PROFESSIONAL CONTROL SOLUTIONS, LLC

NetLinx Module Interface Specification

for a

Phonebook

TABLE OF CONTENTS

Revision History

Date	Initials	Comments
5-17-09	DC	Initial release
11-12-12	DC	Added confirmation for phonebook entry delete

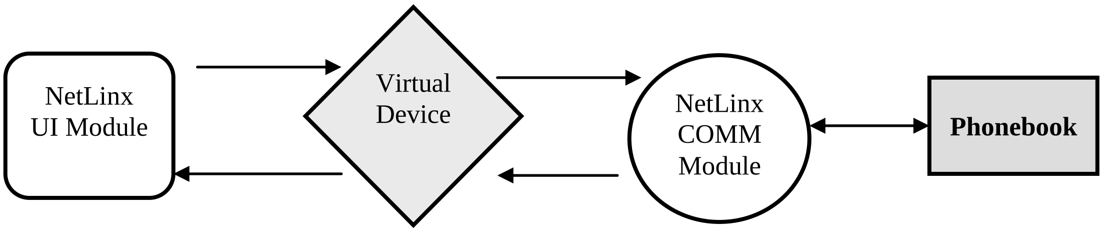
Introduction

This is a reference manual to describe the interface provided between an AMX NetLinx system and an internal phonebook. The phonebook supports 256 names (32 bytes) and numbers (20 bytes) without modification. Add, Change, Delete, and Dial functions are provided as well as an internal sort.

Overview

The COMM module translates between the standard interface described below and the **phonebook** application.

The following diagram gives a graphical view of the interface between the interface code and the NetLinx module.



Implementation

To interface to the PHONEBOOK module, the programmer must perform the following steps:

1. Define the virtual device ID that the PHONEBOOK_COMM module will use to communicate between the main program and comm. module. NetLinx virtual devices start with device number 33001.
2. If a touch panel interface is desired, a touch panel file PHONEBOOK.tp4 has been created for testing and including the pages into your own file.
3. The NetLinx PHONEBOOK module must be included in the program with a DEFINE_MODULE command. This command starts execution of the module and passes in the following key information the virtual device ID for communicating to the main program, the base touch panel device (such as 10001:1:0), the touch panel device containing the phonebook pages (sample set for 10001:3:0), a name of a file to contain a backup of the database (sample – PHONEBOOK.TXT), and a header to pass to the module for display purposes (sample- VTC Phonebook).

An example of how to do this is shown below.

```
 dvMainTP = 10001:1:0  // The touch panel used to accept input from the “keyboard” (must be port 1).
dvPhonebookTP = 10001:3:0  // The touch panel that contains the phonebook pages.
 vdvPHONEBOOK = 33001:1:0       // The virtual device use for communication between the
                                          // Comm module interface and User_Interface (UI) module interface

DEFINE_VARIABLE
CHAR strFILE[]=’PHONEBOOK.TXT’
CHAR strHEADER[]=’VTC Phonebook’
INTEGER nINSTANCE

DEFINE_START       // Place define_module calls to the very end of the define_start section.
// Comm module
DEFINE_MODULE PHONEBOOK_Comm' mdlPHONEBOOK_APP(vdvPHONEBOOK, dvMainTP,
dvPHONEBOOKTP, strFILE[], strHEADER[], nINSTANCE)
```

Buttons

The buttons supported by the COMM module are listed below. These buttons are associated with the with the touch panel device for the particular instance.

Note: # in the descriptions below indicates a specific instance number.

Button	Description
3	Dial phonebook entry. Returns the “DIAL-number” string described in table 3.
4	Add phonebook entry. Also shows “Phonebook Entry #’ popup.
5	Change phonebook entry. Also shows “Phonebook Entry #’ popup.
6	Delete phonebook entry. Also hides “Phonebook Entry #’ popup.
7	Previous Page.
8	Next Page.
9	Variable text for page number.
10	Hide “Phonebook Delete #” confirmation popup.
11	Show “Phonebook Delete #” confirmation popup.

Table 1 - Button Definitions

Channels

The channels supported by the COMM module are listed below. These channels are associated with the virtual device(s) and are independent of the channels associated with the touch panel device.

Note: An ‘*’ indicates an extension to the standard API.

Channel	Description
1	Pulse this channel to force a load from the backup file.
2	Pulse this channel to force a save of the backup file.
3	Pulse this channel to force a display of page 1 of the phonebook.

4	Force a resort of the phonebook.

Table 2 - Channel Definitions

Note: pulsing channels 1 and 4 also will force a redisplay of page 1 of the phonebook.

String Feedback

Interaction with the module is basically performed by the touch panel, with the addition of the channel events. When the end-user presses the dial button, will return the following string. There are no commands that can be currently sent to the module.

String	Description
DIAL-<value>	Indicates that the user wishes to dial the selected number. <value>: Any valid phone/IP number for dialing DIAL-18005551212

Table 3 - String Feedback Definitions

Device Notes

The module will currently attempt to read the database file on startup. There is no automatic save or reload. After presenting the Phonebook screen, the calling program will need to display the database data using a pulse of channel 3. This is a simple phonebook app. It is possible to corrupt the database. The database file is a simple text file that you can FTP to the master. The format is “name”, delimiter, “number”, delimiter, CR/LF. The delimiter is the shifted forward slash (|) key.

Programming Notes

The COMM module is currently set for a maximum of 256 database entries. This can be changed, but will have an impact on memory use. Button numbers are fixed, and it assumed that the programmer will copy the touch panel screens into their own file. The program assumes that there is one panel for interaction. Multiple phonebook instances may be used assuming that the port number on the touch panel screens is kept separate for each instance. The program does provide rudimentary error checking to make sure that data is entered into both the name and number fields. This is an attempt prevent blank entries in the database, which will be at the top of the sort.

IMPORTANT NOTE: To provide for multiple instances of the phonebook application, the port number of the phonebook-specific buttons will need to be changed for the specific instance used. The sample TP4 file uses port 3 for the Phonebook buttons. The SAVE button on the Phonebook Entry screen (the only popup – this is on the keyboard) is set to use the setup port (0) for the channel port. It is important that the COMMAND PORT for this button be set to match the port used by the other Phonebook buttons (in this case port 3). When the SAVE button is pressed, a command of “SAVE PRESSED” is sent to the TP port assigned to the Phonebook buttons. This is required because the strings entered in the Name and Number fields is sent back via the base device port (port 1). The command sent with the press of the SAVE button notifies the module that the string text arriving for the phonebook application is being directed to this specific instance of the module.

The page name for the main Phonebook page is not important to the module, since this page will be controlled by the main program. However, the popup page needs to have the instance number as part of the name. The sample TP4 file contains the popup page named Phonebook Entry 1 for the data-entry popup page. If using multiple copies, the name should match the instance number.

Adding Functions to Modules

Source code is provided so that the module can be modified for a particular installation.