Vol.2011-SLDM-151 No.1
Vol.2011-EMB-22 No.1
2011/7/3

Windows ヒューマノイド・ロボット・コントローラ ALICE の通信速度評価

斎藤卓也[†]　　村岡洋一[††]

非リアルタイム OS である Windows XP によりリアルタイムなヒューマノイド・ロボット・コントローラを実現するロボット・コントローラ ALICE を開発した。ALICE は加速度センサによる姿勢制御などシビアなリアルタイム性が重視される処理を ALICE 上の組込 CPU に任せ、より高度な動作指示を Windows が担当する事により、Windows XP でヒューマノイド・ロボットのリアルタイム・コントロールを実現した。本論文では、Windows PC と ALICE との通信時間を、通信パケットサイズを変化させることにより、通信効率について調べた。そして Windows PC と ALICE 間の通信時間のジッタについて調査を行った。その結果、ALICE を Windows PC とアクチュエータとの間に入れることにより、Windows でリアルタイムなヒューマノイド・ロボットのコントロールが可能であることが分かった。

Evaluation of Windows Humanoid Robot Controller ALICE's Communication Rate

Takuya Saito[†] and Yoichi Muraoka[††]

We developed ALICE robot controller which enabled to control humanoid robots in realtime using Windows which is not realtime OS. ALICE does processing that real time is severely demanded like the posture control with the acceleration sensor etc. by embedded CPU of ALICE. And, the control of the humanoid robot was achieved with Windows XP by the thing that the Windows software does a more advanced movement directive etc. In this paper, we examined the communication time of Windows PC and ALICE by changing the size of the communication packet about the communication efficiency. And, we investigated the jitter at the communication time between Windows PC and ALICE. As a result, it has been understood we are to be able to control a real-time humanoid robot with Windows by putting ALICE between Windows PC and the actuator.

1. はじめに

ヒューマノイド・ロボットには KONDO KHR シリーズをはじめとする 10 万円前後で購入できる安価なタイプから、HOAP シリーズのように、高級乗用車ほどの価格がするもの、そして産業技術総合研究所の HRP シリーズなど、様々なタイプのものがあるが、これら多くのヒューマノイド・ロボットに共通していることは、ロボットのアクチュエータ制御をメイン・コンピュータにより直接制御しようとしていることにある。KONDO KHR シリーズでは非力な組込 CPU で行っているため、予め決められたモーション・パターンを選び、動作させる程度のことしか行えず、大学等での研究用途のロボットとして使うには不十分である。本格的な研究用ロボットとしては HOAP シリーズがあるが、ロボットのアクチュエータ制御には高度なリアルタイム性が要求されるため、OS に RT Linux を用いている。また、産業技術総合研究所の HRP シリーズも、リアルタイム性をさらに向上させた ART Linux により制御されている。

筆者らは低価格かつ高性能なロボット用プラットフォームとして WR-X[1]2を開発し、そのための専用 OS である WR-X OS も開発した。しかし、WR-X OS や RT Linux のような一般的にマイナーな OS 上のプログラムでヒューマノイド・ロボットのコントロールを行うことになると、それら OS 上のソフトウェアを作らねばならず、多くの人々にとってロボット・プログラミングの敷板高くなる欠点がある。

Windows は多くの人々が慣れ親しんでおり、また圧倒的なソフトウェア資産を有している。また、Visual Studio をはじめとするメジャーなプログラミング環境も用意されており、また C, C++, C#, Visual Basic, Java, Python 等、多くのプログラミング言語が使用可能であり、それらの解説本も豊富である。このため、Windows 上で動作するソフトウェアでヒューマノイド・ロボットを制御することができれば、ロボット・プログラミングの敷居が大きく下げられる。また、VOCALOID を用いることにより、ロボットに簡単に歌を歌わせる機能を追加できるなど、Windows 用の多くのソフトウェア資産を使い、効率よく高度な動作を行うヒューマノイド・ロボットの構築が可能になると考えられる。

しかし、そのように理想的と思える Windows が、ヒューマノイド・ロボットの制御用 OS として用いられない一番の理由は、Windows がリアルタイム OS ではないことである。ヒューマノイド・ロボットのアクチュエータ制御には、高度なリアルタイム性が要求される。滑らかに歩行動作を行うためには、一定時間間隔で確実にアクチュエータに動作指示を出し続けなければならない。

