BACnet International

BACnet Development

With Free and Open Source Software

Overview

Why choose free software?

Open source BACnet Protocol Stack libraries & their Licenses

Open Source and Free Compilers

Wireshark, MS/TP Capture

Testing Tools

Why Choose Free Software?

"When we call software “free,” we mean that it respects the users' essential freedoms: the freedom to run it, to study and change it, and to redistribute copies with or without changes. This is a matter of freedom, not price, so think of “free speech,” not “free beer.”” - RMS

A free BACnet reference that can be studied and changed is born.

For a protocol that is constantly changing, sharing the cost of maintaining a high quality protocol stack with others around the world is a good solution.

FOSS Development Model

Collaborative process

Trusted repository

Trusted Developers

Bug reports and bug fixes

Enhancements and contributions merged into trusted repository

Cost sharing between users, typically only for improvements

Community mailing list, forum, help requests

BACnet Protocol Stacks

"The protocol stack is an implementation of a computer networking protocol suite. The terms are often used interchangeably. Strictly speaking, the suite is the definition of the protocols, and the stack is the software implementation of them."

Free Software	Proprietary Software
Yet Another BACnet Explorer (C#) (MIT) VTS (C++ MFC) (Public Domain) BACnet4Linux (C Linux App) (LGPL) BACnet Firewall Router (C Linux App) (PD) BACpypes (Python) (Public Domain) BACsharp (C#) (Public Domain) BACnet4J (Java) (GPL) BACnet Protocol Stack (C) (GPL w/except)	BACnet Protocol Stack SDK (Cimetrics) FreeRange + VSB (PolarSoft). The BACnet Linux Server (SCADA Engine) BACnet Stack (Chipkin) BACnet Stack (MBS) BACnet Stack (SoftDEL Systems) BACnet4J (Infinite Automation Systems)

BACnet Protocol Stack (in C)

This BACnet protocol stack library provides a BACnet application layer, network layer and media access (MAC) layer communications services.

License specifically designed for the embedded BACnet appliance, using a GPL with exception for linking license (like eCos).

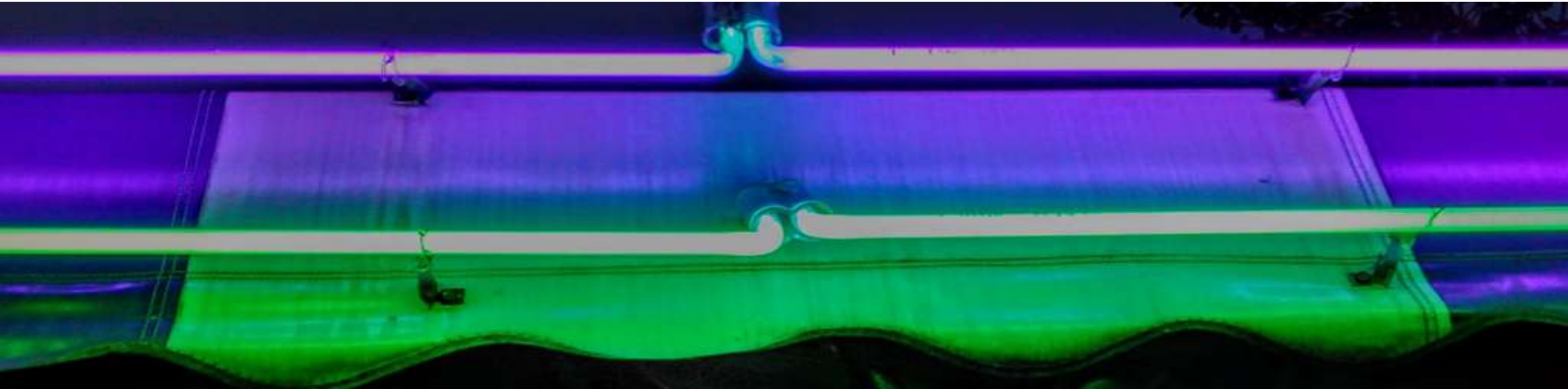
The source code is written in C for portability, and includes unit tests (PC based unit tests) and example application code. Since the code is designed to be portable, it compiles with GCC as well as other compilers, such as Borland C++, Visual C++, MinGW, Code Warrior, IAR, or MicroChip C18. The source code is also designed to be high quality, readable, understandable, and easy to use.

