PUMPKIN™
REAL-TIME SOFTWARE
750 Naples Street • San Francisco, CA 94112 • (415) 584-6360 • http://www.pumpkininc.com

AN-18
Application Note

Building a Salvo Application with Quadravox's AQ430 Development Tools

Introduction

This Application Note explains how to use Quadravox's (http://www.quadravox.com/) AQ430 Development Tools to create a multitasking Salvo application for Texas Instruments' (http://www.ti.com/) MSP430 ultra-low-power microcontrollers.

We will show you how to build the Salvo application contained in `\salvo\ex\ex1\main.c` for an MSP430F149 using the AQ430 Development Tools. For more information on how to write a Salvo application, please see the *Salvo User Manual*.

Before You Begin

If you have not already done so, download and install the AQ430 Development Tools.[1] You will also need an MSP430 Flash Emulation Tool (FET) for debugging. More information is available at http://www.ti.com/sc/msp430.

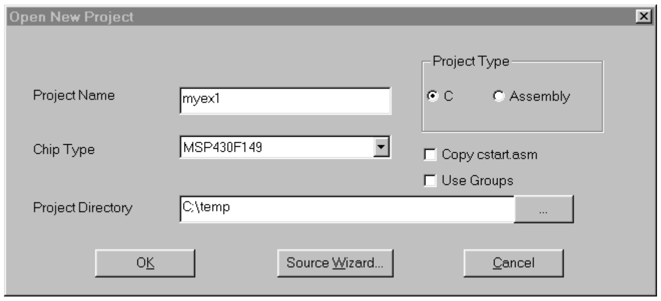
Related Documents

The following Salvo documents should be used in conjunction with this manual when building Salvo applications with Archelon / Quadravox's AQ430 C compiler:

Salvo User Manual
Salvo Compiler Reference Manual RM-AQ430

Creating and Configuring a New Project

Create a new AQ430 project using Project → New. Under Project Name enter a name for the project (we'll use `myex1`), select MSP430F149 as the Chip Type, and navigate to your working directory (in this case we've chosen `c:\temp`) for Project Directory: [2]


Open New Project

Project Name: myex1

Chip Type: MSP430F149

Project Directory: C:\temp

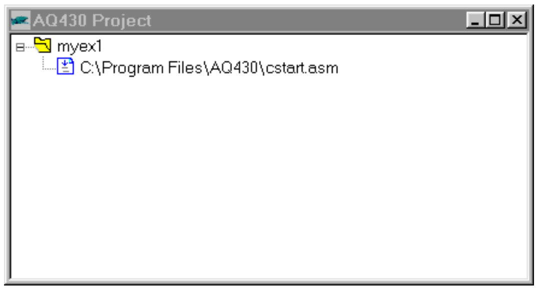
Project Type: [✓ C] Assembly

Copy cstart.asm: unchecked

Use Groups: unchecked

Figure 1: Creating the New Project

Click OK to continue. The AQ430 Project window appears:

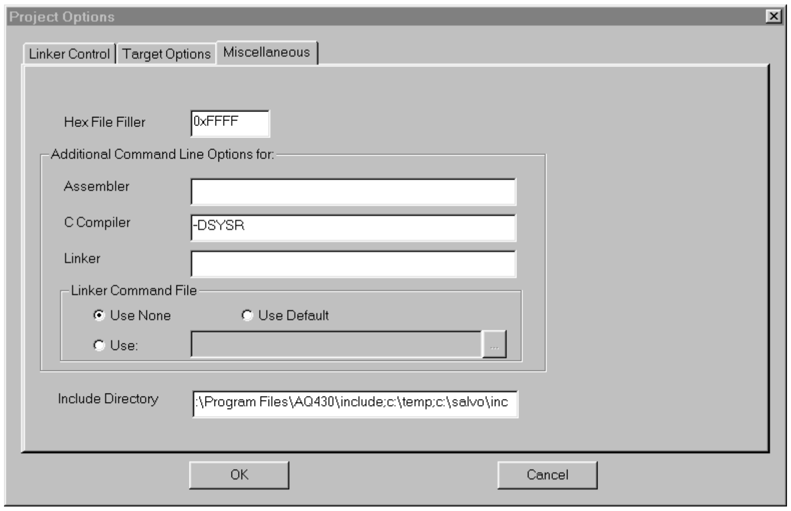

AQ430 Project

myex1
C:\Program Files\AQ430\cstart.asm

Figure 2: AQ430 Project Window

Choose Project → Save to save the project.

