論 文
56-6-24

ActiveX 모듈화를 통한 생체신호 실시간 가시화

Real-time Biomedical Signal Visualization by ActiveX Modulation

尹 泰 皓* · 金 敬 燮† · 辛 承 元** · 李 定 桓***
(Tae-Ho Yoon · Kyeong-Seop Kim · Seung-Won Shin · Jeong-Whan Lee)

Abstract - In this study, a hardware-independent software scheme is proposed to visualize biomedical signals such as an electrocardiogram(ECG) and their relevant diagnostic features in a real-time mode. To minimize the dependency on a specific hardware units and to maximize software portability into the different hardware platforms, objected-oriented visualization codes are implemented by Visual C++ MFC(Microsoft Fundamental Classes) with the integration of ActiveX modules.

Key Words : Biomedical Signal, Visual C++, Visualization, OOP(Object Oriented Programming), ActiveX

1. 서 론

현재, PC기반의 MS Windows 운영체제 환경에서 RS-232C 시리얼 포트와 TCP/IP 데이터 전송 프로토콜에 기반을 둔LAN 포트 그리고 제작사 자체의 개발된 하드웨어를 이용하여 생체신호를 입력받아 이를 컴퓨터 모니터와 프린터를 통해 가시화(visualization)하는 시스템과 소프트웨어들이 의학 분야의 연구 및 의학 진단에 일부 활용되고 있다(그림 1 참조).

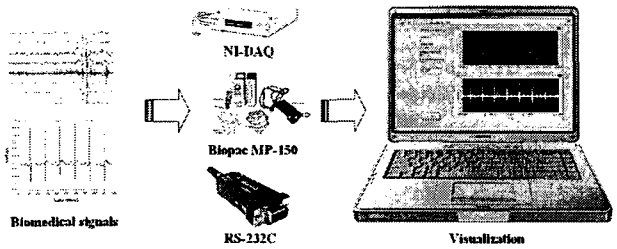

그림 1 특정한 하드웨어에 따른 생체신호 가시화 방법.
Fig. 1 Visualization of biomedical signal in a local hardware environment.

현재 널리 사용되고 있는 생체신호 가시화 소프트웨어는 NI(National Instrument)사의 자체 개발된 DAQ(Data Acquisition) 보드 또는 시리얼 포트와 연동하여 입력된 생체신호를 가시화하고 처리할 수 있는 해석 언어 프로그램인 랩뷰(LabView)[1] 그리고 Biopac[2]사의 생체신호처리 소프트웨어인 'Acknowledge'를 들 수 있다. 그러나 이러한 생체신호 처리 및 가시화 시스템은 관련된 소프트웨어의 구동을 위하여 고가의 하드웨어 장비를 필요로 하고 소프트웨어 또한 관련된 하드웨어 장비에 적합하게 해석코드가 설계되고 또한 구동되기 때문에, 특정한 하드웨어에 의존하지 않고 범용적인 목적으로 생체신호를 가시화하기 어렵다. 따라서 본 연구에서는 PC기반 MS(Microsoft) 윈도우즈 운영체제 환경에서 가장 넓게 사용되고 있는 객체지향 언어(Object Oriented Programming) 프로그래밍 언어인 Visual C++/MFC[3] 라이브러리를 이용하여 생체신호의 가시화 알고리즘을 설계하였으며, 이를 ActiveX 컨트롤 모듈로 변환하여 범용으로 활용 가능하도록 구현하였다. 또한 생체신호 가시화의 실제구현사례를 위하여 별도의 특정한 하드웨어 없이, MIT/BIH[4] 심전도 데이터베이스를 타이머 기능으로 해석, 심전도 신호를 가시화하고 또한 심전도 신호의 중요한 특징 파라미터인 PQRST 변곡점[5][6][7]을 추출하고, 이를 실시간으로 가시화하는 알고리즘을 구현하였다.

2. 방 법

2.1. 객체지향 프로그래밍

객체지향 프로그래밍[8]은 실세계의 현상을 컴퓨터 상에 객체로 실현(모델화)함으로써, 인간이 사물을 사고하는 방식과 유사하게 주어진 문제를 해결하기 위한 프로그램 기법으로서 객체에서 실체(데이터)와 그 실체와 관련되는 동작(절차, 방법, 기능)을 모두 포함한다. 기차역에서 승차권 발매의 예를 들면, 실체인 '손님'과 절차인 '승차권 주문'은 하나의

† 교신저자, 正會員 : 建國大學 醫學工學部 副敎授, 建國大學 醫工學 實用技術 硏究所 · 工博
E-mail : kyeong@kku.ac.kr
* 學生會員 : 建國大學 醫學工學部 博士課程
** 學生會員 : 建國大學 醫學工學部 博士課程
*** 正 會 員 : 建國大學 醫學工學部 助敎授 · 工博
接受日字 : 2007年 4月 26日
最終完了 : 2007年 5月 13日

