EVT-300 Radar Communication ‘C’ DLL
Programmer’s Reference

Source and Linking Requirements

EVT.h

This header file defines the interface to the DLL. It must be included where the Radar routines are used in your source code.

EVT.lib

This library file is required for compile-time linking. In Visual C++ it needs to be added under “Object/library modules” in the “Link” tab of the Project Settings.

EVT.dll

EVT.dll is the actual dynamically linked library. It must be available to your program at run-time. The DLL is not Self-registering; it must reside in either the same directory as your program, somewhere in your system’s path, or in a directory specified in your PATH environment variable.

Interface Description

Functions

DWORD _stdcall Initialize(LPCSTR Device, STATUSPROC * StatProc)

Initialize() opens the supplied Device and begins communications with the Radar Unit. This function also allows you to define a callback function for status messages.

Example:

CString c("COM1");  // Any legal device name

if (!Initialize(c, NULL)) {
    // Initialization was successful
    ... ...
} else {
    // Initialization failed
    ... ...
}

Note: Supplying NULL for StatProc will cause your program to not receive Status Messages. At this time Status Messages are limited to Communications Errors.

Return Value(s):

If successful, Initialize() returns 0. The Radar unit should now be ready to report target data. If an error occurred, Initialize() returns a Windows error code as described in WINERROR.H.
void _stdcall UnInitialize()

This method abandons communications with the Radar Unit and closes the Communications Device.

Example:

    Uninitialize();

Return Value(s):

    This function does not return a value.

void _stdcall NotifyTRM(TRMPROC * TRM)

Sets up a callback function for the reporting of Target Report Messages. If the function pointer supplied is NULL or invalid, your application will not receive Target Report Messages. See the definition of TRMPROC below for details of the callback.

Return Value(s):

    This function does not return a value.

void _stdcall NotifyDT(DTPROC * DT)

Sets up a callback function for the reporting of Dropped Target Messages. If the function pointer supplied is NULL or invalid, your application will not receive Dropped Target Messages. See the definition of TRMPROC below for details of the callback.

Return Value(s):

    This function does not return a value.

DWORD _stdcall GoodPktCount()

Supplies the number of good data packets received from the Radar unit since the DLL was loaded. This includes non-relevant data packets.

DWORD _stdcall BadPktCount()

Supplies the number of bad data packets received from the Radar unit since the DLL was loaded. These packets are discarded.
Callback Definitions

`long CALLBACK TRMPROC(int TargetID, double Range, double Rate, double Azimuth, long TimeStamp)`

Reports one set of active target data.

  - TargetID - The value that distinctly identifies a target. This value has a range of 1 – 255.
  - Rate - The target's velocity in MPH. Precision: 0.1 ft/sec (signed)
  - Range - The target’s range in ft. Precision: 0.1 ft Range: 0 – 6553.5 ft
  - Azimuth - The target's azimuth in radians. Precision: 0.002 radians (signed)
  - TimeStamp - Time Stamp of when the data was received.

`VOID CALLBACK DTPROC(int TargetID, long TimeStamp)`

Reports that a target has been dropped by the Radar unit.

  * See TRMPROC for description of parameters.

`VOID CALLBACK STATUSPROC(DWORD msg)`

Reports a communication event

  - msg: A Windows error code number defined in WINERROR.H describing an error that occurred during communications.