Now let's setup the project's options for Salvo's pathnames, etc. Select Options → Project → Miscellaneous and define any symbols you may need for your project in the Additional Command Line Options for area.[3] In the Include Directory, add the project's own include path and `\salvo\inc\`, separated by semicolons:


Project Options

Linker Control | Target Options | Miscellaneous

Hex File Filler: 0xFFFF

Additional Command Line Options for:

Assembler:

C Compiler: -DSYSR

Linker:

Linker Command File: [✓ Use None] Use Default Use:

Include Directory: :\Program Files\AQ430\include;c:\temp;c:\salvo\inc

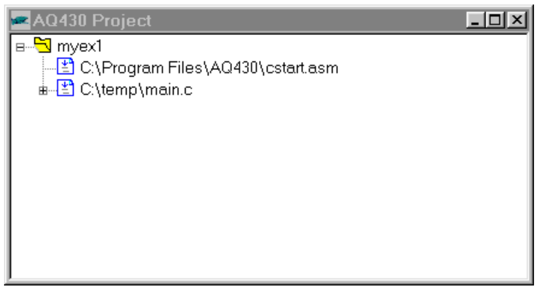
OK Cancel

Figure 3: AQ430 Project Options

Select OK to finish configuring your project.

Adding your Source File(s) to the Project

Now it's time to add files to your project. Choose Project → Add Files, navigate to your project's directory, select your `main.c` and Open. Your AQ430 Project window should now look like this:


AQ430 Project

```
myex1
  C:\Program Files\AQ430\cstart.asm
  C:\temp\main.c
```

Figure 4: AQ430 Project Window with your Source File(s)

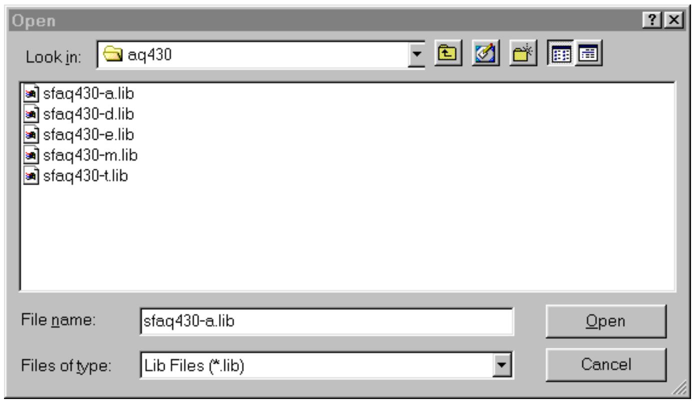
Adding Salvo-specific Files to the Project

Now it's time to add the Salvo files your project needs. Salvo applications can be built by linking to precompiled Salvo libraries, or with the Salvo source code files as nodes in your project.

Adding a Library

For a *library build*, a fully-featured Salvo freeware library for the AQ430 Development Tools is `sfaq430-a.lib`.[4] Select Project →

Add Files, choose Lib Files (*.lib) under Files of type, navigate to the `\salvo\lib\aq430` directory, and select `sfaq430-a.lib`:



sfaq430-a.lib
sfaq430-d.lib
sfaq430-e.lib
sfaq430-m.lib
sfaq430-t.lib

File name: sfaq430-a.lib
Files of type: Lib Files (*.lib)