객체이고, 실체인 '역무원'과 절차인 '승차권 발매'도 하나의 객체이다. 주어진 과제를 처리하기 위해 객체 간에는 메시지(지시)를 주고받는데 메시지를 받은 객체는 동작(절차)을 실행한다. C와 같은 기존 프로그램 언어에서는 동작과 절차를 중심으로 하고 실체를 종속적으로 취급했으나, 객체 지향 프로그램에서는 실체와 동작을 객체로 정의하고 객체 간의 메시지 교환에 주안점을 두어 정보를 처리한다. 즉, 객체 지향은 과정을 중시하는 절차 중심의 설계가 아니고, 실체를 중시하는 설계 방법이다. 객체 지향 프로그램의 또 하나의 중요한 특징은 공통의 성질을 갖는 객체는 같은 유형의 객체 등급으로 정의한다는 점인데, 이는 같은 등급에 속하는 객체들은 그들이 받는 메시지에 대하여 비슷하게 반응하기 때문이다. 객체 등급은 계층화할 수 있으며, 하위 계층의 객체 등급은 상위 등급의 성질과 기능을 계승한다. 따라서 객체 지향은 시스템의 모듈화, 캡슐화를 촉진하여 자칫 복잡화, 거대화되기 쉬운 소프트웨어를, 사용하기 쉽고, 작성하기 쉬우며, 또한 유지 보수하기 쉬운 방향으로 프로그램 코드를 재구축하는 기법이다.

2.2. 생체신호 실시간 가시화 구현 알고리즘

심전도와 같은 생체신호를 가시화 하는 방법은 디스플레이 화면상에서 데이터를 i)오른쪽 방향으로 그래프가 이동하며 가시화하는 RFFB(Rightwise First-in/First-out Buffering), ii)왼쪽 방향으로 데이터를 이동하며 가시화하는 LFFB (Leftwise First-in/First-out Buffering), 그리고 iii)데이터의 입력 순서대로 디스플레이 버퍼의 신호 값을 갱신하여 데이터를 디스플레이 화면에 가시화하는 IMB(Index Mapping Buffering) 등의 3가지 방법들로 구분된다.

2.2.1. RFFB(Rightwise First-in/First-out Buffering)

RFFB는 그림 2와 같이 생체신호 데이터의 값이 왼쪽방향부터 가시화되기 시작하여 결국 데이터 가시화 흐름의 방향은 화면상에서 왼쪽에서 오른쪽으로 이동하는 것과 같은 효과를 보여주는 방법을 나타낸다.

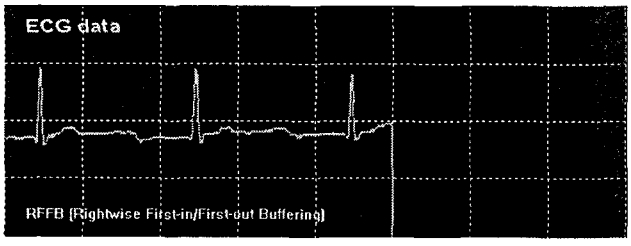

그림 2 RFFB 가시화.
Fig. 2 RFFB visualization.

RFFB 가시화 알고리즘의 구현을 위해서 먼저 생체신호 데이터 버퍼크기를 가시화할 화면의 크기에 맞게 설정하여, 화면의 폭과 동일한 크기의 디스플레이 버퍼를 해당 클래스의 멤버 변수로 선언한다. 데이터의 가시화 구현 속도는 시리얼 포트 혹은 LAN 포트를 통해 데이터가 입력되는 속도에 의존하며, 입력되는 데이터의 종류가 ASCII 포맷일 경우 디스플레이 버퍼에 바로 입력이 가능하여 변환과정이 필요 없지만, HEX 포맷일 경우 ASCII로 변환하여 디스플레이 버퍼에 입력하여 주는 과정이 필요하다. RFFB의 데이터 처리 알고리즘은 그림 3에서 표현된 데이터의 처리과정을 보이게 되며, 데이터를 가시화 구현을 위한 디스플레이 버퍼의 내용을 시리얼 포트 혹은 LAN 포트를 통해 데이터가 입력되는지를 관리하는 스레드 루틴(thread routine)에서 한번 완전한 데이터가 입력되는 순간 시간 t는 1이 증가하고 데이터 처리 방향에 따라서 데이터가 갱신된다. 마지막으로 새롭게 입력된 데이터를 디스플레이 버퍼의 어드레스 0번에 입력하면 프로세싱이 종료되며, 최종적으로 디스플레이 버퍼를 화면에 그래프의 형태로 출력하게 된다.

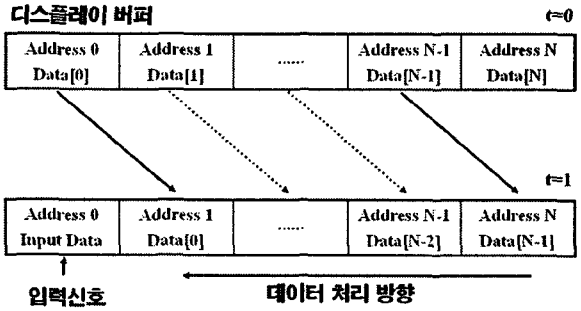

그림 3 RFFB 가시화를 위한 데이터 처리 알고리즘.
Fig. 3 Data processing algorithm of RFFB visualization.

2.2.2 LFFB(Leftwise First-in/First-out Buffering)

LFFB는 그림 4에서와 같이 RFFB와 반대로 생체신호 데이터의 값이 오른쪽 방향부터 가시화되기 시작하여 가시화되는 데이터 흐름의 방향이 오른쪽에서 왼쪽으로 이동하는 것과 같은 효과를 보여주는 방법을 나타낸다.

