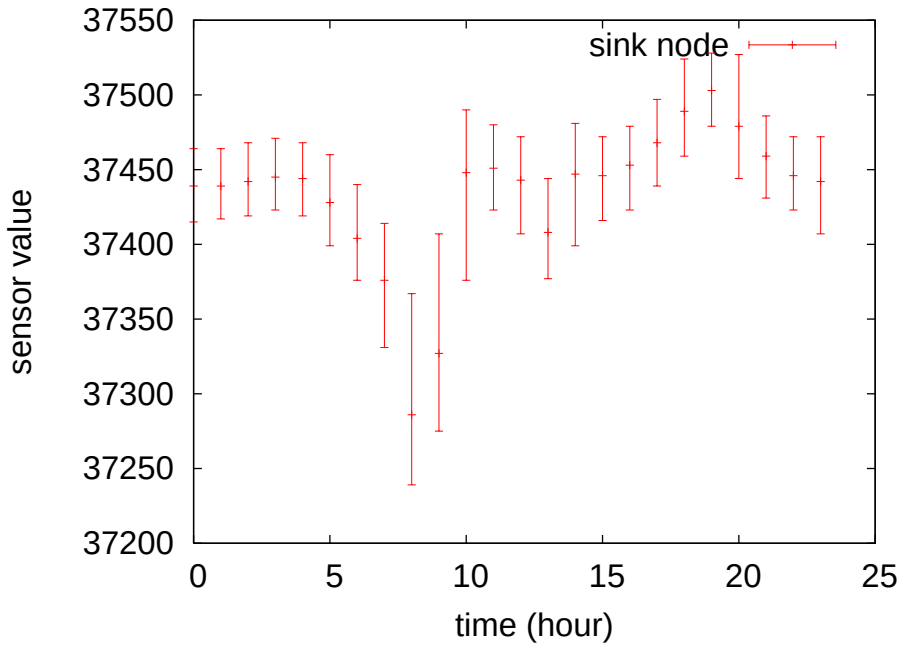

図 3.7　加速度センサの温度依存性

る．センサボードはセンサデータの送信後，一定回数のサンプリングごとにセンサ値のオフセットと，一定区間のセンサ値の二乗和を図 3.9 のパケットで送信する．

次に一括収集時の動作はについて説明する．センサノードは一括収集の開始にアクセス開始のインデックスをセンサボードに送信する．センサボードは，アクセス開始のインデックスを受信すると，一括収集モードに動作モードを変更する．センサボードのモードが変更後に，センサノードは次データの要求を GPIO の出力にて行う．センサボードは次データの要求を受けると，センサデータを送信する．一括収集モードは，サンプリングを止めてデータを一括収集するための動作であり，センサデータの送信後，一定回数のサンプリングごとにセンサ値のオフセットと，一定区間のセンサ値の二乗和は送信しない．

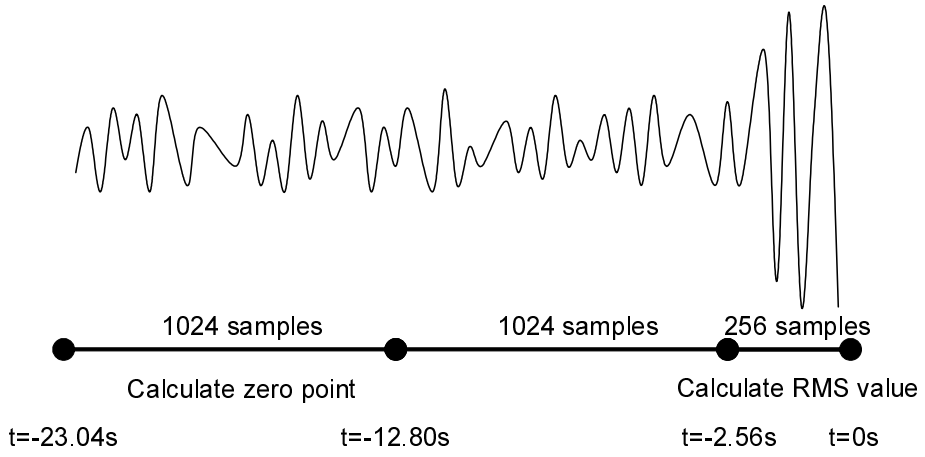

図 3.8　地震検知手法

command (1)	index (4)	data_x (2)	data_y (2)	data_z (2*3)	checksum (2)

command (1)	offset_x (2)	offset_y (2)	offset_z (2)	sum of square (4)	checksum (2)

図 3.9　センサボードとの通信のパケット

シングルコアによる高精度サンプリング

CTC 動作タイマ

スロット動作を実現するために，CTC (Clear timer oncompare match mode) 動作を行うタイマを実装した．CTC 動作をさせることで，サンプリングの開始直後にサンプリングタイミングの計算結果をタイマのコンペアレジスタに代入することで，次のサンプリングの開始時間にタイマが発火する．

また，他のモジュールが Timer1 を用いていることから，CTC 動作時にでも動作するように変更を行った．例えば，Timer1 は無線通信の Alarm である Atm128AlarmC のためのタイマとしても用いられている．そこで，そのモジュールがタイマの値を設定する際に，その値が次のサンプリングタイミングまでの時間を超える場合には，次のサンプリングタイミングの値を引いて設定する．

割込み制御機構

割り込み制御機構の動作の開始にはタイマを用いた．サンプリングの開始前に 2.5 ms 前にすべての割込みフラグの状態をメモリに書き出し，サンプリングと割り込み制御機構のためのタイマの発火以外のすべての割込み可能フラグを不許可にする．そして，サンプリング開始の 1 ms 後にすべての割込みフラグの状態を，サンプリング前にメモリに書き出した割込みフラグの状態に戻す．

加えて，プログラムの途中で変更される割込みフラグの制御機構を実装した．Timer1 compare match resisterC，Serial 通信，SPI 通信の割込み可能フラグは初期化時だけでなくプログラムの途中で変更される．そこで，これらの割込み可能フラグを操作する際には，サンプリングの 1ms 前である場合には，サンプリング前にメモリに書き出した割込みフラグの状態を変更することで，サンプリング後に割込みを許可する．

クリティカルセクション制御機構

TinyOS ではクリティカルセクションを atomic{} を用いて記述する．クリティカルセクションは，コンパイル時に，atomic 節の開始部分を割込み許可フラグを不許可にする関数に置換し，atomic の終了部分を割込み許可フラグを開始時の状態に戻す関数に置換することによって実現されている．本実装では，対応する関数に，進入可否を判定し必要

に応じてループを行う機能を追加する．具体的には次のサンプリングの開始の 1 ms 以内であればサンプリングが開始するまで空ループを実行する．

ここで設定する割込み許可フラグは，すべての割込みを制御する割込みフラグであり，atomic{} で括られた箇所ではリセット以外の割込みが発生しない．本実装では，次のサンプリングの数 ms 前からサンプリングまでの間，アトミックセクションへの進入を待機させるため，atomic の開始時に実行される atm128hardware.h 内の nesc_atomic_start 関数にループを挿入する．

3.4　GPS との同期

機器の接続

本章では GPS と同期した同期サンプリングを行うための時刻同期手法を示す．GPS 受信機は 1 秒ごと PPS(precise positioning service) 信号が出力され，その直後に時刻情報，位置情報，衛星情報を出力する．本実装では，時刻同期には既存の時刻付きプロトコルを用い，GPS 受信機からの情報を用いて UTC 時刻との同期を行う．

図 3.10 のように GPS 受信機の PPS 信号をシンクノードに接続し，タイマのハードウェアにより，GPS の PPS 信号をキャプチャすることで，高精度にタイムスタンプを得る．GPS 受信機の時刻情報等は，PC のシリアルポートから受信し UTC 時間を求め，下式を用いて，次 PPS 信号のタイイングのサンプリングのインデックスとなる *SlotID* を求め，シンクノードに送信する．

$$SlotID = UTCTime\ [\mathrm{s}] * Samplingrate\ [\mathrm{Hz}] \bmod 2^{32}$$

ルートノードの GPS との同期

GPS 受信機を用いた UTC 時間とのシンクノードの同期は，ノード間の時刻同期ライブラリを利用する．シンクノードの時刻同期のライブラリに PPS 信号の情報を入力する際に，あたかも GPS 受信機から時刻同期パケットを受信したように見せることで，既存の時刻同期ライブラリを利用できる．具体的には，PPS 信号受信時に以下のような時刻