Code Compilers

GCC: GNU Compiler Collection C, C++, Objective-C, Fortran, Java, Ada, and Go

Clang: C, C++, Objective C and Objective C++ front-end for the LLVM compiler

Proprietary: IAR, Rowley, MCC, Hi-Tech, Microsoft, Borland, others



Wireshark - Capture Utility

MS/TP Capture

mstpcap utility, some BACnet routers, U+4, mstpsnap utility, extcap

BACnet/IP capture

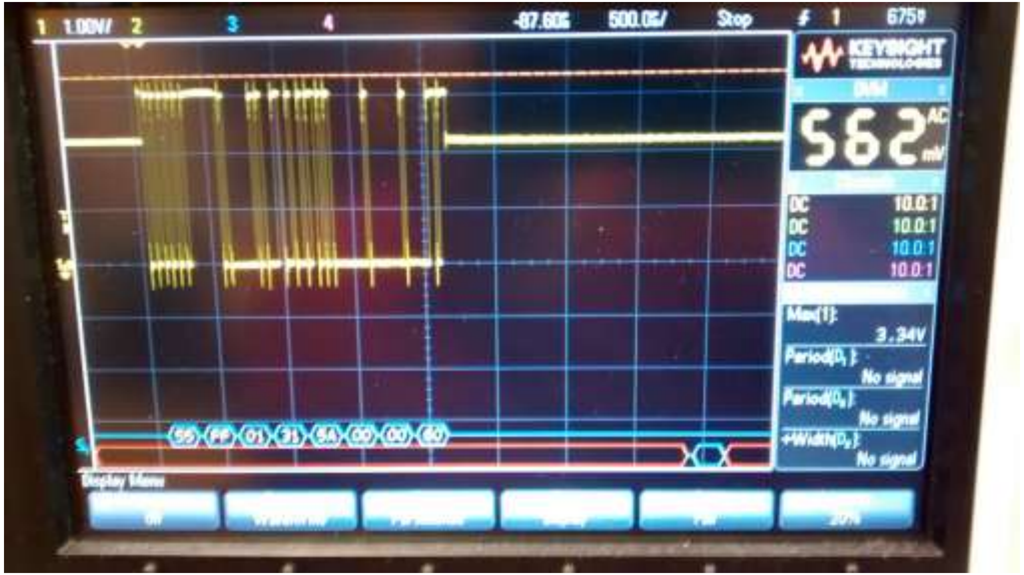
BACnet Ethernet capture

BACnet ARCNET capture

Other BACnet Capture Tools:

BAS-O-Matic (Cimetrics) or BACtrace & BACtrack (Polarsoft)

BACnet Testing Tools



VTS (SourceForge)

BACnet Quick Test (Polarsoft)

BAC-Test (Connect-Ex)