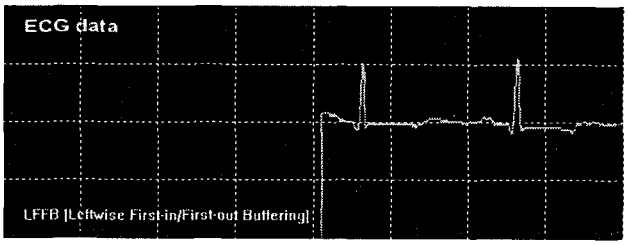

그림 4 LFFB 가시화.
Fig. 4 LFFB visualization.

LFFB 가시화 알고리즘의 구현을 위해서 RFFB의 가시화 알고리즘과 마찬가지로 먼저 출력화면의 크기를 설정하고, 화면의 폭과 동일한 크기의 디스플레이 버퍼를 해당 클래스의 멤버 변수로 선언한다. 데이터 가시화 속도는 시리얼 포트 혹은 LAN 포트를 통해 데이터가 입력되는 속도에 의존하며, 입력되는 데이터의 종류가 ASCII 포맷일 경우 디스플레이 버퍼에 바로 입력이 가능하여 별도의 변환과정이 필요 없지만, HEX 포맷일 경우 ASCII로 변환하여 디스플레이

버퍼에 입력하여 주는 과정이 필요하다. LFFB의 데이터 처리 알고리즘은 그림 5에서 표현된 바와 같이 RFFB 처리 알고리즘과 반대의 디스플레이 버퍼에서 데이터의 이동 과정을 보이게 되며, 데이터를 가시화 하고자 하는 디스플레이 버퍼의 내용을 시리얼 포트 혹은 LAN 포트를 통해 데이터가 입력되는지를 관리하는 스레드 루틴에서 한번 완전한 데이터가 입력되는 순간 시간 t는 1이 증가하고 데이터 처리 방향을 따라 데이터가 갱신된다. 마지막으로 새롭게 입력된 데이터를 디스플레이 버퍼의 어드레스 N번에 입력하면 프로세싱이 종료되며, 최종적으로 디스플레이 버퍼의 생체신호를 화면에 가시화 하게 된다.

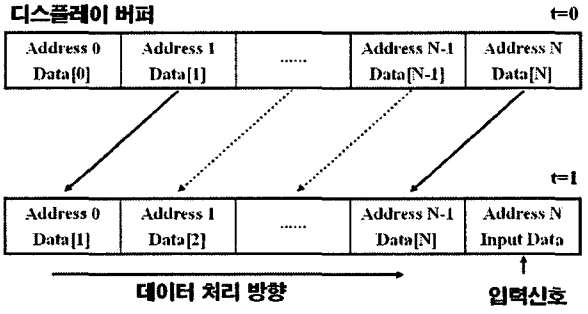

그림 5 LFFB 가시화를 위한 데이터 처리 알고리즘.
Fig. 5 Data processing algorithm of LFFB visualization.

2.2.3 IMB(Index Mapping Buffering) 가시화

앞 절에서 설명한 RFFB와 LFFB 가시화 알고리즘은 디스플레이 버퍼 내에서 데이터의 처리과정을 통해 생체신호를 가시화하는 것과 같은 효과를 보여주는 것이었으나 근본적으로 2가지의 문제점을 가지고 있다. 이중 첫 번째 문제점은 앞선 두 방법 모두 동일 버퍼 내에서 데이터를 이동하여 처리하는 과정이기 때문에 시간, 비용 그리고 효과의 측면으로 보았을 때 비효율적임을 알 수 있다. 또한 데이터 처리 과정 동안 시리얼 포트나 LAN 포트를 통하여 입력되는 데이터의 처리가 지연될 수 있는 문제점이 발생할 수 있다. 두 번째 문제점은 그림 2와 4를 비교해 보면 데이터가 역방향으로 가시화되는 것을 확인할 수 있다. 즉 RFFB 알고리즘의 경우 데이터의 입력이 왼쪽에서 들어오는 것처럼 가시화되기 때문에 예를 들어서 생체신호인 심전도의 경우 PQRST의 순서로 나타나는 심장활동의 특징이 역방향 즉, TSRQP 순서대로 가시화되는 문제가 발생하게 된다. 역방향으로 심전도의 특성이 가시화되면, 심전도 진단에 치명적인 오류가 발생할 수 있으므로 생체신호의 가시화에는 적합하지 않다. 따라서 위의 두 가지 문제점을 극복할 수 있는 방법으로 IMB 가시화 방법을 활용해야 한다. IMB 가시화 방법은 그림 6과 같이 고정된 그래프에서 순서대로 입력되는 한 개의 버퍼 값만 갱신하기 때문에 데이터의 이동 과정이 없고, 또한 생체신호의 특징을 쉽게 가시화할 수 있는 장점이 있다. 또한 데이터 이동으로 인한 지연이 없어 시간, 비용 그리고 효과의 측면으로 보았을 때 가장 효율적임을 알 수 있다.