同期パケットを受信した用に見せる．

$$LocalTime = \quad PPS\text{信号のキャプチャタイミング}$$
$$GlobalTime = \quad SlotID * GlobalTime\text{のサンプリング間隔}$$

なお，UTC 時刻との同期に当たりセンサノードのプログラムは変更はない．

タイマの利用方法

UTC 時刻と同期しつつ，その時刻同期情報を他ノードに転送するためシンクノードは複数のタイムスタンプ取得が必要である．UTC 時刻と同期するため PPS 信号のタイミングのタイムスタンプ取得と，センサノードへの時刻同期パケットの送信のため送信無線チップの SFD 信号のタイミングのタイムスタンプ取得がが必要である

しかしながら，ATmega1281 の各々のタイマは Input Capture を 1 つのみ具備するため，このままでは複数のタイムスタンプを一つの時間軸上で取得することができない．そこで，各々のタイムスタンプの取得を行うタイマは異なるが，それら 2 個のタイマのカウンタを図 3.11 のように同期させて動作させることで，同一の時間軸上のタイムスタンプを得ることができる．複数のタイマのカウンタの同期は，タイマの初期化時にカウンタの値を同期させてから実行することで可能である．

3.5　常時収集

本システムではセンサデータの常時収集を行う．センサデータの常時収集を行うことで，揺れの検出についての偽陽性（False positive），偽陰性（False negative）を抑えることができる．前者は，建造物内での人の動きなどによる振動の発生や，センサのノイズに

図 3.10　GPS 受信機，PC，シンクノードの接続

よって発生する．後者は，地震の揺れが軽微であるときに発生する．そこで本システムでは，100Hz でサンプリングしたセンサデータの常時収集を行う．前述の通り，全てのセンサデータの信頼性のある収集は現実的でないことから，指定したノードから，end-to-end の信頼性のある収集を実行する．また，常時収集の収容台数を増やすために，必要に応じてリサンプリングを行い，収集するセンサデータ量を低減させる．

3.6　おわりに

本章では，実展開に向けた無線型地震観測システムの設計と実装を示した．本システムは機能として， UTC 時刻と同期した高精度な同期サンプリング中に，センサデータの常時収集，センサノードの状態収集，センサノードへの指令拡散を行うことを特徴とする．本章ではこの中でも実装と設計の要点となる，高精度な同期サンプリング，GPS 受信器の出力する UTC 時刻との同期，常時収集について詳細を述べた．

次章では本章で設計実装を行った無線型地震観測システムを評価し，サンプリングジッ

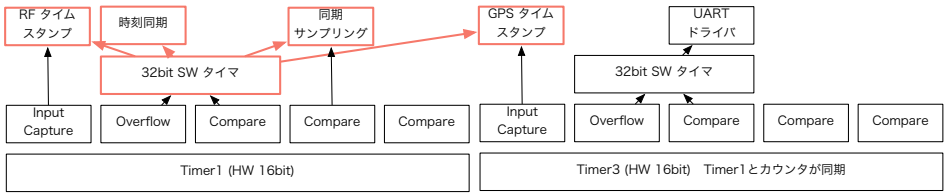

図 3.11　タイマの利用方法

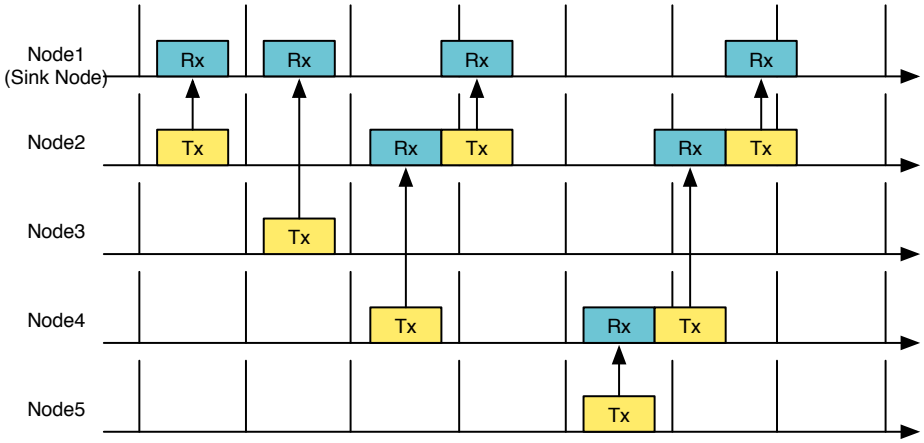

図 3.12　タイマの利用方法

タ，高精度な同期サンプリング実行時の通信のスループット，システムの CPU 使用率を明らかにし，システムが実用的かつ現実的であることを示す．

第 4 章

評価

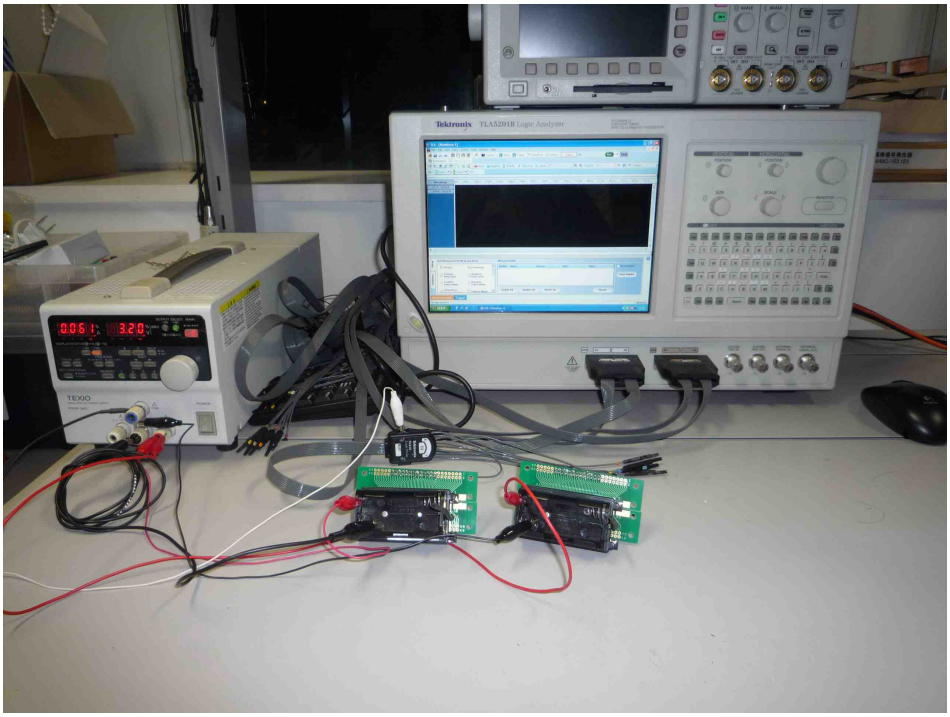

図 4.1　サンプリングジッタの測定

4.1　はじめに

本章では，前章で設計実装した無線型地震観測システムを評価する．はじめに，サンプリングジッタの評価を行い高精度な同期サンプリングが実現されていることを示す．つぎに，高精度な同期サンプリング通信のスループットから，システムの常時収集の収容台数を示す．最後に，システムの CPU 使用率を明らかにし，システムの処理の余裕と他処理実行の可能性を示す．これらの評価から無線型地震観測システムについてシステムが実用的かつ現実的であることを示す．

4.2　サンプリングジッタ

地震観測システムのサンプリングジッタ

実装した UTC 時刻と同期した高精度同期サンプリング機構についてジッタを評価した．各ノードは 100 Hz で同期サンプリングを模擬し，実際のサンプリングの代わりに