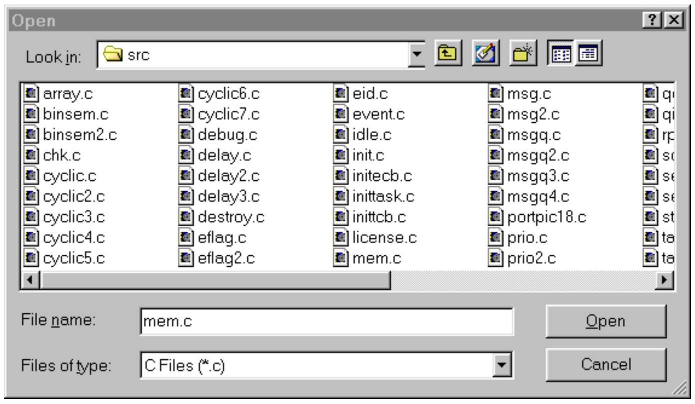
Figure 5: Adding the Library to the Project

Select Open when you are finished. You can find more information on Salvo libraries in the *Salvo User Manual* and in the *Salvo Compiler Reference Manual RM-AQ430*.

Tip You can rearrange the files in the AQ430 Project window by clicking and dragging on the file(s) you want to move up or down.

Adding Salvo's mem.c

Salvo library builds also require Salvo's `mem.c` source file as part of each project. Choose Project → Add Files, navigate to `\salvo\src`, select `mem.c` and Open.

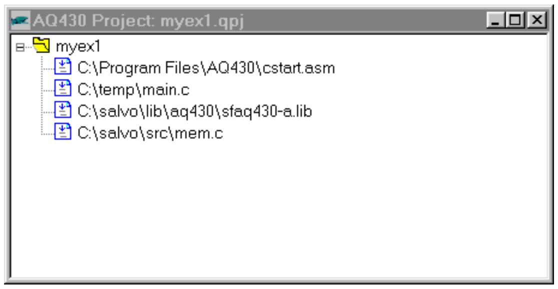


array.c, binsem.c, binsem2.c, chk.c, cyclic.c, cyclic2.c, cyclic3.c, cyclic4.c, cyclic5.c, cyclic6.c, cyclic7.c, debug.c, delay.c, delay2.c, delay3.c, destroy.c, eflag.c, eflag2.c, eid.c, event.c, idle.c, init.c, initecb.c, inittask.c, inittcb.c, license.c, mem.c, msg.c, msg2.c, msgq.c, msgq2.c, msgq3.c, msgq4.c, portpic18.c, prio.c, prio2.c

File name: mem.c
Files of type: C Files (*.c)

Figure 6: Adding Salvo's mem.c to the Project

Your AQ430 Project window should now look like this:


AQ430 Project: myex1.qpj

```
myex1
  C:\Program Files\AQ430\cstart.asm
  C:\temp\main.c
  C:\salvo\lib\aq430\sfaq430-a.lib
  C:\salvo\src\mem.c
```

Figure 7: AQ430 Project Window for Library Build

The salvocfg.h Header File

You will also need a `salvocfg.h` file for this project. A working `salvocfg.h` for the library selected in Figure 5 must contain at a minimum:

```
#define OSUSE_LIBRARY          TRUE
#define OSLIBRARY_TYPE         OSF
#define OSLIBRARY_CONFIG       OSA
```

Listing 1: salvocfg.h for a Library Build

Create this file and save it in your project directory, e.g. `c:\temp\salvocfg.h`.

Proceed to *Building the Project*, below.

Adding Salvo Source Files

If you have a Salvo distribution that contains source files, you can do a *source code build* instead of a library build. The application in `\salvo\ex\ex1\main.c` contains calls to the following Salvo user services:

```
OS_Delay()              OSInit()
OS_WaitBinSem()         OSSignalBinSem()
OSCreateBinSem()        OSSched()
OSCreateTask()          OSTimer()
OSEi()
```

You must add the Salvo source files that contain these user services, as well as those that contain internal Salvo services, to your project. The *Reference* chapter of the *Salvo User Manual* lists the source file for each user service. Internal services are in other Salvo source files. For this project, the complete list is:

```
binsem.c                mem.c
```

```
delay.c                 portaq430.asm
event.c                 qins.c
idle.c                  sched.c
init.c                  timer.c
inittask.c
```