[†] 早稲田大学大学院 基幹理工学研究科
Graduate School of Fundamental Science and Engineering, Waseda University
[††] 早稲田大学理工学術院
Faculty of Science and Engineering, Waseda University

Vol.2011-SLDM-151 No.1
Vol.2011-EMB-22 No.1
2011/7/3

しかし、実際の人間の脳の仕組みを調べてみると、小型ヒューマノイド・ロボットの制御に用いられている約 20ms ごとの筋肉制御を、人間の大脳が直接行なっているわけではない。医学的研究から、哺乳類では筋肉の基本動作パターンは脳幹により行われており、さらに非常にリアルタイム性が重要な制御は脊髄により行われていることが知られている。そして大脳はより高度な指令を行っている。この仕組を模倣することにより、Windows には大脳に相当する、より高度な処理を行わせるようにし、脳幹や脊髄に相当する歩行動作や姿勢制御等のリアルタイム性が重要な処理は、ALICE[3]という高性能組込 CPU を用いたロボット・コントローラを Windows PC とアクチュエータ制御の間に入れることにより、非リアルタイム OS である Windows を用いながら、ヒューマノイド・ロボットのリアルタイム制御を実現することを考えた。

本論文では、Windows PC と ALICE 間の通信について、どの程度の通信速度が出るのか、また通信速度変化のジッタを調べることにより、Windows のリアルタイム性はどれくらいあるのかを実験により調べることにより、Windows + ALICE システムにより、ヒューマノイド・ロボットのコントロールが可能であるかどうかについて、Windows PC と ALICE 間の通信の観点から調査を行った。

2. ALICE を用いたロボットの構成

ALICE は Windows PC とアクチュエータとの間に介在し、Windows OS 上のソフトウェアで、ヒューマノイド・ロボットのリアルタイム制御を可能にするデバイスである。人間の仕組を大まかに分類すると、大脳部分が人間全体の活動を司る、最も高度な処理をしている。大脳で物を考えて、様々な行動を起こしたり、周囲からの刺激に対して何らかのインタラクティブな行動を起こしたりなど、人間活動全体の最も高度な制御を行っている。哺乳類には脳幹部分に歩行中枢（歩行誘発野）と呼ばれる部分が存在することが Shik ら [4]により発見された。さらに森らによる脳幹の上丘の前端で断ち切られた除脳ネコによる実験[5][6][7][8]により、脳幹に電気刺激を加えることにより、４つ足で立ったり、歩いたり、起きたりすることができることが分かっている。このことより、脳幹部分に基本的な動作パターンにおけるモーション・パターンが存在しており、それを大脳が状況に応じてどの動作パターンを指示していると考えられている。また、三半規管からの姿勢制御等は脳幹により行われ、さらに熱いものに触れると勝手に手が動く動作は脊髄反射により行われている。

ALICE を用いたヒューマノイド・ロボット・システムでは、大脳及び動作パターン生成までを Windows PC により行う。そして、動作パターン再生及びアクチュエータの角度情報や、加速度センサ情報を ALICE 内蔵の SuperH/SH2 CPU により高速に処理することにより、人間の脳幹や脊髄が行っているリアルタイム性が要求される姿勢制御等の処理を ALICE が担当する。

このシステムがうまく機能するためには、Windows と ALICE がどれだけの速度で通信できるかということと、その通信速度のバラつきはどれくらいの範囲に収まっているのかを調べる必要がある。もしも通信速度が十分であったとしても、Windows に余りにもリアルタイム性が欠けていて、通信速度のバラつきが大きすぎては、ヒューマノイド・ロボットを持続的にスムーズにコントロールできないからである。

本論文では ALICE を用いた実験用ヒューマノイド・ロボットを用いて、実際に Windows XP 上で動作する C#により書かれたプログラムにより、Windows PC と ALICE との平均通信時間及びその標準偏差を Modulation Domain Analyzer により測定し、平均通信速度とそのバラつきがどの程度に収まっているのかを測定した。