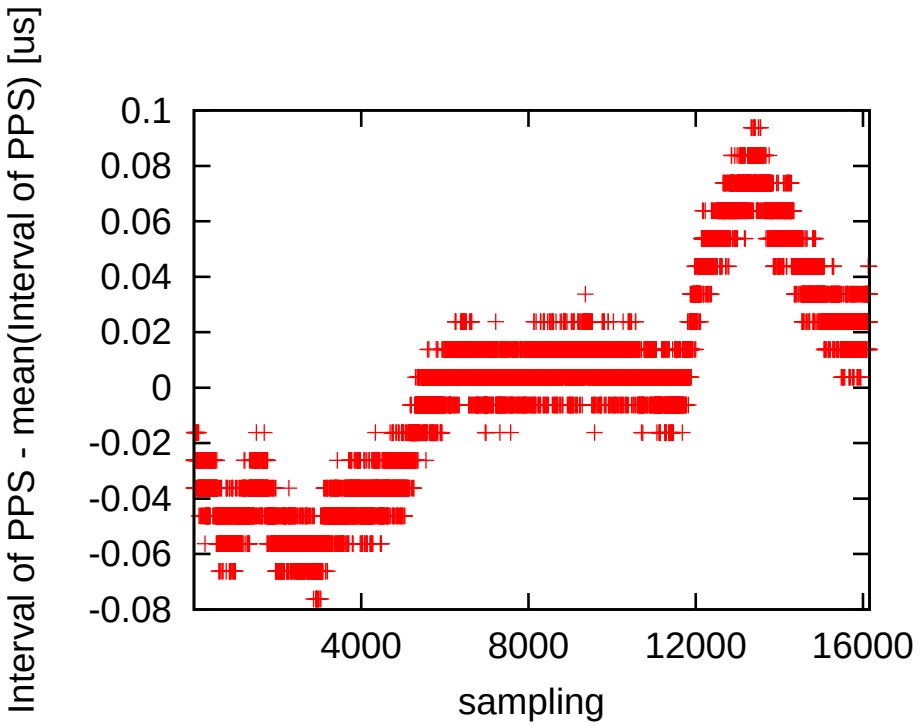

図 4.2　GPS 受信機のクロック変動

CPU の GPIO を立て，図 4.1 のようにロジックアナライザでタイミングを測定した．ロジックアナライザには Tektronix 社 の TLA5201B を用い，サンプリングの分解能は 10 ns に設定した．時刻同期パケットの送信間隔は 3 s ごとである．他処理による遅延がないことを示すため，時刻同期のための通信，状態収集，常時収集，指令の拡散を行った状態で評価を行った．評価においては 10 台のノードを用いて実験を行い，すべてのノードが起動して同期サンプリングを始めた後に 2 時間半程度計測を行った．なお，実験に用いた GP S 受信機 CCA-453JP の PPS 信号の変動を測定した結果を図 4.2 に示す．データシート上では 1 us の精度を保証しており，実際には，0.1 pps 程度の精度あることがわかる．

表 4.1 にシンクノードと 7 台のセンサノードを用いた際のシンクノードとのサンプリングタイミングのジッタを示す．サンプリングジッタは 7 ホップで最大で 14.48 us，標準偏差が 2.97 us とサンプリング周期の 10ms に対して十分に小さい．これは通信などの他のタスクがある場合についても，サンプリングのジッタが時刻同期精度と同程度に抑えられていることがわかる．

図 4.3 に 7 hop したノードのサンプリングタイミングのジッタのヒストグラムを示す．

表 4.1 地震観測システムのサンプリングのジッタ

node	Average	STD	Max
node1(sink node)	1.36us	0.69us	3.36us
node2	0.95us	0.89us	4.07us
node3	0.81us	1.12us	5.62us
node4	0.75us	1.44us	6.28us
node5	0.66us	2.01us	9.16us
node6	0.53us	2.56us	11.80us
node7	0.42us	2.97us	14.48us

2.2 us 程度のサンプリングジッタが発生していることがわかる．これは，主に時刻同期の誤差に起因する誤差である．

図 4.4 に同期サンプリング機構と制御機構を用いた際のサンプリングジッタの時系列を示す．約 300 サンプリングごとに段ができていることがわかる．これは，時刻同期パケットを 3 s ごとに受信し，グローバルタイムとの skew と offset を再計算して更新したために発生する．また，それぞれの段が 1 us 程度の幅があることがわかる．シンクノードとのクロックの周波数の固有差によりジッタがごとサンプリングごと変化し，その変化が 1 クロック分まで蓄積した際に修正されているためである．

制御機構を用いた場合のサンプリングジッタ

次に，割り込み制御とクリティカルセクション制御機構を有する同期サンプリング機構についてジッタを評価した．サンプリングタイミングの測定方法，時刻同期パケットの送信間隔は，地震観測システムのジッタの評価と同じである．本実験では通信によるジッタの増加を評価するため，時刻同期のための通信のみの場合（W/O Communication）と，それに加え同期サンプリングの直後に TinyOS の CTP ライブラリを用いた通信を行っている場合（W/ Communication）の評価を行った．評価においては 5 台のノードを用いて実験を行い，すべてのノードが起動して同期サンプリングを始めた後に 5 分程度計測を行った．なおシンクノードのジッタを測定したところ，最大 7 ns 程度であった．

表 4.2 に 4 台のセンサノードを用いた際のシンクノードとのサンプリングタイミングの

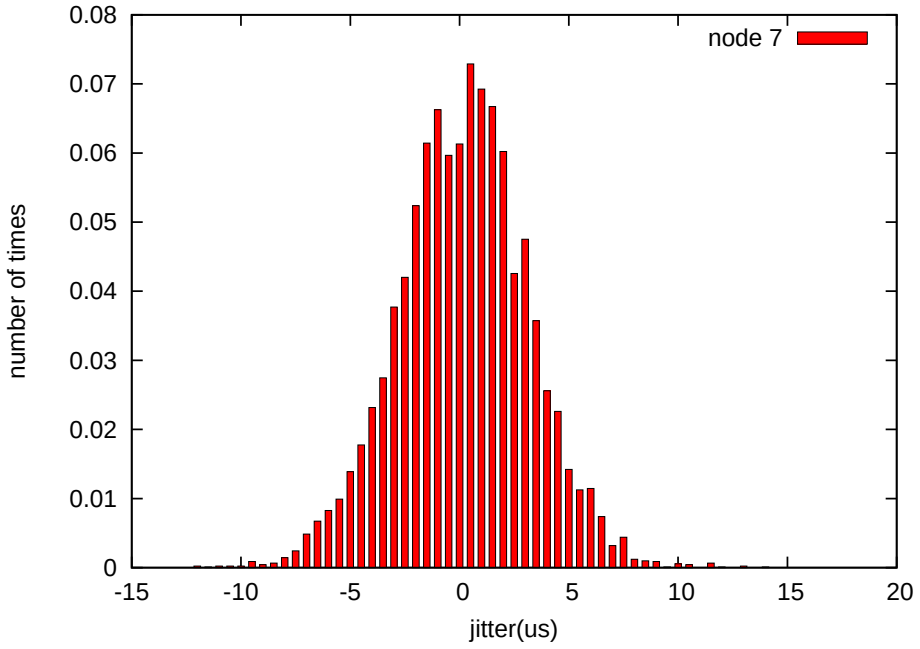

図 4.3　地震観測システムのサンプリングジッタのヒストグラム

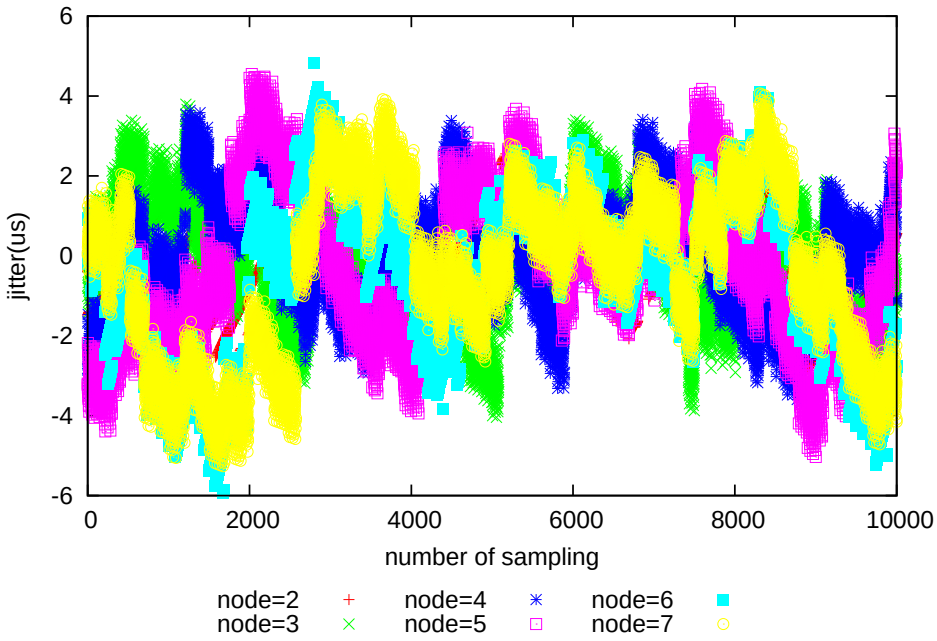

図 4.4　地震観測システムのサンプリングジッタの時系列

表 4.2 制御機構を用いた同期サンプリングのジッタ

Configuration	Average	STD	Max
W/O Guard, W/O Communication	-5.2 us	8.4 us	65.9 us
W/O Guard, W/ Communication	9.1 us	9.3 us	69.1 us
W/ Guard, W/O Communication	-2.8 us	2.0 us	3.0 us
W/ Guard, W/ Communication	-3.1 us	2.2 us	3.8 us