To add these files to your project, select Project → Add Files, select C Files (*.c) under Files of type, navigate to the `\salvo\src` directory, and select[5] the `*.c` files listed above:

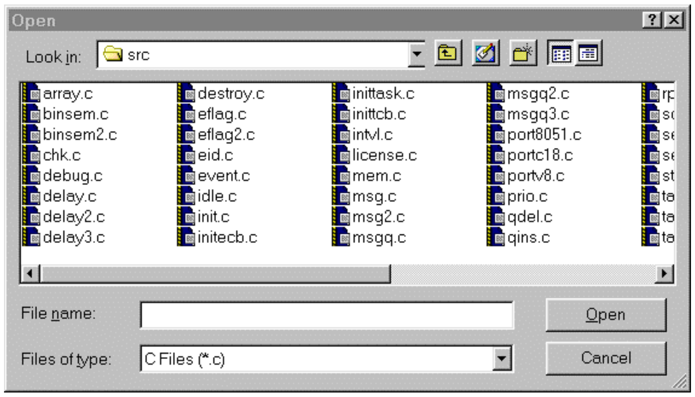
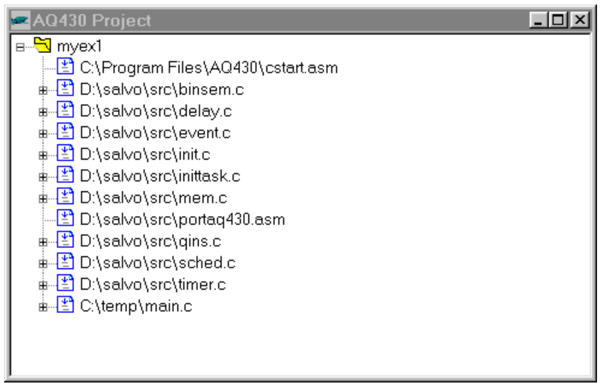


Figure 8: Adding Salvo Source Files to the Project

Select Open when finished. Repeat using Source Files (*.asm) under Files of type in order to add `\salvo\src\portaq430.asm` to your project. Your AQ430 Project window should now look like this:


AQ430 Project

```
myex1
    C:\Program Files\AQ430\cstart.asm
    D:\salvo\src\binsem.c
    D:\salvo\src\delay.c
    D:\salvo\src\event.c
    D:\salvo\src\init.c
    D:\salvo\src\inittask.c
    D:\salvo\src\mem.c
    D:\salvo\src\portaq430.asm
    D:\salvo\src\qins.c
    D:\salvo\src\sched.c
    D:\salvo\src\timer.c
    C:\temp\main.c
```

Figure 9: Project Window for a Source Code Build

The salvocfg.h Header File

You will also need a `salvocfg.h` file for this project. Configuration files for source code builds are quite different from those for library builds (see Listing 1, above). For a source code build, the `salvocfg.h` for this project contains at a minimum:

```
#define OSBYTES_OF_DELAYS           1
#define OSENABLE_IDLING_HOOK        TRUE
#define OSENABLE_BINARY_SEMAPHORES  TRUE
#define OSEVENTS                    1
#define OSTASKS                     3
```

Listing 2: salvocfg.h for a Source Code Build

Create this file and save it in your project directory, e.g. `c:\temp\salvocfg.h`.

Building the Project

For a successful compile, your project must also include a header file (e.g. `#include <msp430x14x.h>`) for the particular chip you are using. Normally, this is included in each of your source files (e.g. `main.c`), or in a header file that's included in each of your source files (e.g. `main.h`).