3. ALICE の概要

図 1 に ALICE の外観を示す。また、ALICE の主な仕様を表 1 に、またブロック・ダイアグラムを図 2 に示す。ALICE は大きさが Pico-ITX と同一の 100 mm×72 mm で設計されており、VIA EPIA PX 5000EG 及び PX 10000G とのスタック接続を考慮し、Pico-ITX マザーボードのメモリ等凹凸に合わせて、極力狭いスペースでスタック接続できるように部品を配置している。また Pico-ITX マザーボードとは USB により接続され、電源は USB 経由で供給される。ALICE は 32 ch の双方向 PWM 通信または 115,200 bps 半２重 UART 通信機能を備えている。PWM 通信では通常の RC サーボがコントロール可能であり、さらに KONDO の Red Version サーボにおける双方向 PWM 通信やコマンド送信機能にも対応している。さらに、UART 通信は 115,200 bps で信号線プルアップによる半２重通信に対応しており、偶数・奇数パリティにも対応している。この規格は KONDO KRS シリアル・サーボに対応しており、ICS3.0 に準拠した通信を実現している。これら PWM または UART 通信機能は FPGA 内部回路の構成を変更することにより、32 ch のサーボ通信用信号線に自由にマッピングすることが可能である。現在の実験用ロボットでは 8 ch を UART に、残り 24 ch を PWM に割り当てている。

UART は各サーボをデイジー・チェイン接続することが可能であり、片足 5 個のサーボは 1 ch の UART のみで接続されているため、実際に使用しているのは両足分で 2 ch である。PWM では各サーボに 1 ch を割り当てており、両腕及び頭部用として、計 9 ch が割り当てられている。

ALICE と Windows PC とは USB により接続されるが、ALICE の USB コントローラには FT245RL USB パラレル変換 IC を用いている。ALICE 内部では FT245RL は CPU のデータバスに直結されており、さらに FT245RL の割り込み要求信号線は CPU の割り込み入力に接続しているため、CPU・USB チップ間の通信を極力高速に行える構造を採用している。

また、USB チップメーカーFTDI から Windows 用仮想 COM ポート・デバイス・ド

Vol.2011-SLDM-151 No.1
Vol.2011-EMB-22 No.1
2011/7/3

ライバが用意されているため、ALICE システムでは、FTDI 純正仮想 COM ポート・ドライバ経由で Windows PC と ALICE との通信を行っている。