ジッタを示す．割り込み制御がない場合は最大で 69.1 us 程度のジッタが発生することがわかる．割込み制御及びクリティカルセクション制御がない場合，サンプリングジッタが 9.3 us 程度発生する．

通信がある場合のにサンプリングジッタが大きいのは，通信がない場合に比べ，時間あたりのクリティカルセクションや割込みが発生する割合が高いからである．通信の有無にかかわらず，サンプリングのジッタが 2.2 us 以下に抑えられていることがわかる．またこのジッタは割込み制御及びクリティカルセクション制御ありの場合，通信の有無にかかわらず，サンプリングのジッタが同程度に抑えられていることがわかる．

図 4.5 に制御機構があり通信を行っている場合のサンプリングタイミングのジッタのヒストグラムを示す．2.2 us 程度のサンプリングジッタが発生していることがわかる．これは，主に時刻同期の誤差に起因する誤差である．

図 4.6 に同期サンプリング機構と制御機構を用いた際のサンプリングジッタの時系列を示す．約 300 サンプリングごとに段ができていることがわかる．これは，時刻同期パケットを 3 s ごとに受信し，グローバルタイムとの skew と offset を再計算して更新したために発生する．また，それぞれの段が 1 us 程度の幅があることがわかる．シンクノードとのクロックの周波数の固有差によりジッタがごとサンプリングごと変化し，その変化が 1 クロック分まで蓄積した際に修正されているためである．

4.3 転送成功率

次に，高精度な同期サンプリングを行っている場合の通信のスループットと通信の成功率の評価を図 4.7 のように行った．具体的には，(a) データをすべてリアルタイムに収集するアプリケーションに用いることができるかを知るため，サンプリングごとにサンプリ

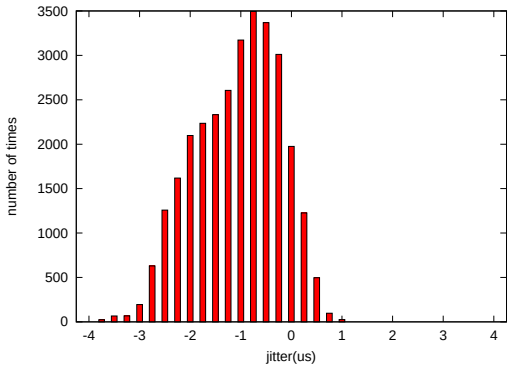

図 4.5　制御機構を用いた場合のサンプリングジッタのヒストグラム

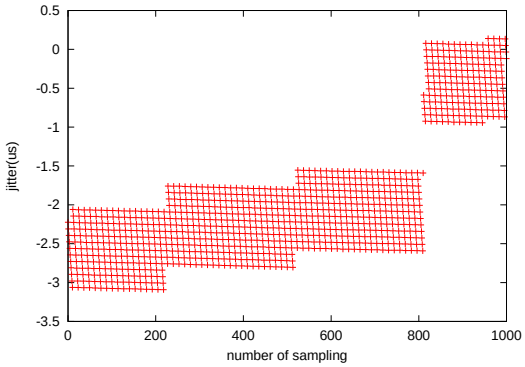

図 4.6　制御機構を用いた場合のサンプリングジッタの時系列

ングデータを送信した際の転送成功率，(b) (a) の送信間隔を 1/8 とした場合の転送成功率，(c) 必要に応じてデータを収集するアプリケーションにおける通信のパフォーマンスを知るため，10,000 パケットを送信完了する時間から計算される通信のスループットの 3 つを評価した．

本評価では，同期サンプリングを行わないプログラム，同期サンプリングのみプログラム，同期サンプリングと制御機構を用いたプログラムの 3 種類で評価を行った．本評価においては TinyOS の CTP [28] ライブラリを用いた通信を行い，成功率やスループットはパケットロスがあった際には再送を行い評価している．なお，パケットサイズは 3 軸の加速度センサで 16bit でサンプリングすることを想定している．1 サンプリングごとに送信する場合はデータが 10 byte，パケット全体で 33 byte であり，8 サンプリングごと送信する場合はデータが 52 byte，パケット全体で 78 byte である．

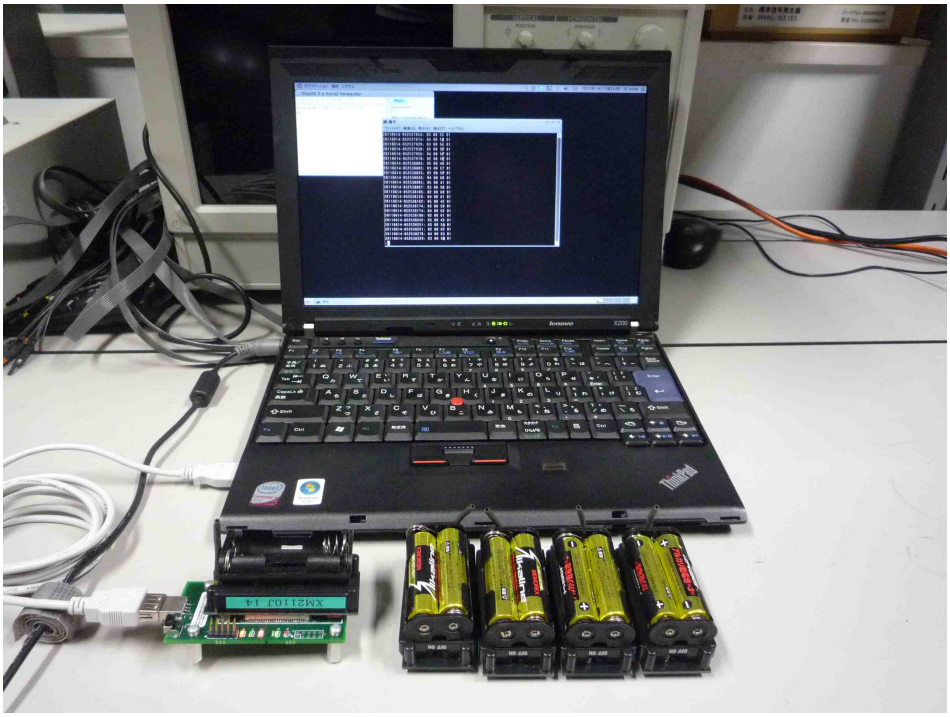

図 4.7　転送成功率の測定

1 サンプリングごとに通信をした場合の通信の成功率を図 4.8 に示す．通信の成功率は同期サンプリング後に各サンプリングデータがシンクノードまで到達した割合から評価した．いずれの条件においてもパケットのロスが発生している．CTP では通信の信頼性を確保するためソフトウェアの ACK を用いることから，これらのパケットのロスは送信のキューのオーバーフローによりデータが破棄されてしまったことに起因する．このことから，[29] のようなプロトコルの適用が考えられる．

同期サンプリング機構の有無による成功率の低下は 1 % 程度であり影響が小さい．これは，時刻同期パケットは 3 s ごとの頻度で送信されるため，時刻同期を行った際の通信量の増加が小さいためである．また，後の評価で示されるとおりノードで発生する同期サンプリングのための計算量は比較的小さいためである．

また，図 4.8 から制御機構の有無による低下が 10 % 程度発生していることがわかる．これは，各種制御機構によりセンサノードの動作が待たされてしまうため送信が遅れ，サンプリングしたデータが破棄されることが多くなるためである．

