



PCAN_Developer

PCAN-Developer 는 CAN 과 CAN FD 연결을 통해 Windows 응용 프로그램들을 만들 수 있습니다. 특히 API는 프로그래밍 언어 C, C ++ 및 Delphi 용 헤더 파일 및 샘플 프로젝트뿐만 아니라 32 비트 및 64 비트 인터페이스 DLL을 통한 광범위한 프로그램 라이브러리를 제공합니다. 조만간 C #, C ++ / CLR 및 Visual Basic과 같은 .NET 호환 언어도 추가될 예정입니다. 이러한 것들은 .NET에서 PCAN-API를 간단히 사용하기 위해 고안된 특수 .NET 어셈블리에 의해 구현됩니다.

```
CanApi4.h - Microsoft Visual Studio
File Edit View Project Debug Team Tools Test Analyze Window Help
CanApi4.h
702 // Message types
703
704 #define CAN_MSGTYPE_STANDARD 0x0000 // Standard Data frame (11-bit ID)
705 #define CAN_MSGTYPE_EXTENDED 0x0001 // 1, if Extended Data frame (CAN 2.0B, 29-bit ID)
706 #define CAN_MSGTYPE_SELFRECEIVE 0x0002 // 1, if message shall be/has been self-received by the controller
707 #define CAN_MSGTYPE_HW_SELFRECEIVE 0x0004 // 1, if selfreceive was performed by hardware. 0, if hardware is incapable of selfreceive
708 #define CAN_MSGTYPE_SINGLESHOT 0x0008 // 1, if no re-transmission shall be performed for the message (self ACK)
709 #define CAN_MSGTYPE_BRS 0x0010 // bit rate switch: 1, if CAN FD frame data was sent with higher bit rate
710 #define CAN_MSGTYPE_ESI 0x0020 // error state indicator: 1, if CAN FD transmitter was error active
711
712 // Type definitions
713
714 typedef UINT8 HCANHW; // type 'Hardware handle'
715 typedef UINT8 HCANNET; // type 'Net handle'
716 typedef UINT8 HCANCLIENT; // type 'Client handle'
717 typedef UINT8 HCANOBJECT; // any handle type
718 typedef UINT32 can_status_t; // status value/return code
719
720 // PCAN device types
721 #define enum can_device
722 {
723     pcan_unknown,
724     pcan_isa,
725     pcan_pci,
726     pcan_usb,
727     pcan_pccard,
728     pcan_virtual,
729     pcan_lan,
730     pcan_dng,
731     dci_can
732 } can_device_t;
733
734 #pragma pack(push, 1) // the following records are byte-aligned
735
736 // common record header
737 #define struct can_recordheader_struct
738 {
739     UINT16 size; // #0 +0x00 absolute length of record in bytes
740     UINT16 type; // #1 +0x02 type code of the record (only LSBs with mask 0x3fff)
741 } can_recordheader_t;
742
743 // Record "basedata"
744 // type = CAN_RECORDTYPE_basedata = 0x1000
745 // hierarchy = Is a base record for these other records:
746 // basemsg, hwstatus, errorframe, errorcounter_decrement, busload.
747 // size = 22 = 0x16 bytes, aligned to 1 bytes
748 // info = Abstract base class
749 // Identical header for all data records (msg and events),
750 // so all records can be cast to this
751 //
752 //
```

소위 클라이언트를 통한 CAN 통신 덕분에 많은 것들이 가능해졌습니다. 그 중 하나는 최대 64 개의 채널이 있는 최대 64 개의 애플리케이션들이 PEAK 시스템의 하드웨어 를 통해 통신 할 수 있습니다. 따라서 예를 들어 최대 64 개의 PCI interface card 또는 64 개의 USB interface card 연결들이 동시에 실행될 수 있습니다.