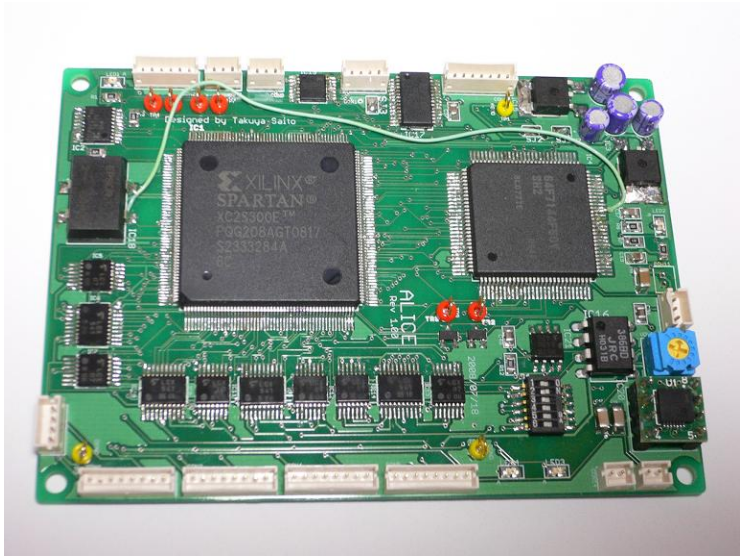

図 1　ALICE の外観

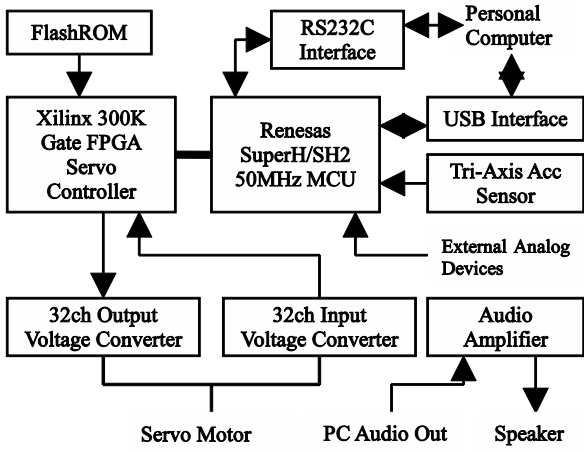

図 2　ALICE のブロック・ダイアグラム

Windows 用のプログラミング言語では、ほぼ全て COM ポートを制御することができるため、仮想 COM ポートを用いることにより、あらゆるプログラミング言語により、ALICE のコントロールを可能にしている。また、COM ポートとは言っても実際には USB 接続された仮想の COM ポートであるため、実際の通信速度は COM ポートに設定された 115200bps のような遅いものではなく、実測で約 1Mbps 程度の高速通信が行えていることから、仮想 COM ポートにおいても十分な通信速度が確保されていると考えられる。また、本論文は、この仮想 COM ポートを用いた通信で、Windows 上のプログラミング言語 C#を用いて作られたプログラムとの通信速度を測定している。それにより、非常に簡単にプログラム可能な C#により、ロボットのコントロールに必要十分な通信速度が実現できるかどうかについて実験により調査する。

表 1　ALICE の仕様

CPU	Renesas Technology SuperH/SH2 SH7144
クロック周波数	48MHz
FPGA	Xilinx Spartan2E XC2S300E 300K Gate
USB 規格	USB2.0 バスパワー対応
USB コントローラ	FTDI FT245RL
加速度センサ	秋月電子　3 軸加速度センサモジュール　KXM52-1050
PWM 制御範囲	0～20 ms
UART 通信方法	Pullup による半二重通信　8 bit　偶数・奇数パリティ
UART 通信速度	115200 bps
大きさ	100 mm　×　72 mm
重さ	約 30 g

4.　実験環境

4.1　Windows の環境

本実験には、現在開発中の Windows PC を搭載したヒューマノイド・ロボットを用いた。Windows PC には VIA の Pico-ITX EPIA-10000G を用いており、片足 5 自由度、片手 4 自由度、頭 1 自由度の計 19 自由度のヒューマノイド・ロボットである。実験用ヒューマノイド・ロボットの主な仕様を表 2 に示す。

表 2　実験用ヒューマノイド・ロボットの仕様

Windows PC 用マザーボード	VIA EPIA PX 5000EG Pico-ITX
Windows PC CPU	500MHz VIA Eden ULV Processor

Vol.2011-SLDM-151 No.1
Vol.2011-EMB-22 No.1
2011/7/3

Windows PC メモリ	1GB DDR2 533 SO-DIMM
Windows PC HDD	InnoDisk SATADOM 8GB
Windows OS	Windows XP
ロボット・コントローラ	ALICE Rev.1.00
頭部サーボモータ	JR PROPO DS386 PWM 制御 1 個
腕部サーボモータ	共立電子 プチブラケットサーボ WR-ES500 PWM 制御 8 個
足股部及び足首ロール軸	KONDO KRS-2552HV UART 制御 4 個
その他足部サーボ	KONDO KRS-4034HV UART 制御 6 個
大きさ	100 mm × 72 mm

Windows の測定用のプログラムは Windows XP 上で Visual Studio 2005 の Visual C# を用いてプログラムした。測定用の C#のプログラムは、プロセスの優先度を以下のようにしてリアルタイムに設定した。

System.Diagnostics.Process.GetCurrentProcess().PriorityClass =
System.Diagnostics.ProcessPriorityClass.RealTime;

また、測定用プログラムの実行は、Visual Studio 2005 のデバッグモードではなく、プログラム単体で実行した。ロボットの測定風景を図 3 に示す。

左側のロボットが ALICE と Pico-ITX マザーボードを内蔵した実験用ロボットであり、右側のディスプレイとキーボードはこのロボットに内蔵された Pico-ITX マザーボード上で動いている Windows XP の Visual Studio 2005 で測定プログラムを作成しているところである。また、FTDI の仮想 COM ポート・ドライバの設定は、BM オプションの待ち時間をデフォルトの 16ms から最速の 1ms に変更した。それ以外の設定は全てデフォルト設定である。

測定用プログラムは、C#プログラムの ALICE CONNECT というロボット／コントロール･プログラム上に BackgroundWorker コンポーネントで別スレッドとして動作させている。メイン・プログラムと通信プログラムとは別スレッドとして動作させるため、このように実使用に近い形で実装した。

4.2 ALICE の環境

ALICE 内蔵の SuperH/SH2 CPU のプログラムは、KPIT GNU により Windows 上の Renesas HEW により C 言語で作成されている。USB コントローラ FT245RL の送受信割り込み要求は SuperH/SH2 の割り込み入力に接続されており、送信レディや受信データが来た時には CPU へハードウェア割り込みをかけることにより､割り込みハンドラが呼び出されデータが処理される。