8 サンプリングごとに通信をした場合の通信の成功率を図 4.9 に示す．いずれの条件に

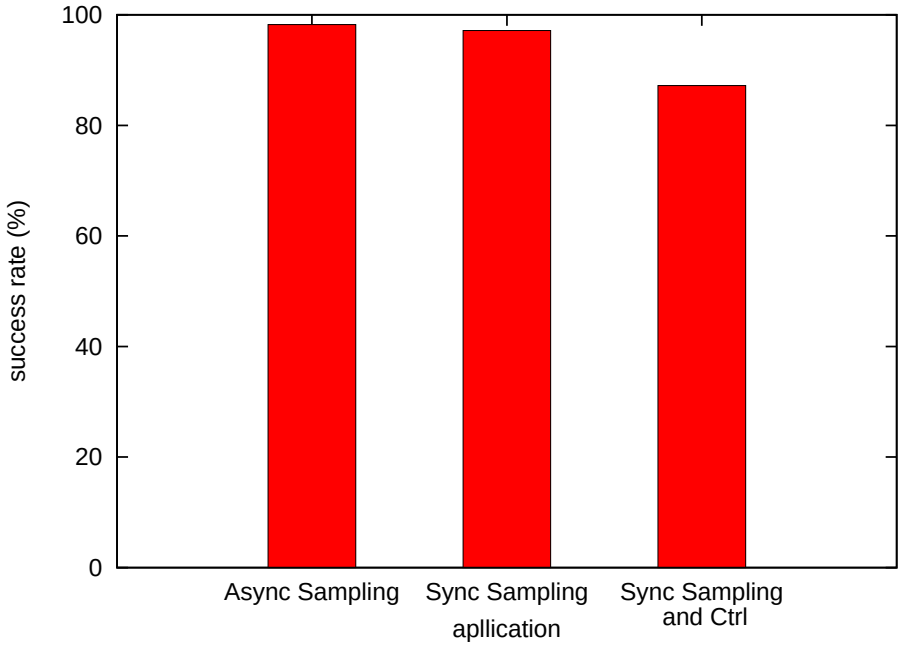

図 4.8　通信の成功率

おいてもパケットのロスは多少発生しているが，同期サンプリング機構と制御機構を用いた場合でも通信成功率が同期サンプリング機構を使わない場合の 99.8% 程度である．このことから，数回に 1 回の通信頻度であれば，制御機構のある同期サンプリング機構は，同期サンプリング機構を用いずにサンプリングを行うプログラムと同様に通信が行えることがわかる．

10000 パケットを送信した際の送信時間から算出した通信のスループットを図 4.10 に示す．本評価では空き時間には常に通信を行わせるため，データの通信が終わったらすぐに新しいデータを送信のキューに入れた．通信の成功率の評価同様，同期サンプリング機構の有無による成功率の低下は 5 % 程度であり影響が小さいことがわかる．また，制御機構の有無による低下が 18 % 程度であり影響が大きいことがわかる．

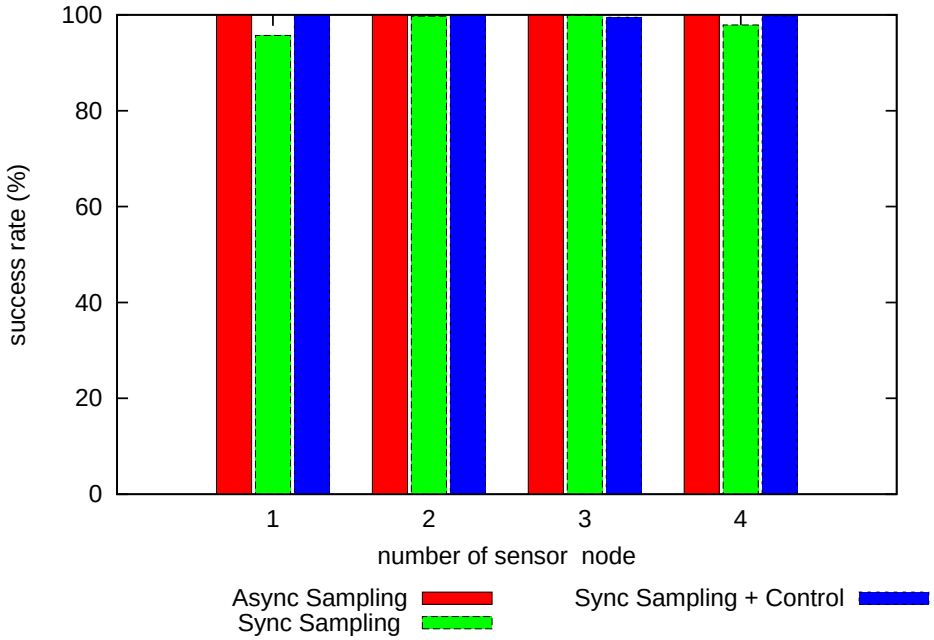

図 4.9　8 パケットごとに送信した際の通信の成功率

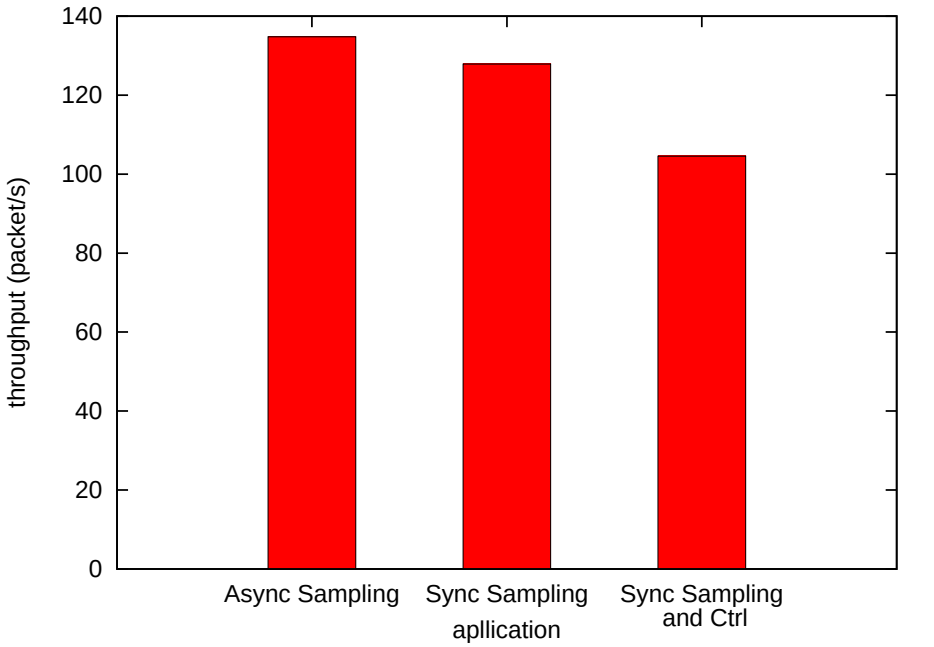

図 4.10　通信のスループット

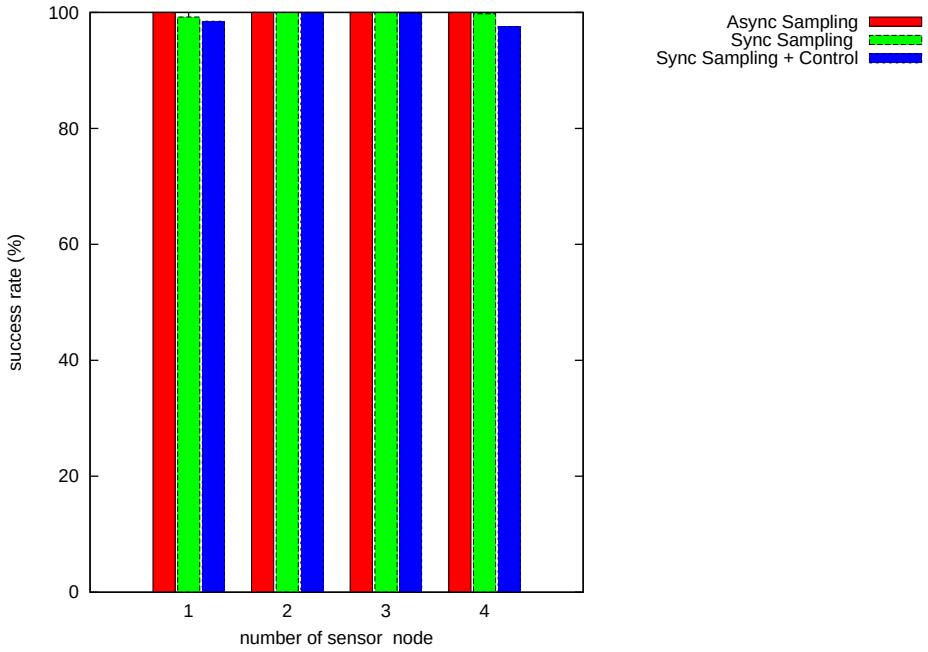

図 4.11　8 パケットごとに通信タイミングが重ならないように送信した際の通信の成功率

4.4　CPU 使用率

地震モニタリングシステムの CPU 使用率

時刻同期の計算とサンプリングタイミングの計算の CPU 使用率と，同期サンプリング機構と制御機構を用いた際のプログラムの CPU 使用率を評価した．それぞれ，処理の開始と終了時に CPU の GPIO に出力し，ロジックアナライザを用いてその間隔を測定した．タスクの処理時間を計測には，図 4.12 に示されるようにタスクスケジューラによるタスクの実行開始の前後に GPIO に出力した．また，タスクスケジューラ含めた処理の実行時間をの計測には，図 4.13 に示されるようにスリープの前後で GPIO に出力した．