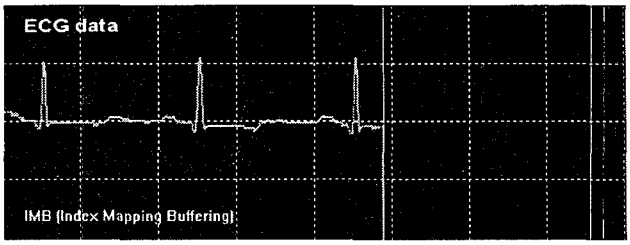

그림 6 IMB 가시화
Fig. 6 IMB visualization

IMB 가시화의 데이터 처리 알고리즘은 그림 7에서와 같이 디스플레이 버퍼에서 데이터의 이동과정이 존재하지 않는다. 다만 그래프 형태로 출력하고자 하는 디스플레이 버퍼의 내용을 시리얼 포트 혹은 LAN 포트를 통해 데이터가 입력되는지를 관리하는 스레드 루틴에서 한번 완전한 데이터가 입력되는 순간 시간 t는 1이 증가하게 되고 *index*의 값 역시 1 증가하여 디스플레이의 데이터 입력지점을 갱신한다. 최종적으로 *index*의 값이 디스플레이 버퍼의 크기보다 커지면, *index*는 다시 0으로 갱신되어 어드레스 0번의 주소 값에 입력한다. 즉 데이터의 이동과정을 생략하고, 갱신될 디스플레이 버퍼의 *index*를 이용하여 *index*의 주소에 위치한 1개의 버퍼 값만을 갱신함으로써 데이터 처리의 효율성이 극대화 된다. 최종적으로 디스플레이 버퍼를 화면에 라인 그래프의 형태로 가시화하게 된다.

그림 7 IMB 가시화를 위한 데이터 처리 알고리즘.
Fig. 7 Data processing algorithm of IMB.

2.2.4 생체신호 실시간 가시화

생체신호의 실질적인 가시화를 위하여 더블 버퍼링(double buffering)[9]과 그리드 설정(grid setting) 기법을 필요로 한다. 더블 버퍼링 기법은 그림 8(a)에서와 같이 DC(Device Context)에 그림을 표현 할 때 현재 화면을 담당하는 Visible DC에 직접 가시화는 것이 아니라, 그림 8(b)에서처럼 화면에 직접 가시화하지 않고 Hidden DC에 먼저 표현한 다음 Visible DC로 전송하는 방법으로 가시화를 위해 가장 많이 사용되는 프로그래밍 기법이다.

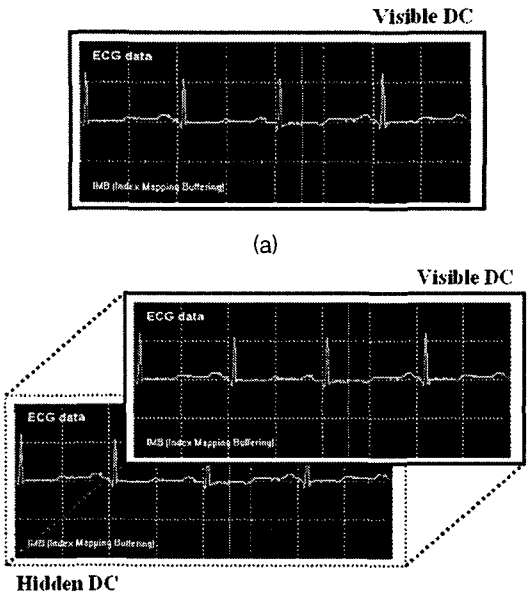

그림 8 싱글 버퍼링과 더블 버퍼링의 비교 개념도.
(a) 싱글 버퍼링
(b) 더블 버퍼링

Fig. 8 Schemes of double buffering and single buffering.

더블 버퍼링 알고리즘으로 생체신호를 처리하지 하지 않을 경우 화면이 깜박이는 현상이 발생한다. 즉 더블 버퍼링을 적용하는 근본적인 이유는 화면이 깜박이지 않는 것은 데이터를 메모리에 쓰는 속도와 비디오 카드의 메모리에 쓰는 속도의 차이 때문이다. 결국, 더블 버퍼링을 구현한다는 것은 2개의 별도 화면을 가지고 있는 것과 같다. 즉 첫 번째 화면에 미리 데이터 가시화를 하고, 두 번째 화면은 실제로 가시화 하는 화면이다. 결국 첫 번째 버퍼에 미리 그림을 그린 후 화면 버퍼로 고속 전송하며 그리는 중간 과정을 숨겨진 내부 버퍼에서 처리함으로써 사용자는 최종 결과만 보도록 한다. 그러나 첫 번째 버퍼에서 일어나는 일은 사용자에게 보이지 않기 때문에 그림이 아무리 복잡해도, 화면을 다 지운 후 다시 그리더라도 깜박임을 전혀 목격할 수가 없다. 뿐만 아니라 그리는 순서에 따라 이미지간의 수직적인 아래위를 지정할 수 있으며 여러 개의 이미지를 동시에 움직이는 것도 아주 부드럽게 처리할 수 있다.

