



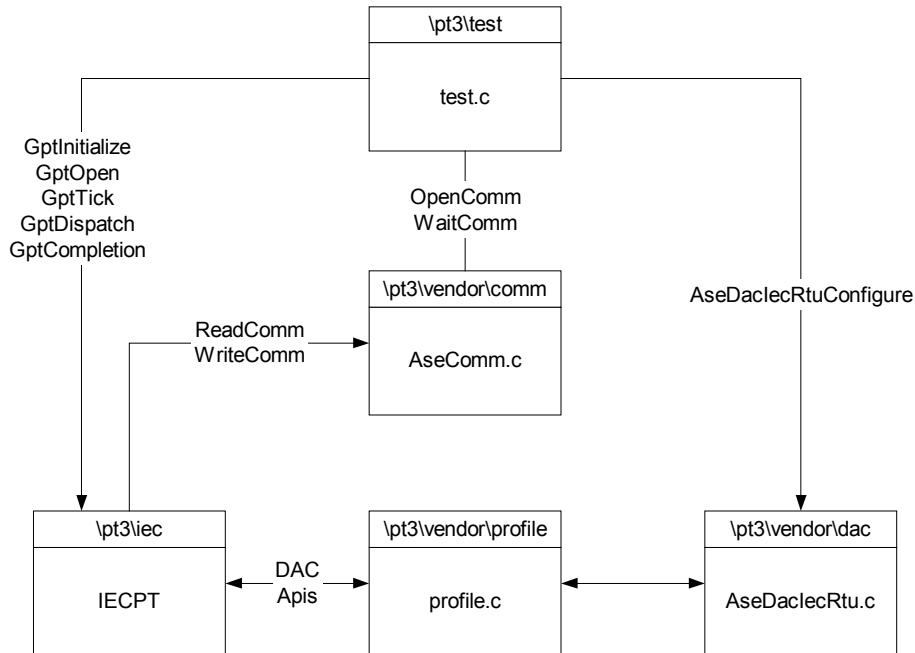
Applied Systems Engineering, Inc.

Technical Note #43
IEC Example Code

About IEC Sample Code

The general release of the GPT provides sample software that can be used to test any of the protocols supported by the GPT. This document describes a simplified version of the example software that only supports the IEC870-5-101 protocol. This simplified software supports a single protocol IEC870-5-101. The simple IEC implementation reduces the amount of code the user must review before beginning a port to the users hardware. The following figure illustrates the components of the simple IEC.

Figure 1. IEC Sample Code Overview



Components of the Simple IEC release include:

- **Test.c.** Main program running on a WIN32 platform. The user will replace test.c with a version the operates on the user's target hardware platform. The main program calls IECPT, COMM and DAC APIs to initialize the test environment. Once the test environment is initialized the main program calls GPT, COMM and DAC APIs to simulate changes in the process and report those changes to a remote master. Routines called out of the main() program are:
 - **GptInitialize.** Initialize the IECPT and place all of its properties into a default state.
 - **GptOpen.** Start the IECPT.
 - **GptTick.** Deliver a timer interrupt to the GPT.
 - **GptDispatch.** Call the GPT to process any pending events (communication timeouts, background polling).
 - **GptCompletion.** Deliver to the GPT the COMM completion events read complete (EVT_READ_COMPLETE) or write complete (EVT_WRITE_COMPLETE).

TechNote 43: IEC Sample Code

- **OpenComm.** Initialize and open the communications component.
- **WaitComm.** Check for read or write completions.
- **AseDacIecRtuConfigure.** Creates a test database of process variables.
- **IECPT.** IEC870-5-101 library. The library code is defined by the directories \pt3\iec, \pt3\gpt, and \pt3\include. The user should not have to modify the library code. The library code should directly port to the user's target hardware platform as long as the target supports and ANSI-C compiler.
- **AseComm.c** Communication APIs running on a WIN32 platform. The user will replace AseComm.c with a component that supports the user's communication hardware.
- **Profile.c.** IEC870-5-101 object profile.
- **AseDacIecRtu.c.** ASE implementation of IEC DAC APIs. The user will replace this implementation with a version that supports access to the user's process data.

Simple IEC builds on a WIN32 or DOS platform. The user must replace AseComm.c and AseDacIecRtu.c with implementations that work with the user's hardware (refer to the section in README.DOC that describes porting the GPT to another hardware platform).

AseDacIecRtu.c

The DAC device profile for a protocol is defined in \vendor\profiles\profile.c. The device profile defines table entries for all objects supported by a protocol. Each profile entry provides methods (procedure pointers) to user-written code that supports read/write access to an objects data. The module AseDacIecRtu.c initializes these method pointers when the entry point AseDacIecRtuConfigure is called. All DAC data and methods are defined statically in the module AseDacIecRtu.c. This module shares no code or data with the other components in the GPT release. The user can replace AseDacIecRtu.c with another implementation e.g. MyDacIecRtu.c without having to change other components of the GPT. The same is true for AseComm.c. AseComm.c is self-contained. The user can replace AseComm.c with a new implementation MyComm.c that supports the user's hardware.