表 4.3 に地震観測システムの CPU 使用率を示す．全体の CPU 使用率の合計が 14.52% と比較的小さいことから，パケットロスは無線通信は CPU がボトルネックではないことがわかる．また，サンプリング間隔の計算は 1 サンプリングあたり 0.398ms 程度であることから，100 Hz 以上のサンプリングレートでサンプリング可能であることがわかる．なお，本評価においてサンプリング時間の計算は次節のサンプリング時刻の計算より大き

```
command void Scheduler.taskLoop() {
  for (;;) {
    uint8_t nextTask;
    atomic {
      while ((nextTask = popTask()) == NO_TASK) {
        call McuSleep.sleep();
      }
    }
    PORTA |= 0x01; // start task
    signal TaskBasic.runTask[nextTask]();
    PORTA &= ~0x01; // end task
  }
}
```

図 4.12 タスク実行時間の取得

```
async command void McuSleep.sleep() {
  uint8_t powerState;
  powerState = mcombine
(getPowerState(), call McuPowerOverride.lowestState());
  SMCR = (SMCR & 0xf0) | 1 << SE |
         read_uint8_t(&atm128PowerBits[powerState]);
  sei();
  // All of memory may change at this point...
  asm volatile ("sleep" : : : "memory");
  cli();
  CLR_BIT(SMCR, SE);
}
```

図 4.13 スリープしていない時間の取得

くなっている．これは，本システムでははは次サンプリング時刻の計算に，サンプリング後のタスクを実行するためのタスクの追加を行っているためである．

制御機構を用いた場合の CPU 使用率

時刻同期の計算を行う処理と，次サンプリングのタイミングを計算する処理の CPU 使用率を計測した結果を表 4.4 に示す．それぞれの処理の CPU 使用率は少ないことがわか

表 4.3 地震観測システムの CPU 使用率

処理		CPU usage
同期サンプリング	サンプリング時間の計算	3.98%
	時刻同期の計算	0.25%
地震ボードの処理	シリアル通信	0.72%
	受信パケットのコピー	0.52%
データ収集の動作	フォワーディング	3.44%
	ルーティング	0.04%
データ散布の動作	受信データの散布	0.00%
無線通信	SPI 通信	2.5%
	バックオフ時間の計算	0.76%
その他		2.98%
合計		14.52%

る．このことから，高精度な同期サンプリングを行っている場合であっても，信号処理などの高負荷な処理を行うことができると言える．しかし，時刻同期の計算時間が長いことがわかる．これは，TinyOS の FTSP ライブラリは線形回帰を利用して時刻同期を行っており，ここで発生する浮動小数の除算が CPU に対して負荷が大きいためである．そのため，時間制約のある他のタスクを行う場合，時刻同期の計算を行うスロットと同じスロットで発生しないようにスケジューリングを行う必要がある．

サンプリングデータの通信を行っていない場合のプログラム全体の CPU 使用率を計測した結果を図 4.14 に示す．同期サンプリングを行わないプログラム，同期サンプリングのみのプログラム，同期サンプリングと制御機構を用いたプログラムにおいて計測を行った．割込み制御機構が働いている区間でタスクが実行されていない間は，割込み制御機構が CPU を使用している状態として測定した．同様に，クリティカルセクション制御機構がクリティカルセクションの実行を待機させている場合も CPU を利用できないことから CPU を使用状態とした．

同期サンプリング機構を用いた場合，用いない場合に比べて割込みの CPU 使用率が増えていることがわかる．サンプリングの割込みの処理の中に次サンプリングタイミングの

表 4.4 CPU 使用率

task	average	STD	CPU usage
time synchronization	5.925 ms	508. us	0.20 %
timing calculation	113 us	0.4 us	1.13 %

計算をおこなっているためである．逆に，同期サンプリング機構を用いた場合であってもタスクによる CPU 使用率が増えていないことがわかる．時刻同期の計算はタスクとして行っているがそれによる CPU 時間の消費は小さいことがわかる．

制御機構を用いた場合には割込み制御による CPU 時間の消費が 34.5 % と大きい．また，割込みの CPU 使用率が，制御機構の有無で変化が少ないことから，割込み制御機構は割込みによる動作であるが，CPU 時間をあまり消費しないことがわかる．

サンプリングデータの通信を 1 サンプリングごとに行っている場合のプログラム全体の CPU 使用率を計測した結果を図 4.15 に示す．制御機構がある場合にタスクの CPU 使用時間が減少していることがわかる．本評価ではタスク内で通信を開始しているため，タスクによる CPU 時間がデータを送信をしない場合に比べて大きく増加している．制御機構がある場合には通信のスループットが低いため，送信するタスクを実行しても送信が busy のため，送信せずにタスクを終えてしまうためである．

また，本実装では 10 ms のサンプリング間隔のうち 3.5 ms の時間は割込みが制御されているが，割込み制御による CPU の使用率が 25.5 % 程度であることがわかる．これは割込み制御中であってもタスクを実行することができるためである．このことから，割込みから発生する処理を行えない時間が大きいが，タスクを実行することができる時間は制御機構によりあまり減少しないことがわかる．

4.5 おわりに

本章では，前章で設計実装した無線型地震観測システムの評価を行った．．はじめに，サンプリングジッタの評価を行い高精度な同期サンプリングが実現されていることを示した．つぎに，高精度な同期サンプリング通信のスループットから，システムの常時収集の収容台数を示した．最後に，システムの CPU 使用率を明らかにし，システムの処理の余