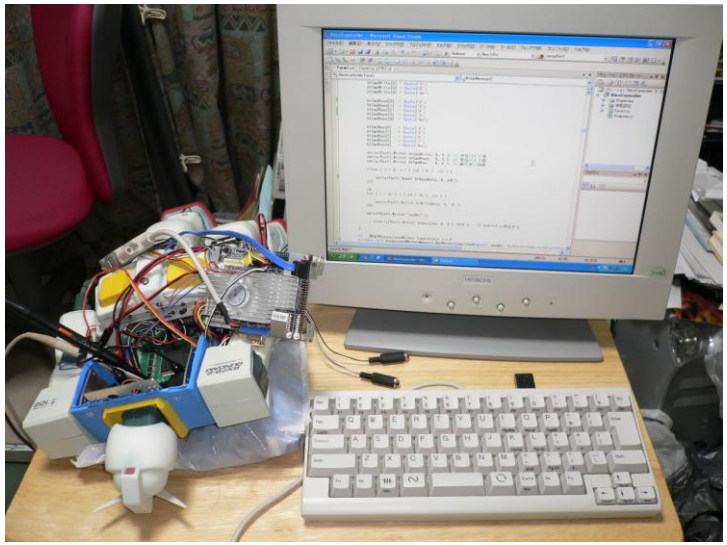

図 3 測定風景

4.3 測定用信号

測定用信号は、ALICE 上の測定用に用意された SuperH/SH2 の I/O ポートからタイミングを出力させることにより行った。

測定用端子は以下の順番で HIGH/LOW に変化するようにプログラムした。

1. Windows から測定開始コマンド及び測定用パラメータを受信する（15 バイト）
2. I/O ポートから HIGH を出力
3. USB データの送受信
4. Windows から測定終了コマンドを受信する（4 バイト）
5. I/O ポートから LOW を出力

4 の項目の後に I/O ポートの出力を LOW にしている理由は、これがないと、実際に Windows との USB データの送受信が終わる前に､バッファにデータが転送された段階で送受信終了と判断されてしまうためである。

4.4 測定機材

測定機材には、波形観察のためにオシロスコープを、そして、パルスの時間間隔を

Vol.2011-SLDM-151 No.1
Vol.2011-EMB-22 No.1
2011/7/3

統計的に測定するために Modulation Domain Analyzer（MDF）を用いた。測定に用いた機材を表 3 に示す。

表 3 測定機材

オシロスコープ	HP 54645A 100MHz Digital Storage Oscilloscope
Modulation Domain Analyzer	HP 53310A Modulation Domain Analyzer

5. 測定結果

MDF による測定結果の一例を図 4 に示す。

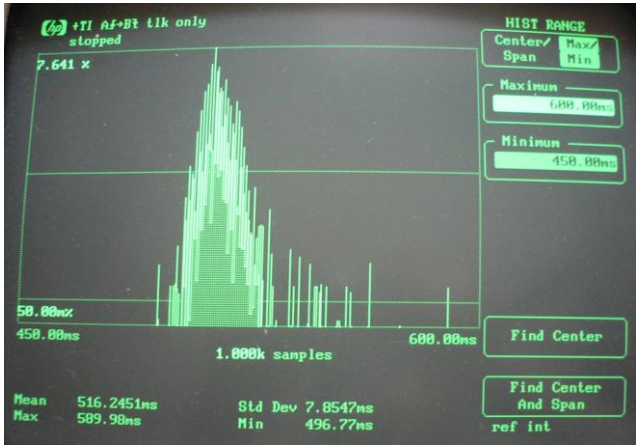

図 4 MDF による 2 バイトの Windows 送信時のジッタヒストグラム測定結果例

測定結果を図 5～図 7 に示す。それぞれ MDF により 1000 サンプル分データを測定した。MDF の測定結果から、パルスのジッタ図 4 のようになるため、はほぼ正規分布をしているとみなし、ジッタヒストグラムの標準偏差 σ も測定した。測定値の平均速度を Mean Speed、それに±3σ したものを図 5～図 7 に示した。

ALICE が Windows へデータを送信し、Windows が 1024 バイトのデータを受信したとき、C#で serialPort.Read()メソッド 1 回につき受信するパケットサイズを変化させることにより、通信速度がどのように変化するかを測定した結果を図 5 に示す。