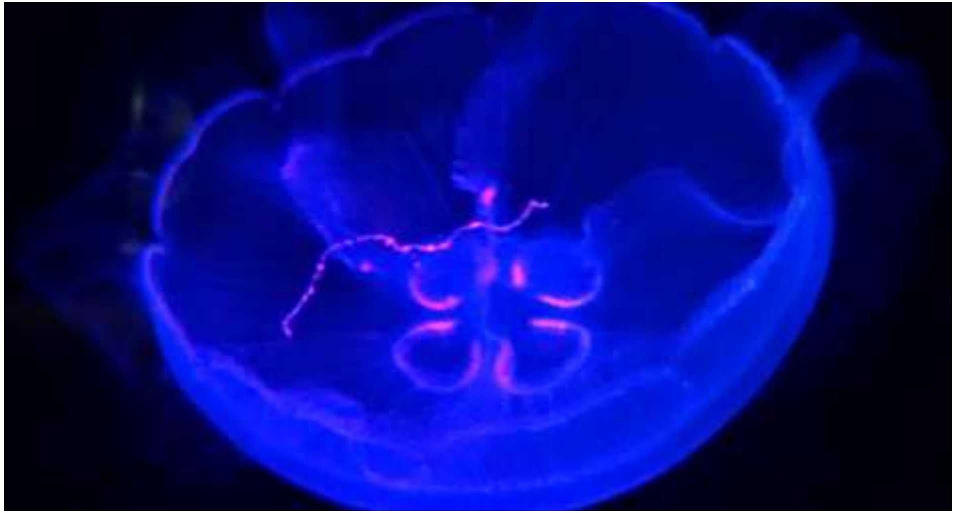
BTF (MBS)

BACnet Tools (demo applications with BACnet Stack)

BACnet Testing Tools - Scripts

Using BACnet Tools (demo from bacnet.sourceforge.net project)

```
@echo OFF
rem Script to ReadProperty FIRMWARE_REVISION value from a Device.
rem The script rem uses a text file with a Device ID on each line,
rem which can be copied from the bacwi command and pasted into a file.
rem The script increments the Invoke ID to prevent ack collisions.

if "%~1"=="" goto failparse
SETLOCAL EnableDelayedExpansion

set BACNET_APDU_TIMEOUT=1000
set BACNET_INVOKE_ID=1

for /F "tokens=*" %%A in (%~1) do (
echo Device %%A Version:
START /B /WAIT bacrp %%A 8 %%A 44 2>nul
if %ERRORLEVEL% GEQ 1 echo FAILED:Readproperty from Device %%A
call :increment
)

goto end
:increment
set /A BACNET_INVOKE_ID=%BACNET_INVOKE_ID%+1
if %BACNET_INVOKE_ID% gtr 255 set BACNET_INVOKE_ID=1
exit /b 0

goto end
:failparse
echo Needs the file name with Device IDs on each line
goto end
:end
```



BACnet Testing Tools - Scripts

Using BACnet Tools (demo from bacnet.sourceforge.net project)

```
rem 9.24.1.3 Finite Time Duration
rem Purpose: To verify the correct execution of the
rem DeviceCommunicationControl request service procedure
rem when finite time duration is specified.
rem Test Steps:
rem 1. TRANSMIT DeviceCommunicationControl-Request,
rem 'Time Duration' = (a value T > 1, in minutes, selected by the tester),
set DCC_TIME_DURATION=5
rem 'Enable/Disable' = DISABLE,
rem COMMUNICATION_ENABLE = 0,
rem COMMUNICATION_DISABLE = 1,
rem COMMUNICATION_DISABLE_INITIATION = 2,
set DCC_STATE=1
rem 'Password' = (any appropriate password as described in the Test Concept)
set DCC_PASSWORD=filister
rem TRANSMIT, and
rem 2. RECEIVE BACnet-Simple-ACK-PDU
bacdcc %1 %DCC_STATE% %DCC_TIME_DURATION% %DCC_PASSWORD%
rem 3. WAIT Internal Processing Fail Time
TIMEOUT /T 5
set READ_CYCLES=6
@FOR /L %%x in (0,1,%READ_CYCLES%) DO (
rem 4. TRANSMIT ReadProperty-Request,
rem 'Object Identifier' = (Device, X),
rem 'Property Identifier' = (any required non-array property of the Device object)
bacrp %1 8 %1 75
rem 5. WAIT (T )
TIMEOUT /T 60
rem 6. CHECK (Verify that the IUT did not transmit any messages between the acknowledgment in step 2 and expiration oftimer T)
rem 7. VERIFY (Device, X),
rem (any required non-array property) = (the value for this property specified in the EPICS)
)
```