다음으로 가시화 영역의 그리드 설정은 시간에 따른 생체신호의 변화를 판단하기 위해 현재 입력되는 생체신호의 샘플링 비율을 이용하여 그림 9와 같이 특정한 구간을 점선의 형태로 표현하는 것을 나타낸다. 가시화 영역에서의 그리드 설정과 더블 버퍼링의 기법을 적용하여 생체신호의 가시화 하는 순서는 먼저 2개의 DC를 생성한 후, 화면에 보이지 않은 Hidden DC에 펜 클래스(CPen)와 브러쉬 클래스(CBrush)를 설정하고, DC의 멤버 함수인 ::MoveTo()와 ::LineTo()를 이용하여 그리드 출력과 생체신호 데이터를 출력 한다. 다음으로 Hidden DC에 그려져 있는 내용을 설정 화면의 크기에 맞게 비트맵화(bitmapped) 하여 추출한 후, 화면에 보이는 Visible DC에 비트맵 이미지를 붙여 넣어 출력하게 된다.

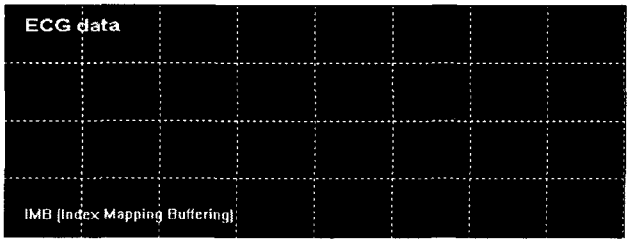

그림 9 가시화 영역의 그리드 설정.

Fig. 9 Grid setting of visualization region.

2.2.5 생체신호의 이득 값 조절 및 오프셋 설정

일반적으로 생체신호는 그 크기가 mV 단위로 매우 작다. 따라서 이득(gain) 값을 조절하여 생체신호 가시화의 범위를 효율적으로 설정해 주어야 하는데 mV의 경우 실수형(float type)으로 값이 입력되지만 실제 화면상에서는 설정한 화면의 출력단위인 화소(pixel)의 크기를 표현할 수 있는 부호 없는 정수형(unsigned integer type) 데이터로 변환되어야 한다. 따라서 크기가 작은 실수형의 값을 화면 내에 표현 가능한 정수형 값으로 변환하기 위하여 이득 값을 조절해 주어야 한다. 또한 생체신호가 양자화(Quantization)되어 있을 경우 그 값은 음수를 가질 수 없으므로, 베이스라인이 원하는 위치에 고정되지 않는다. 따라서 베이스 라인을 조절하기 위하여 오프셋 설정을 하여야 한다. 그림 10(a)는 생체신호인 심전도 신호에서 이득 값과 오프셋이 설정되지 않은 결과를 보여주며, 그림 10(b)는 이득 값과 오프셋을 설정한 결과를 보여준다. 본 연구에서는 생체신호 실시간 가시화를 위하여 그림 11에서와 같이 심전도 생체신호를 타이머 기능(Timer method)을 이용하여 실시간으로 출력하였고, ActiveX 컨트롤[10]로 변환이 용이하도록 버퍼링 종류, 가시화 방법,

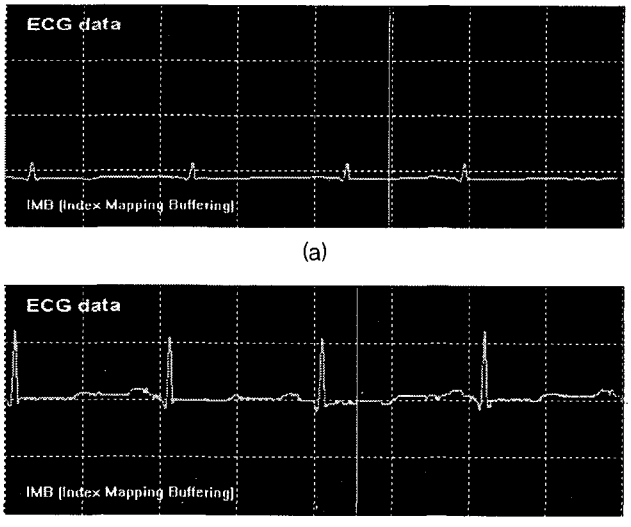

그림 10 생체신호의 이득 값 및 오프셋 설정.
(a) 이득 값과 오프셋이 설정 안 된 생체신호 출력.
(b) 이득 값과 오프셋이 설정된 생체신호 출력.

Fig. 10 Gain and off-set manipulation.

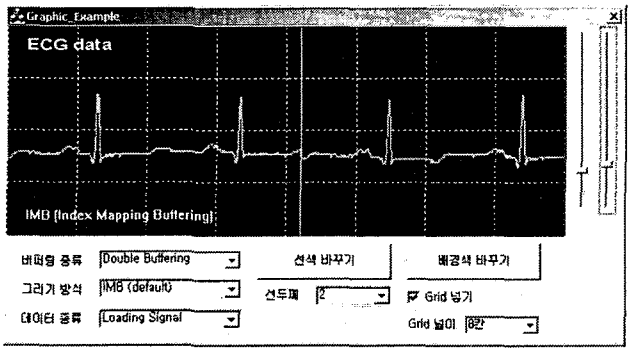

그림 11 심전도 신호의 실시간 가시화
Fig. 11 Realtime Visualization of ECG signal.