図 5 より、1 回あたり 1 バイトで serialPort.Read()で受信すると、100kbps 以下しか速度がでないが、パケットサイズを大きくしていくに従って通信速度が上がってゆき、16 バイト単位で受信したときが最も通信速度が速くなり、約 1Mbps の通信速度に達している。それ以上パケットサイズを大きくすると、128 バイト以上は速度が一定となり、約 700kbps となっている。

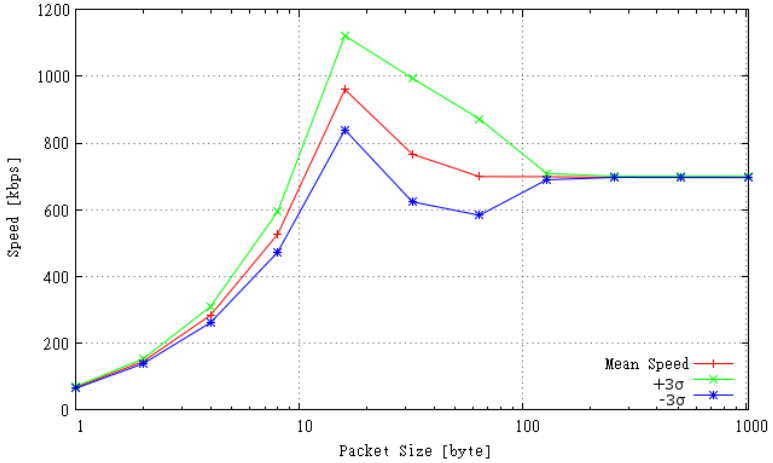

図 5 Windows データ受信速度のパケットサイズによる変化

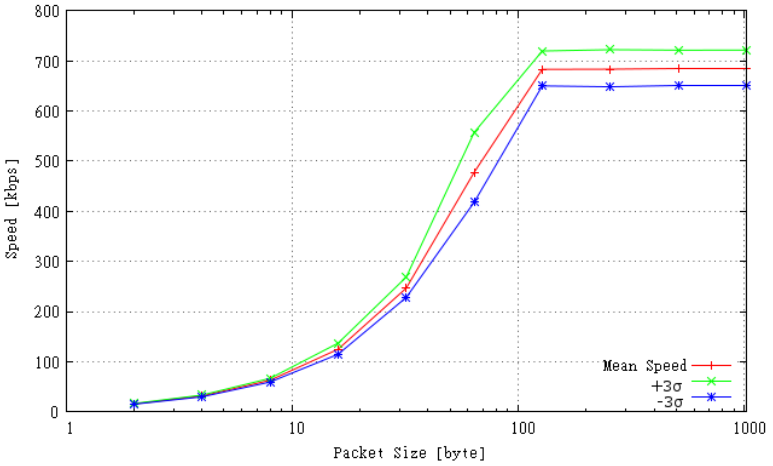

図 6 Windows データ送信速度のパケットサイズによる変化

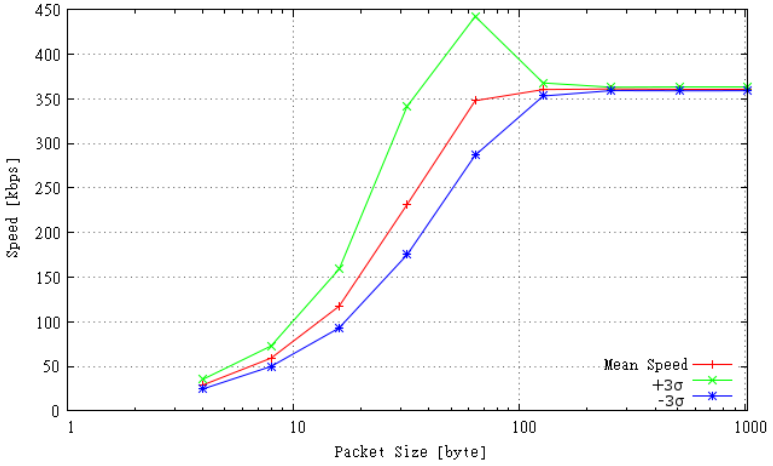

図 7 Windows データ送受信速度のパケットサイズによる変化

次に、Windows から ALICE へデータを送信し、ALICE が 1024 バイトのデータを受信したとき、C#で serialPort.Write()メソッド 1 回につき送信するパケットサイズを変化させることにより、通信速度がどのように変化するかを測定した結果を図 6 に示す。