With everything in place, you can now build the project using Build → Make or Build → Rebuild All. The build results can be seen in the AQ430 User Information window:

```
Building Project : [myex1.qpj]
Assembling C:\Program Files\AQ430\cstart.asm
Pass 1: 0 errors, Pass 2: 0 errors, 0 warnings
Compiling D:\salvo\src\binsem.c
0 errors, 0 warnings
Assembling D:\salvo\src\binsem.asm
Pass 1: 0 errors, Pass 2: 0 errors, 0 warnings
…
Compiling D:\salvo\src\timer.c
0 errors, 0 warnings
Assembling D:\salvo\src\timer.asm
Pass 1: 0 errors, Pass 2: 0 errors, 0 warnings
Compiling C:\temp\main.c
0 errors, 1 warnings
Assembling C:\temp\main.asm
Pass 1: 0 errors, Pass 2: 0 errors, 0 warnings
Linking C:\temp\myex1.rxc
0 errors
Erasing Info Memory
Done
Erasing Main Memory
Done
Programming Main Memory
1396 bytes written to flash memory
First free RAM location: 0x022E
Done
Building symbol tables...
Done
```

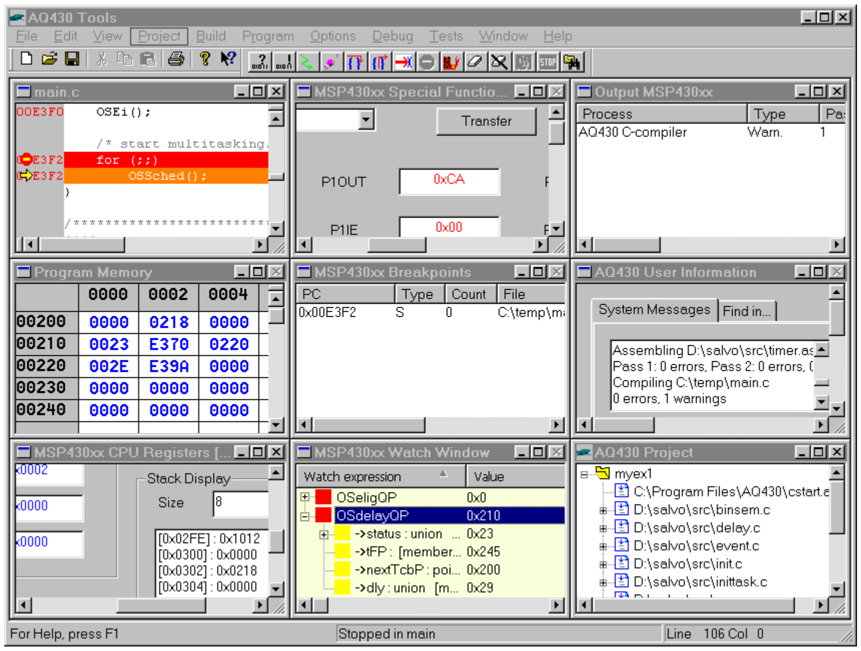
Listing 3: Source Code Build Results (Abbreviated)

Note The projects supplied in the Salvo for TI's MSP430 distributions contain additional help files in the project's Salvo Help Files group.

Testing the Application

You can test and debug this application using the Flash Emulation Tool. The AQ430 debugger launches automatically following a successful make or build.

You can use all of the IDE's supported features when debugging and testing Salvo applications. This includes breakpoints, profiling, watch windows, tracing, etc.



```
00E3F0      OSEi();

            /* start multitasking.
 E3F2       for (;;)
 E3F2           OSSched();
}

/************************
```

P1OUT	0xCA
P1IE	0x00

Process	Type	Pa
AQ430 C-compiler	Warn.	1

	0000	0002	0004
00200	0000	0218	0000
00210	0023	E370	0220
00220	002E	E39A	0000
00230	0000	0000	0000
00240	0000	0000	0000

PC	Type	Count	File
0x00E3F2	S	0	C:\temp\m

```
Assembling D:\salvo\src\timer.as
Pass 1: 0 errors, Pass 2: 0 errors, (
Compiling C:\temp\main.c
0 errors, 1 warnings
```

```
x0002
x0000
x0000
Size 8
[0x02FE] : 0x1012
[0x0300] : 0x0000
[0x0302] : 0x0218
[0x0304] : 0x0000
```