선색 변경, 배경색 변경, 선두께 변경, 그리드 설정 변경 등의 옵션을 설정할 수 있도록 구현 하였으며, 또한 이득 값 설정과 오프셋 설정을 할 수 있도록 하였다.

2.3. ActiveX 컨트롤

ActiveX 컨트롤은 독립된 사용자 인터페이스를 갖는 COM 서버의 일종이다. 예를 들면, MFC의 컨트롤 대부분이 ActiveX 컨트롤로 만들어져 있으며, 또한 ActiveX 컨트롤은 Visual C++로 만든 프로그램 뿐 아니라 Visual Basic과 같은 다른 개발 도구에서도 쉽게 가져다 쓸 수 있으며, 웹 페이지에도 끼워 넣을 수 있다. ActiveX 컨트롤은 여러 가지 COM 기술이 총체적으로 결합되어 만들어지며, 가장 널리 사용되고 있는 COM 기술의 결정판이라고 할 수 있다. ActiveX 컨트롤은 단독으로 실행되는 프로그램이 아니라 다른 프로그램에 삽입되어 동작하기 때문에 ActiveX 컨트롤을 동작시키기 위한 프로그램을 컨테이너(container)라고 한다. ActiveX 컨트롤과 컨테이너는 COM 인터페이스를 통해 통신하며 마치 한 몸인 것처럼 동작한다.

ActiveX 컨트롤과 컨테이너는 그림 12, 13에서 표현한 바와 같은 표준 COM 인터페이스를 사용한다.

ActiveX 컨트롤의 기능은 프로퍼티(property), 메서드(method), 이벤트(event)에 의해서 결정된다. 즉 프로퍼티나 메서드를 호출함으로써 ActiveX 컨트롤의 동작을 제어할 수 있고, 이벤트를 발생시킴으로써 자신에게 어떤 일이 일어났는지 컨테이너에게 알릴 수 있다.

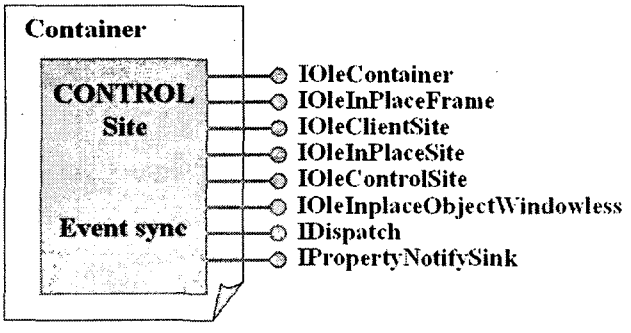

그림 12 ActiveX 컨테이너.
Fig. 12 ActiveX container.

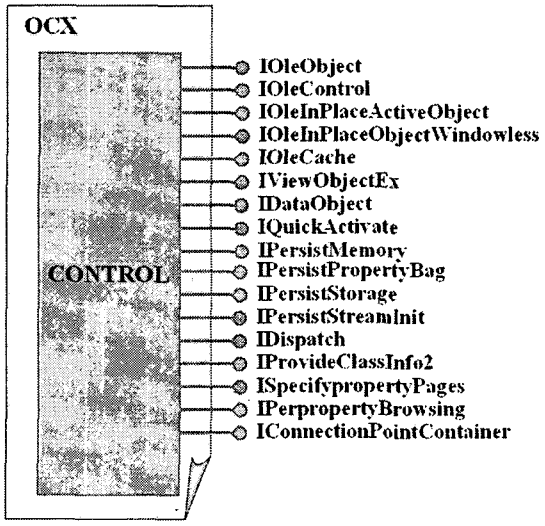

그림 13 ActiveX 컨트롤.
Fig. 13 ActiveX control.

3. 결 과

본 연구에서는 앞서 제시한 생체신호 가시화 알고리즘을 최종적으로 설계하고, ActiveX 컨트롤로 제작하여 향후 제작된 ActiveX 컨트롤의 등록만으로 쉽게 생체신호를 시각화 할 수 있는 방법을 구현하고자 하였다. ActiveX 컨트롤은 MFC ActiveX Control Wizard에서 초기설정을 통해 프로젝트를 생성하고, Dialog의 전체크기를 출력 화면의 크기로 정하여 가변적으로 화면의 크기가 조절 될 수 있도록 하였으며, 그림 14에서와 같이 생체신호 가시화 ActiveX 컨트롤 모듈의 내부를 설계하였다. 또한 생체신호 가시화 알고리즘이 구현된 ActiveX 컨트롤을 외부의 클래스에서 제어할 수 있도록 ActiveX 컨트롤 모듈 내에 프로퍼티를 설정해 주었다. 설정된 프로퍼티는 생체신호를 ActiveX 컨트롤의 내부 멤버변수에 입력시켜주는 함수, RFFB, LFFB, 그리고 IMB 중 가시화 하고 싶은 방법을 설정해 주는 함수, ActiveX 컨

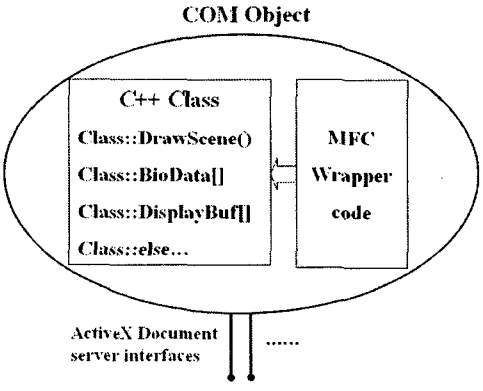

그림 14 생체신호 가시화를 위한 ActiveX 컨트롤 모듈
Fig. 14 ActiveX control module for biomedical signal visualization.