図 6 より、2 バイトや 4 バイトでは数十 kbps 程度しか速度が出ていないが、パケットサイズが大きくなっていくに従って通信速度は速くなって行き、128 バイトで最高速度約 700kbps 弱に達している。それ以上パケットサイズを大きくしても通信速度に変化はほとんどない。

最後に、実際の通信では送信と受信を繰り返すため、Windows と ALICE 間で送信と受信各 1024 バイトを繰り返し、パケットサイズを変化させたときの総合的な通信速度の測定結果を図 7 に示す。

図 7 より、やはりパケットサイズが 4～8 バイト等小さいと 50kbps 程度しか速度が出ないが、パケットサイズが 64 バイトあたりでほぼ最高速度である 350kbps に達し、それ以上パケットサイズを大きくしても余り変化はないことが確認できた。

6. 結論

実験結果より、ALICE から Windows へのデータ送信ではパケットサイズを 16 バイト以上、Windows から ALICE へのデータ送信ではパケットサイズを 128 バイト以上に設定すると高速に Windows と ALICE 間で通信できることが分かった。そして、送受信を繰り返す場合では、送受信パケットサイズがそれぞれ 64 バイト以上では約 350kbps 程度の通信速度が確保できることが確認できた。

現在 ALICE ではヒューマノイド・ロボットの各アクチュエータ制御を 20ms 周期で行っているが、この場合、全 19 個のアクチュエータの角度情報の送信及び受信に、それぞれ最低 38 バイト必要となる。350kbps の通信速度で送受信それぞれ約 40 バイト通信すると仮定すると、1 フレームのデータ送受信が約 0.91ms で行えることになる。

現在は各サーボの制御周期は 20ms で行っているので、1 フレームのデータ送受信に約 1ms と仮定しても、1/20 の帯域でこれら通信を行うことができることになる。つまり、VIA Eden 500MHz という遅い CPU 上で動作する Windows XP で、動作速度が遅い C#により作られたプログラムと ALICE との通信においてであっても、通信速度は十分高速であり、ロボットのアクチュエータ制御が十分可能であることが分かった。また、図 5～図 7 より、パケットサイズが 128 バイト以上では、通信速度の±3σの範囲も非常に小さくなり、ジッタが十分小さく、ロボット制御用通信として十分な速度が十分小さいバラつき内に収まっていることが確認できた。

以上より、Windows PC を大脳、ALICE を人間の脳幹や脊髄とした ALICE ロボット・コントローラは、Windows と ALICE を間の通信速度やそのバラつき具合より、十分実現可能であることが分かった。実際には 16 フレーム（320ms）等、ある程度まとまったフレーム数のデータのやり取りをする構造をダブル・バッファリングにて実装しているため、別スレッドにより多少 Windows の通信スレッドのリアルタイム性が一時的に落ちたとしても、十分実現可能であると考えられる。

参考文献

1) 斎藤卓也, 村岡洋一: “高性能ロボット用総合プラットフォーム WR-X の開発”, 日本ロボット学会誌, Vol.27, No.1, pp.43-54, 2009.
2) T. Saito, Y. Muraoka: “Development of WR-X Vision, Image Analysis Accelerator for Robots,” Proc. of The 2007 International Conference on Embedded Systems & Applications (WORLDCOMP’07), pp. 148-154, June 25-29, 2007.
3) T. Saito, Y. Muraoka: “Development of Alice, a High-Performance Robot Controller Extended by Pico-ITX,” Proc. of the 2009 Int’l Conf. on Embedded Systems and Applications (ESA’09), pp.3-8, 2009.
4) Shik, M. L., Severin, F. V. & Orlovsky, G. N.:Control of walking and running by means of electrical stimulation of the midbrain, Biophysics, 11, pp.756-765 (1966).
5) 森茂美: 運動の制御Ⅰ, ブレインサイエンス, vol.7 No.2 pp.81-86
6) 森茂美: 運動の制御Ⅱ, ブレインサイエンス, vol.7 No.3 pp.91-98
7) 森茂美: 運動の制御Ⅲ, ブレインサイエンス, vol.7 No.4 pp.63-71
8) 森茂美: 運動の制御Ⅳ, ブレインサイエンス, vol.8 No.1 pp.83-96