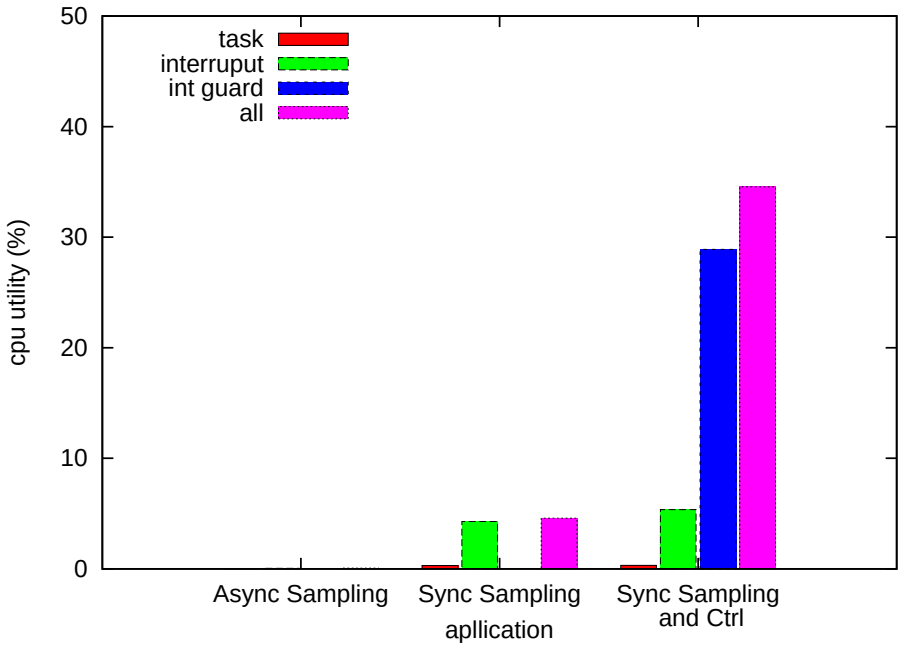

図 4.14　通信をしていない場合の CPU の使用率

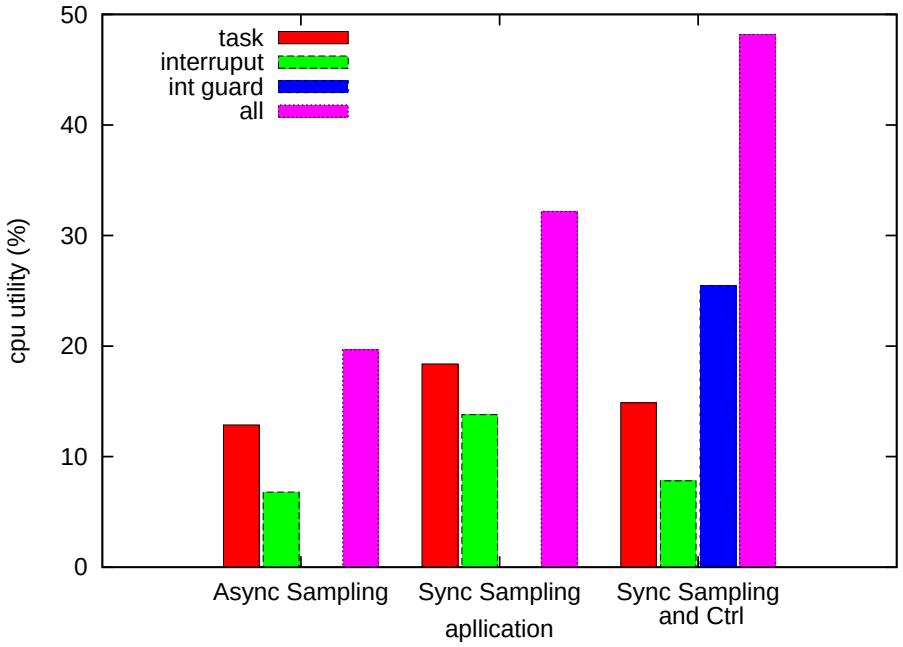

図 4.15　1 サンプリングごとに送信している際の CPU 使用率

裕と他処理実行の可能性を示した．これらの評価から無線型地震観測システムについてシステムが実用的かつ現実的であることを示された．

第 5 章

結論

5.1 本論文の主たる成果

本論文では，実展開に必要な機能を有する無線型地震観測システムの設計および実装を示し，評価によりシステムが実用的かつ現実的であることを示した．

無線型地震観測システムの実展開に必要な機能として， UTC 時刻と同期した高精度な同期サンプリング中に，センサデータの常時収集，センサノードの状態収集，センサノードへの指令拡散を行うことを示した．そして，実装と設計としてシステムの要点である高精度な同期サンプリング，GPS 受信器の出力する UTC 時刻との同期，常時収集について詳細を述べた．

システムが実用的かつ現実的であることを示すため評価を行った．無線センサノード高密度・広範囲の実空間の情報を収集するため，コスト面から省資源性が求められていることから， CPU の処理能力の制約，無線通信の帯域の制約により困難となる．そこで，提案した設計手法にて高精度な同期タスク実行が実行できることを示した．次に，高精度な同期タスク実行中に複数のノードからのセンサデータの常時収集が可能であることを示した．最後に他の計算が可能であることを示した．これらの評価によりシステムが実用的かつ現実的であることが明らかになった．

5.2 今後の課題

本研究では無線型地震観測システムの実展開に向けて，必要な機能の明確化，システムの設計と実装，実用性の評価を行った．筆者は無線型地震観測システムの実展開に向けて，時刻同期の同期誤差の解析，高速な一括収集についても研究を進めている．

本論文の評価によりシステムの時刻同期誤差以外のサンプリングジッタの原因は，タイマやクロックのの分解能や A/D コンバータの遅延であることがわかる．そこで，時刻同期誤差が解析的に明らかになれば，システムのサンプリングジッタをセンサノードやセンサボードの設計から明らかにすることができる．実際に，カルマンフィルタを用いた時刻同期プロトコルについて誤差を解析的に求める等の検討を行っている．特に，センサノードの具備するクロックの周波数変動を求めることが困難であることから，真のモデルと計算に用いるモデルが異なる場合の誤差を解析的に求め検討を行っている．

また，高速なセンサデータの一括収集についてはマルチチャネルバルク転送プロトコ

ルについての研究を進めている．無線型地震観測システムでは，振動を計測するために 100 Hz という高いレートでサンプリングを行うため，膨大なデータが発生する．一括収集型アプリケーションでは，データ収集時間をいかに短縮するかが重要である．収集時間短縮のためには，キャリアセンスなどの送信ごとに生じる遅延，衝突によるパケットロス，シンクノードの受信待機時間のオーバヘッドの削減が課題となる．これらの解決に向けて，筆者らはマルチチャネル通信，ブロック転送，並列化を利用するマルチチャネルバルク転送機構の開発を進めている．実際に Busy Ack を用いた MAC プロトコルなどの有用性を示すなどの研究を進めている．

謝辞

はじめに，本研究を進めて行くにあたりご指導を頂いた森川博之教授に深く感謝いたします．研究に関することのみならず，自らの限界を知ることなどといった人間性についてもご指導をいただきました．研究環境の整備にご尽力下さった翁長久教授，今泉英明准教授に感謝いたします．翁長久准さんには自身の仕事上での経験談を話していただき，仕事上でのコミュニケーションの重要さを教えていただきました．今泉英明さんには光通信やネットワークの分野の研究や技術について教えていただきました．研究を進める上では，学部生の間は猿渡俊介助教，修士課程では鈴木誠助教に多大なるご指導いただきました．猿渡俊介助教からは研究内容についてだけでなく，論理的に考える方法，研究者の心構えをご指導を頂きました．ありがとうございます．また，研究が進まず自信を失ったときにかけてくださった「やり方さえ知っていれば，何だってできる」という励ましの言葉は忘れられません．鈴木誠助教からは修士学生として自立的に研究をするように指導してくださったこと，失敗を恐れず挑戦するようご指導をいただきました．ありがとうございます．柏の葉キャンパス駅で「成功すればまわりから賞賛してもらえる，失敗しても誰も自分を責めるようなことはしない．自分で落ち込むだけだ」と話してくださったことは忘れられません．研究環境の整備に尽力して頂きました秘書の川北敦子さん，石崎智子さんには深く感謝いたします．学生が多くの時間を研究に割くことができるように配慮していただいたこと，研究室生活について相談してくださったこと本当にありがとうございます．

社会人博士の黒岩拓人さんには多大なるご指導を頂きました．研究のアドバイスだけでなく，自立的な研究を行うように議論に多くの時間を費やしてくださったことに感謝いたします．本研究室の OB の荒木靖宏さん，川西直さん，高木潤一郎さん，博士課程の石田繁巳さんには，研究の議論をしていただいただけでなく，研究室のネットワークの運用方法やサーバの管理方法について多くのことを教えていただきました．ありがとうございます．博士課程の金昊俊さん，劉進志さん，Brogi Guillaume さん，彭さん，研究員の森戸

貴さんが真摯に研究に向き合う姿勢を示してくださったことは，研究活動のモチベーションの維持につながりました．ありがとうございます社会人博士の諸橋知雄さん，山本享弘さん，浅井光太郎さん，加美伸治さんの研究や会社の話から多くの刺激を頂きました．ありがとうございます．

岩元啓君，中嶋毅彰君，菅沼久浩君，高超君，長い間共に過ごした研究生活のなかで大変お世話になりました．岩元啓君の研究に集中し，研究室イベントにも積極的な姿は，研究室に在籍する学生の鏡です．つらいときであっても明るく振る舞う中嶋毅彰君は，研究室を全体を明るくしてくれました．他分野の論文を多く読んだり，最新技術を積極的に調べる菅沼久浩君からは多くの情報だけでなく，研究や技術への好奇心をいただきました．高超君の現状に満足しないという向上心と，楽しむときは徹底的に楽しむから姿からとても刺激を受けました．修士 1 年の奥井寛樹君，下城拓也君，田代論拡君は修士過程から森川研究室に配属されたにも関わらずすぐに研究室のシステムを把握して仕事をひきついでくれました．ありがとう．研究室配属後の早い段階から研究に取りかかりった学部 4 年の山下靖貴君，吉田早人君，米川慧君の姿から，ゴールを見据えて行動せねばと気づかされました．ありがとう．研究生の Theerat Sakdejayont 君の学習意欲にはとても驚かされるとともに，意欲的にものを進めることに気づかされました．ありがとう．

このほかにも，研究室のメンバーや OB・OG の方々，研究室外の方々にお世話になりました．この 3 年間ありがとうございました．

2012 年 2 月 8 日

参考文献

[1] M. Suzuki, N. Kurata, S. Saruwatari, and H. Morikawa, “Demo abstract: A high-density earthquake monitoring system using wireless sensor networks,” Proceedings of the 4th International Conference on Embedded Networked Sensor Systems, pp.373–374, Nov. 2007.