트롤 모듈 내에서 가시화되고 있는 출력 값의 이득을 조절해 주는 함수, 가시화되고 있는 출력 값의 오프셋을 조절해 주는 함수, 그래프의 선색을 설정해 주는 함수, 배경의 색을 설정할 수 있는 함수, 그리드의 크기와 그리드 출력여부를 설정해 주는 함수, 원하는 위치에 텍스트를 출력하도록 설정해 주는 함수, ActiveX 컨트롤 모듈 내에 디스플레이 버퍼에 저장된 데이터를 외부의 클래스에서 읽어낼 수 있는 함수로 구성하였다 (그림 15 참조).

ActiveX 컨트롤 모듈과의 프로퍼티 연결은 MFC의 add property 기능을 이용하여 쉽게 설정이 가능하다. 즉 ActiveX 컨트롤 모듈에 프로퍼티를 설정하는 'Set Method'와 ActiveX 컨트롤 모듈로부터 결과 값을 반환받는 'Get Method'가 한 번의 설정만으로 동시에 생성된다.

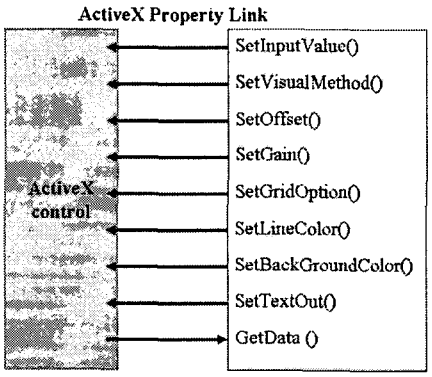

그림 15 생체신호 가시화를 위한 ActiveX의 프로퍼티 연결.
Fig. 15 ActiveX Property Link for biomedical signal visualization.

실제 ActiveX 컨트롤의 형태로 제작된 실시간 생체신호 가시화 모듈의 실험을 위해 MIT/BIH 심전도 데이터베이스를 로딩한 후, MFC의 Timer() 함수를 이용하여 실제 하드웨어에서 PC에 데이터가 입력되는 것과 동일한 조건을 만들어 생체신호를 가시화하고, 심전도 신호의 중요 파라미터인 PQRST를 실시간으로 검출하여 가시화 하도록 설계하였다 그림 16의 ActiveX 컨트롤을 이용한 실험 프로그램에서 GUI 화면에서 위에 배치된 생체신호 가시화 부분이 ActiveX 컨트롤로 설계된 생체신호 가시화 모듈이다. 가시화 방법은 RFFT로 설정되었으며, 그리드 설정은 가로를 12개의 화면으로 분할하고, 세로를 3개로 분할하였다. 그리고 MIT/BIH 심전도 데이터베이스에서 추출한 심전도 신호는 양자화 되어있기 때문에 좌측 하단의 그래프 우측과 하단에 MFC의 슬라이드 컨트롤을 이용하여 이득 값과 오프셋을 설정할 수 있도록 하였다. 좌측 하단의 그래프는 PQRST 검출 알고리즘[5][9][10]을 적용하여 ActiveX 컨트롤을 사용하지 않고 LFFB 가시화 방법으로 가시화 한 결과를 나타낸다. 우측 하단은 MFC의 리스트 컨트롤을 사용하여 심전도의 중요 파라미터인 R-R 간격, 심박동율 그리고 검출된 PQRST 변곡점 각각의 검출 시간을 리스트화 하여 저장하고 이를 보여주고 있다.

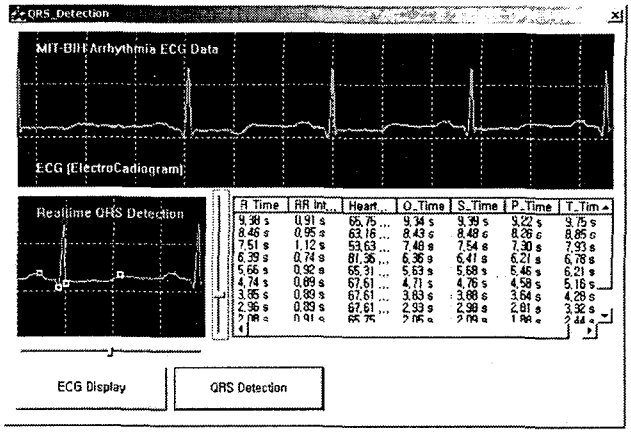

그림 16 ActiveX 컨트롤을 이용한 실험 프로그램.
Fig. 16 Visual C++ test program integrated by ActiveX control.