Watch expression	Value
OSeligQP	0x0
OSdelayQP	0x210
->status : union ...	0x23
->tFP : [member...	0x245
->nextTcbP : poi...	0x200
->dly : union [m...	0x29

```
myex1
  C:\Program Files\AQ430\cstart.a
  D:\salvo\src\binsem.c
  D:\salvo\src\delay.c
  D:\salvo\src\event.c
  D:\salvo\src\init.c
  D:\salvo\src\inittask.c
```

For Help, press F1 | Stopped in main | Line 106 Col 0

Figure 10: Testing a Salvo Application in AQ430 Development Tools

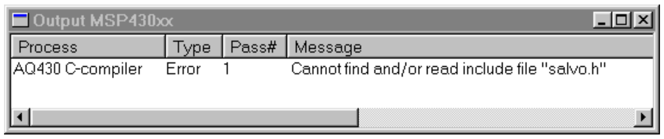
---

Note The AQ430 Development Tools supports debugging at the source code level. Only applications built from the Salvo source code enable you to step through Salvo services (e.g. `OSCreateBinSem()`) at the source code level. Regardless of how you build your Salvo application, you can always step through your own C and assembly code in the IDE / debugger.

Troubleshooting

AQ430 C-compiler Error: Cannot find and/or read include file(s)

If you fail to add `\salvo\inc` to the project's include paths (see Figure 3) the compiler will generate an error like this one:

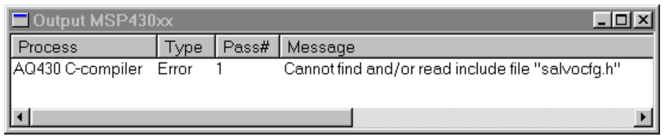

Output MSP430xx

Process	Type	Pass#	Message
AQ430 C-compiler	Error	1	Cannot find and/or read include file "salvo.h"

Figure 11: Compiler Error due to Missing \salvo\inc Include Path

By adding `\salvo\inc` to the project's include path, you enable the compiler to find the main Salvo header file `salvo.h`, as well as other included Salvo header files.

If you fail to add the project's own directory to the project's include paths (see Figure 3) the compiler will generate an error like this one:


Output MSP430xx

Process	Type	Pass#	Message
AQ430 C-compiler	Error	1	Cannot find and/or read include file "salvocfg.h"

Figure 12: Compiler Error due to Missing Project Include Path

By adding the project's own directory to the project's include path, you enable the compiler to find the project-specific header file `salvocfg.h`.

Cannot Find Project Window Upon Opening Project

If you can't see the AQ430 Project window (or any other window) after opening an AQ430 project that's part of a Salvo distribution, it may be because your display's resolution is less than that used to create the project. Select Window → Tile to make all open windows visible.

Build Error

An error that looks like this when you make or rebuild your project:

```
Building Project : [ex1lite.qpj]
Compiling C:\salvo\src\mem.c
-1 errors, 0 warnings
Archelon URCC C 3.16 2002/09/30
aq430c: aq430pp: No such file or directory
aq430pp error return -1 - no compilation
```

is most likely because the startup file `cstart.asm` is not the first file in the AQ430 Project window, and your computer's `PATH` environment variable does not contain the AQ430 install directory (usually `c:\Program Files\AQ430`). The solution is to add

```
SET PATH=%PATH%;c:\Program Files\AQ430
```

to your computer's `autoexec.bat` file and reboot.

Hardware Check (MSP430 FET) Fails

Be sure to set[6] the mode of your computer's parallel (LPT) port to ECP (instead of bi-directional, SPP or EPP), or else Options → Hardware Check is likely to fail.

Freezes After Opening Project

Opening a source-code-build project in a Salvo Pro distribution may cause the AQ430 IDE to freeze momentarily while it analyzes the project's dependencies. Simply wait for it to finish. When ready, the AQ430 Project window will be visible.

Application Crashes After Changing Processor Type

Remember to `#include` the appropriate header file for your MSP430 variant (see *Building the Project*, above). While the common SFR locations are consistent across the entire MSP430 family, the interrupt vectors are not. Therefore mainline code may work correctly, but the application will crash if interrupt vectors are not in the right locations.