[2] S. Kim, S. Pakzad, D. Culler, J. Demmel, G. Fenves, S. Glaser, and M. Turon, “Health monitoring of civil infrastructures using wireless sensor networks,” Proceedings of the 6th international conference on Information processing in sensor networks, pp.254–263, 2007.

[3] G. Werner-Allen, K. Lorincz, J. Johnson, J. Lees, and M. Welsh, “Fidelity and yield in a volcano monitoring sensor network,” Proceedings of the 7th symposium on Operating systems design and implementation, pp.381–396, 2006.

[4] N. Xu, S. Rangwala, K.K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, and D. Estrin, “A wireless sensor network for structural monitoring,” Proceedings of the 2nd international conference on Embedded networked sensor systems, pp.13–24, 2004.

[5] M. Maróti, B. Kusy, G. Simon, and A. Lédeczi, “The flooding time synchronization protocol,” Proceedings of the 2nd international conference on Embedded networked sensor systems, pp.39–49, SenSys ’04, 2004.

[6] G. Simon, M. Maróti, A. Lédeczi, G. Balogh, B. Kusy, A. Nádas, G. Pap, J. Sallai, and K. Frampton, “Sensor network-based countersniper system,” Proceedings of the 2nd international conference on Embedded networked sensor systems, pp.1–12, 2004.

[7] M. Aoun, J. Catalano, and P. Stok, “Distributed task synchronization in wire-

less sensor networks," Proceedings of the 6th European Conference on Wireless Sensor Networks, pp.150–165, 2009.

[8] 鈴木　誠，倉田成人，猿渡俊介，森川博之，"無線センサネットワークによる地震モニタリングシステムの実装と評価," 信学技報, USN2007-66，pp.65–70，2008.

[9] S. Saruwatari, M. Suzuki, and H. Morikawa, "A compact hard real-time operating system for wireless sensor nodes," Proceedings of the 6th international conference on Networked sensing systems, pp.167–174, 2009.

[10] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, "System architecture directions for networked sensors," SIGPLAN Not., vol.35, pp.93–104, Nov. 2000.

[11] C. Lenzen, T. Locher, and R. Wattenhofer, "Tight bounds for clock synchronization," Proceedings of the 28th ACM symposium on Principles of distributed computing, pp.46–55, 2009.

[12] R. Fan and N. Lynch, "Gradient clock synchronization," Proceedings of the twenty-third annual ACM symposium on Principles of distributed computing, pp.320–327, 2004.

[13] C.-J.M. Liang, J. Liu, L. Luo, A. Terzis, and F. Zhao, "Racnet: a high-fidelity data center sensing network," Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems, pp.15–28, 2009.

[14] J. Sallai, B. Kusy, A. Ledeczi, and P. Dutta, "On the scalability of routing integrated time synchronization," 3rd European Workshop on Wireless Sensor Networks (EWSN 2006), pp.115–131, Feb. 2006.

[15] Y. Zeng, B. Hu, and S. Liu, "Vector kalman filter using multiple parents for time synchronization in multi-hop sensor networks," Proceedings of the Fifth Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks, pp.413–421, 2008.

[16] C. Lenzen, P. Sommer, and R. Wattenhofer, "Optimal clock synchronization in networks," Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems, pp.225–238, 2009.

[17] T. Schmid, P. Dutta, and M.B. Srivastava, "High-resolution, low-power time syn-

chronization an oxymoron no more," Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks, pp.151–161, 2010.

[18] T. Schmid, Z. Charbiwala, Z. Anagnostopoulou, M.B. Srivastava, and P. Dutta, "A case against routing-integrated time synchronization," Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems, pp.267–280, 2010.

[19] F. Ferrari, M. Zimmerling, L. Thiele, and O. Saukh, "Efficient network flooding and time synchronization with glossy," Proceedings of the 10th International Conference on Information Processing in Sensor Networks (IPSN), pp.73–84, 2011.

[20] J. Elson, L. Girod, and D. Estrin, "Fine-grained network time synchronization using reference broadcasts," SIGOPS Oper. Syst. Rev., vol.36, pp.147–163, Dec. 2002.

[21] S. Ganeriwal, R. Kumar, and M.B. Srivastava, "Timing-sync protocol for sensor networks," Proceedings of the 1st international conference on Embedded networked sensor systems, pp.138–149, 2003.

[22] T. Schmid, Z. Charbiwala, R. Shea, and M.B. Srivastava, "Temperature compensated time synchronization," Embedded Systems Letters, IEEE, vol.1, no.2, pp.37–41, aug. 2009.

[23] Z. Zhong, P. Chen, and T. He, "On-demand time synchronization with predictable accuracy," INFOCOM, 2011 Proceedings IEEE, pp.2480–2488, april 2011.

[24] Y. Zeng, B. Hu, and S. Liu, "Vector kalman filter using multiple parents for time synchronization in multi-hop sensor networks," Sensor, Mesh and Ad Hoc Communications and Networks, 2008. SECON '08. 5th Annual IEEE Communications Society Conference on, pp.413–421, june 2008.

[25] B.R. Hamilton, X. Ma, Q. Zhao, and J. Xu, "Aces: adaptive clock estimation and synchronization using kalman filtering," Proceedings of the 14th ACM international conference on Mobile computing and networking, pp.152–162,2008.

[26] "http://tinyos.cvs.sourceforge.net/tinyos/tinyos-1.x/tos/system/timerc.nc."

[27] "http://tinyos.cvs.sourceforge.net/tinyos/tinyos-1.x/apps/highfrequencysampling/microtimerm.nc."

[28] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis, "Collection tree protocol," Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems, pp.1–14, 2009.

[29] S. Moeller, A. Sridharan, B. Krishnamachari, and O. Gnawali, "Routing without routes: The backpressure collection protocol" Proceedings of the 9th International Conference on Information Processing in Sensor Networks, pp.279–290, 2010.

[30] B. Raman, K. Chebrolu, S. Bijwe, V. Gabale, "Pip: A connection-oriented, multi-hop, multi-channel tdma-based mac for high throughput bulk transfer" Proceedings of the 8th International Conference on Embedded Networked Sensor Systems, pp.15–28, 2010.

発表文献

[1] 角田仁, 鈴木誠, 森川博之, “無線センサネットワークにおける分散同期サンプリング機構の評価,” 信学技報, USN2011-17, Jul. 2011.

[2] Hitoshi Kakuta, Makoto Suzuki, Shunsuke. Saruwatari, and H. Morikawa, “Obtaining speaking time in a meeting room by/with source localization,” The 3rd Asia-Europe Workshop on Ubiquitous Computing 2010 (AEWUC’10), Helsinki, Finland,, May. 2010.

[3] 角田仁, 中嶋毅彰, 石田繁巳, 猿渡俊介, 森川博之, “社会実装に向けたヘルスケア情報共有基盤,” 第 5 回人間情報学会, Dec. 2010. Poster

[4] 黒岩拓人, 角田仁, 鈴木誠, 長山智則, 森川博之, “無線センサネットワーク向け高速バルク転送機構における衝突回避に関する検討,” 信学技報, Oct. 2001.

[5] 木下 聡, 喜田弘司, 鈴木 誠, 角田仁, “データストリーム処理による混雑度分析の実証実験と評価,” 信学技報, B-20-48, Mar. 2011.

[6] Fernando Panjaitan, Eric M. Okawa, Hitoshi Kakuta, Johan Hjelm, Toshikane Oda, Shunsuke Saruwatari, and Hiroyuki Morikawa, “Appliance Recognition Using Power State Segregation at Low Sampling Rate,” 信学技報, B-20-58, Mar. 2011.