표 1은 ActiveX 컨트롤의 프로퍼티 값을 설정하기 위해서 구현된 ActiveX의 내부 옵션 값을 보여주고 있다. 실제로 생체신호 데이터의 가시화를 위해서는, 데이터를 ActiveX 컨트롤의 내부 버퍼에 입력시켜주는 함수인 SetInputValue() 함수를 통해 입력 순서대로 입력하여 가시화 한다. ActiveX 내부 버퍼에 입력된 데이터는 GetData() 함수를 이용하여 외부의 클래스로 반환받을 수 있다. 따라서 추가적인 메모리의 할당 없이 생체신호 가시화 ActiveX 컨트롤만으로 실시간 신호처리 알고리즘 구현이 가능하다.

표 1 생체신호 가시화 ActiveX 컨트롤 모듈의 내부 연결 함수
Table 1 ActiveX property Link functions for biomedical signal visualization

함수 이름	인 자	기 능
SetInputValue()	double INPUT	생체신호 가시화 ActiveX의 내부 버퍼에 데이터 입력
SetVisualMethod()	short METHOD	생체신호 가시화의 3가지 방법 중 1개를 설정(default IMB)
SetOffset()	double OFFSET	가시화되는 생체신호 데이터의 오프셋 변경
SetGain()	double GAIN	가시화되는 생체신호 데이터의 이득 값 조절
SetGridOption()	bool GSET, double GSIZE	그리드 설정여부와 설정 시 그리드 간격 지정
SetLineColor()	COLORREF LColor	출력되는 생체신호의 색 변경
SetBackGroundColor()	COLORREF BColor	출력되는 공간의 배경색 설정
SetTextOut()	CPoint Locate, CString Text	출력되는 텍스트의 위치와 입력 텍스트 설정
GetData()	반환 값 (char Data[])	데이터 배열을 ActiveX 컨트롤에서 반환 값으로 받음

5. 결 론

본 연구는 객체지향 언어 중 가장 넓게 사용되고 있는 프로그래밍 언어인 Visual C++ 을 이용하여 효율적인 생체신호 가시화 알고리즘 설계하고, 이 알고리즘을 ActiveX 컨트롤 모듈로 변환하여 손쉽게 GUI를 설계할 수 있도록 구현하였으며, 여러 종류의 비주얼 프로그래밍 언어에서 범용으로 사용가능하도록 설계하였다. 또한, 제작된 생체신호 가시화 ActiveX 모듈을 사용하여 MIT/BIH 심전도 데이터베이스를 이용, 심전도 신호를 PC상에 가상으로 입력받아 PQRST 변곡점을 검출하였고, 이를 이용하여 심전도의 중요 파라미터인 R-R 간격, 심박동율 그리고 검출된 PQRST 변곡점 각각의 검출 시간을 계산하여 리스트화 하였다. 결과적으로 시리얼포트와 LAN 포트를 연동할 수 있는 MFC 클래스를 구현, 이를 ActiveX 컨트롤 모듈로 변환함으로써 실시간으로 생체신호를 측정하고 이를 가시화 할 수 있다.

감사의 글

본 연구는 정보통신부 및 정보통신 연구진흥원의 정보통신 선도 기반기술 개발사업의 연구결과로 수행되었습니다.

참 고 문 헌

[1] 곽두영, LABVIEW EXPRESS(고급), OHM사, 2003.
[2] http://www.biopac.com/
[3] 김용성, Visual C++ 6 완벽가이드 2nd Edition, 학술정보, 2004.
[4] http://ecg.mit.edu/
[5] Webster, John G, Encyclopedia of medical devices and instrumentation, New York : Wiley, press, 1988.
[6] Jiapu Pan, Willis J. Tompkins, "A Real-Time QRS Detection Algorithm", Biomedical Engineering, IEEE Transactions on Volume BME-32, Issue 3, March 1985 pp. 230 - 236
[7] Patrick S. Hamilton, Willis J. Tompkins, "Quantitative Investigation of QRS Detection Rules Using the MIT/BIH Arrhythmia Database" Biomedical Engineering, IEEE Transactions on Volume BME-33, Issue 12, Dec. 1986 pp. 1157 - 1165
[8] 최영근, 김경섭, 김일민, 안동언, 하숙정, 객체지향 원리로 이해하는 ABSOLUTE C++, 학술정보, 2006.
[9] Richard S. Wright, Jr. Micheal Sweet, OpenGL SUPERBIBLE Second Edition, 인포북, 2000.
[10] 정현주, ACTIVEX와 OLE, 비앤씨, 1997.

저 자 소 개

윤 태 호 (尹 泰 皓)
2003년 건국대학교 의학공학부 졸업. 동대학원 석사(2005). 2005년~현재 동대학원 박사과정 재학 중.

김 경 섭 (金 敬 燮)
1979년 연세대학교 전기공학과 졸업. 동대학원 석사(1981). Louisiana State University, Electrical Engineering, M.S. (1985). The University of Alabama in Huntsville, Electrical & Computer Engineering, Ph.D. (1994). 2001년~현재 건국대학교 의학공학부 부교수.

신 승 원 (辛 承 元)
2005년 건국대학교 의학공학부 졸업. 동대학원 석사(2007). 2007년~현재 동대학원 박사과정 재학 중.

이 정 환 (李 定 桓)
1992년 연세대학교 전기공학과 졸업. 동대학원 석사(1994) 및 박사(2000). 2004년~현재 건국대학교 의학공학부 조교수.