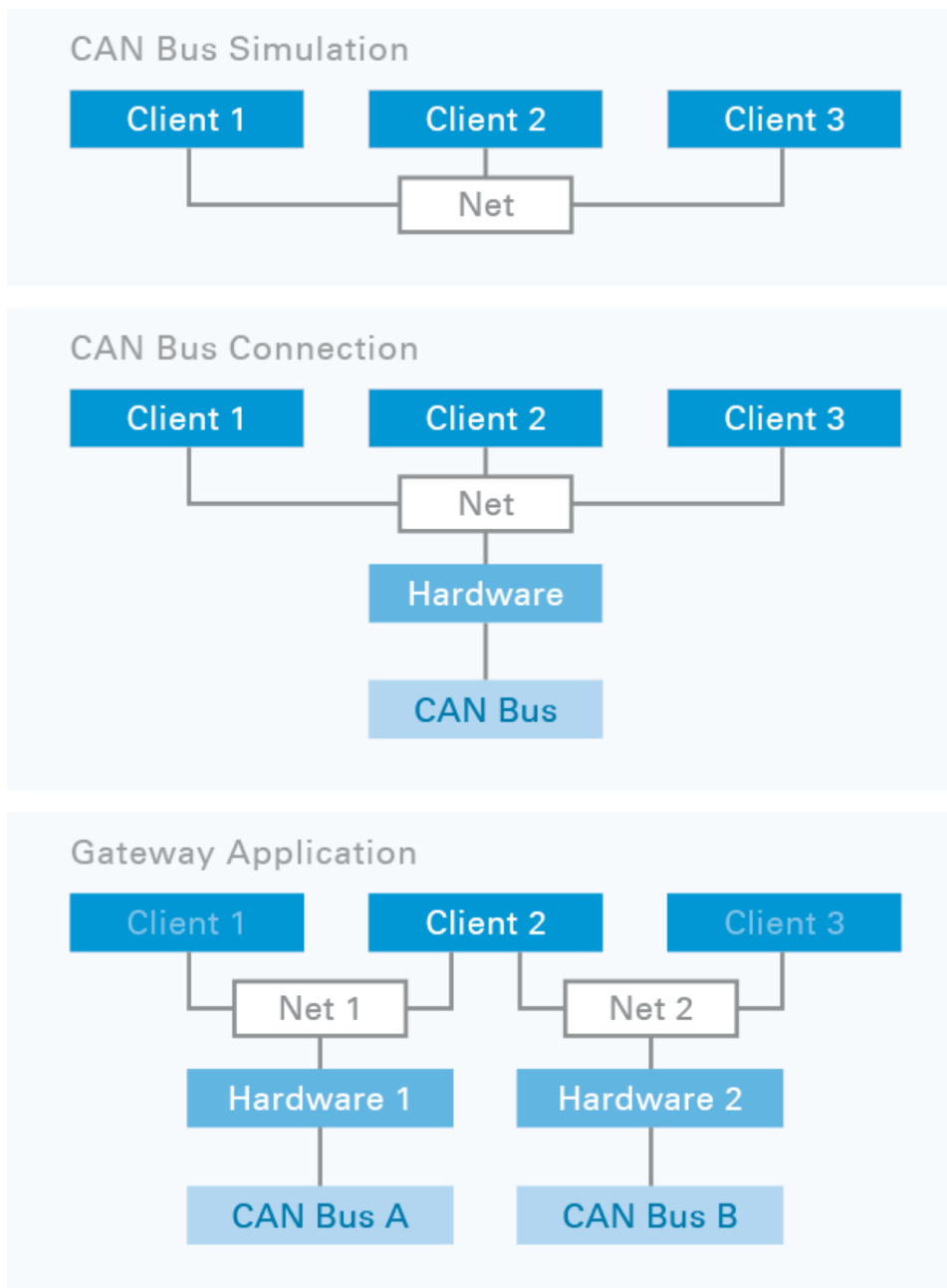
또한 시뮬레이션 된 CAN 및 CAN FD 네트워크가 애플리케이션과 클라이언트간에 구현 될 수 있습니다. 이 경우 테스트 목적으로 CAN 오류 프레임과 같은 오류들이 생성될 수 있습니다. 또한 전문 PCAN-API는 CAN 트래픽 모니터링을위한 listen-only 모드와 같은 모든 하드웨어 매개 변수들에 대한 액세스를 제공합니다. 또한 PCAN-LAN 드라이버를 통해 PCAN-Gateway의 CAN 채널에 액세스 할 수 있습니다.



PCAN-View, PCAN-Status Display 및 PCAN-Nets Configuration과 같은 관련 툴들과 함께 개별 시스템 구성에 유용하게 사용할 수 있는 PCAN-Developer는 Windows 용의 복잡한 응용 애플리케이션들을 개발하기 위한 전문적인 다용도 패키지입니다.