No Source-Code-Level Debugging in Salvo Pro Libraries

In order to step through Salvo library services at the source-code / C level, Salvo Pro users must:

- Select a Salvo library with embedded debugging information (library option is set to `i`), e.g. `slaq430ia.lib`, and
- Under Options → Project → Miscellaneous, select Modify File Names in Object Files

When you make or (re-)build the project, if the AQ430 debugger cannot locate the source file that is referenced from the library's object files, you will be prompted to supply a path name:

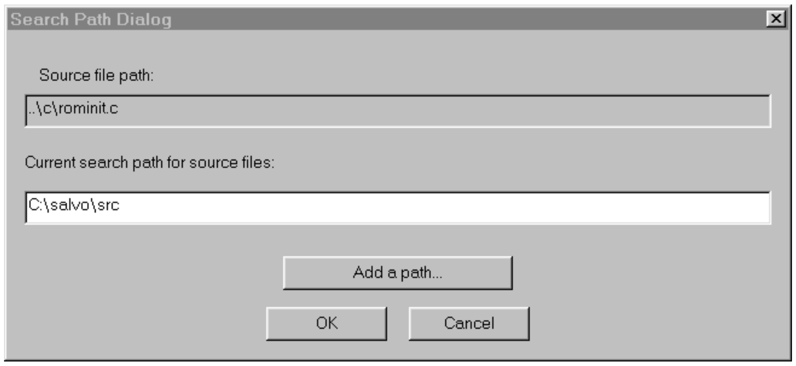


Figure 13: Search Path Dialog Window

Add the Salvo source directory (usually `\salvo\src`) to the path by editing the Current search path for source files or by clicking on Add a path … and navigating to that directory.

Tip The Search Path Dialog window will prompt you for a search path whenever it cannot find the source file for a particular object. In Figure 13, it is looking for the source file associated with an AQ430 library function, for which source is not provided. Just click on the Search Path Dialog window's OK button and that file will be effectively skipped.

Example Projects

Example projects for the AQ430 Development Tools are found in the `\salvo\tut\tu1-6\sysr` directories. The include path for each of these projects includes `\salvo\tut\tu1\sysr`, and each project defines the `SYSR` symbol.

Complete projects using Salvo freeware libraries are contained in the project files `\salvo\tut\tu1-6\sysr\tu1-6lite.qpj`. These projects also define the `MAKE_WITH_FREE_LIB` symbol.

Complete projects using Salvo standard libraries are contained in the project files `\salvo\tut\tu1-6\sysr\tu1-6le.qpj`. These projects also define the `MAKE_WITH_STD_LIB` symbol.

Complete projects using Salvo standard libraries with embedded debugging information are contained in the project files `\salvo\tut\tu1-6\sysr\tu1-6prolib.qpj`. These projects also define the `MAKE_WITH_STD_LIB` symbol.

Complete projects using Salvo source code are contained in the project files `\salvo\tut\tu1-6\sysr\tu1-6pro.qpj`. These projects also define the `MAKE_WITH_SOURCE` symbol.

[1] A 30-day fully-functional trial version is available on the Quadravox web site.

[2] Since this App Note was originally written, AQ430 has added support for Groups in the Project Window. We recommend using Groups for clarity. New AQ430 projects contained in the Salvo distributions use the following group names: Startup Module, Salvo Configuration File, Salvo Help Files, Salvo Libraries, Salvo Sources, and Sources.

[3] This Salvo project supports a wide variety of targets and compilers. For use with AQ4430 Development Tools, it requires the `SYSR` defined symbol, as well as the symbols `MAKE_WITH_FREE_LIB` or `MAKE_WITH_STD_LIB` for library builds. When you write your own projects, you may not require any symbols.

[4] This Salvo Lite library contains all of Salvo's basic functionality. The corresponding Salvo LE and Pro library is `slaq430-a.lib`.

[5] You can Ctrl-select multiple files at once.

[6] This is usually done in the computer's BIOS.