클라이언트를 통한 CAN 연결

PCAN-API를 기반으로하는 애플리케이션은 Nets에 액세스하기 위해 소위 클라이언트를 사용합니다. Net은 적절한 하드웨어를 통한 외부 CAN 버스 연결과 여러 애플리케이션들의 상호 연결을 포함합니다.





다음과 같은 것들을 클라이언트를 통한 CAN 연결에 활용합니다:

- 하나 또는 여러 개의 클라이언트를 인터넷에 연결할 수 있습니다.
- 하나의 클라이언트는 여러 개의 네트들과 연결할 수 있습니다.
- 한 개의 네트는 연결되지 않은 또는 정확히 하나의 활성 하드웨어에 연결됩니다
- 하나의 하드웨어에 대해 서로 다른 네트들에 대한 다중 연결을 정의 할 수 있습니다
- 네트에 정의된 최대 한 개의 연결이 하나의 하드웨어에 대해 활성화될 수 있습니다.
- 클라이언트가 전송할 때, 메시지는 네트에 연결된 다른 모든 클라이언트와 하드웨어를 통해 외부 CAN 버스로 전달됩니다
- 하드웨어가 메시지를 수신하면 네트에 연결된 모든 클라이언트가 메시지를 수신합니다

PCAN-Developer 기능 및 특징

- CAN 및 CAN FD 연결을 이용한 어플리케이션 개발을 위한 전문 API
- CAN 규격 2.0 A / B 와 FD 준수
- ISO 와 비 ISO 표준에 대한 CAN FD 지원이 전환 가능.
- 운영 체제 Windows® 10, 8.1, 7 (32 / 64-bit)
- 물리적 CAN 채널 당 최대 64 개의 애플리케이션을 동시에 실행 가능
- 지원되는 모든 하드웨어 타입에 대한 프로그래밍 인터페이스 (API)로 32 비트 또는 64 비트 용 Windows® DLL 사용
- 각각의 하드웨어 타입은 최대 64 개의 채널 사용
- Nets를 사용한 PCAN PC 하드웨어 채널들 사이의 간편한 전환
- 새로운 PCAN LAN 하드웨어 타입을 통하여 PCAN Gateway의 CAN 채널들을 액세스
- 클라이언트 당 최대 32,768 CAN 메시지의 드라이버 내부 버퍼링
- API를 통한 버퍼 크기 설정 가능
- 최대 1 μ s 까지 정밀한 수신 메시지에서의 타임 스탬프 (사용되는 PEAK CAN 인터페이스에 따라 다름)
- listen-only모드와 같은 사용 가능한 모든 하드웨어 매개 변수에 대한 액세스
- 메시지 수신시 Windows® 이벤트를 통한 응용 프로그램 알림
- 드라이버 이벤트를 통한 응용 프로그램 알림. 변경된 하드웨어 매개 변수, PCAN PC 하드웨어의 플러그 인 / 플러그 아웃 또는 버스 로드 정보
- 드라이버로 인터럽트 트리거된 CAN 메시지의 수신. 타임 스탬프를 이용하여 FIFO 버퍼에 저장
- 메시지를 전송할 때 타임 스탬프를 지정하여 전송 시기 제어
- 단일 샷 및 자체 수신 요청 전송 설정
- CAN 오류 프레임 지원
- 클라이언트를 통한 애플리케이션 들 간의 모의 CAN 통신
- 클라이언트별로 메시지 필터를 자유롭게 정의
- 연결 활성화 중 하드웨어 재설정
- 제어판 또는 응용 프로그램을 이용하여 하드웨어 설정
- 실행 가능한 샘플 소스 코드와 PDF 형식의 상세한 설명서



- CAN 시스템 구성, 매개 변수화 및 시각화를 위한 도구들 모음
- Thread-safe API

시스템 설치 요건

- Windows® 10, 8.1, 7 (32/64-bit)
- 최소 2 GB RAM 과 1.5 GHz CPU
- CAN bus 연결 : PEAK-System 사의 PC CAN 인터페이스 이용

주의: Parallel port CAN 인터페이스들은 32 비트 시스템에서만 지원

PCAN-Developer 라이선스

PCAN-Developer 에는 개발 패키지를 사용할 수 있는 한 개의 single user 라이선스와 PCAN-Developer 를 배포할 수 있는 한 개의 distribution 라이선스가 포함되어 있습니다. 이 패키지에는 소프트웨어 제품인 PCAN-View, PCAN-Nets Configuration, PCAN-Status Display 뿐만 아니라 API DLL 도 들어 있습니다.