TMS320 DSP
DESIGNER'S NOTEBOOK

Sharing Header Files in C and Assembly

APPLICATION BRIEF: SPRA205

Alan Davis
Digital Signal Processing Products
Semiconductor Group

Texas Instruments
December 1992

TEXAS INSTRUMENTS

IMPORTANT NOTICE

Texas Instruments (TI) reserves the right to make changes to its products or to discontinue any semiconductor product or service without notice, and advises its customers to obtain the latest version of relevant information to verify, before placing orders, that the information being relied on is current.

TI warrants performance of its semiconductor products and related software to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are utilized to the extent TI deems necessary to support this warranty. Specific testing of all parameters of each device is not necessarily performed, except those mandated by government requirements.

Certain application using semiconductor products may involve potential risks of death, personal injury, or severe property or environmental damage ("Critical Applications").

TI SEMICONDUCTOR PRODUCTS ARE NOT DESIGNED, INTENDED, AUTHORIZED, OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT APPLICATIONS, DEVICES OR SYSTEMS OR OTHER CRITICAL APPLICATIONS.

Inclusion of TI products in such applications is understood to be fully at the risk of the customer. Use of TI products in such applications requires the written approval of an appropriate TI officer. Questions concerning potential risk applications should be directed to TI through a local SC sales office.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards should be provided by the customer to minimize inherent or procedural hazards.

TI assumes no liability for applications assistance, customer product design, software performance, or infringement of patents or services described herein. Nor does TI warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of TI covering or relating to any combination, machine, or process in which such semiconductor products or services might be or are used.

TRADEMARKS

TI is a trademark of Texas Instruments Incorporated.

Other brands and names are the property of their respective owners.

CONTACT INFORMATION

US TMS320 HOTLINE	(281) 274-2320
US TMS320 FAX	(281) 274-2324
US TMS320 BBS	(281) 274-2323
US TMS320 email	dsph@ti.com

Contents

Figures

Sharing Header Files in C and Assembly

Abstract

Sometimes it is useful to be able to define named constants that can be used in both C and assembly language.

One method is to have separate header files that define the same symbols: a C include file with `#define` directives, and an assembler include file with `.set` or `.asg` directives.

The procedure shown in this document produces a single, shared header file that defines the symbols once for both C and assembler.

Design Problem

Sometimes it is useful to be able to define named constants that can be used in both C and assembly language.

One method is to have separate header files that define the same symbols: a C include file with `#define` directives, and an assembler include file with `.set` or `.asg` directives. But can you have a single, shared header file that defines the symbols once for both C and assembler?

Solution

The file shown in Figure 1 can be used normally as a C include file (ASMDEFS not defined). It can also be used to generate an assembler include file: compile with ASMDEFS defined and use `-k` to keep the output:

```
cl30 -dASMDEFS -k defs.h
```

```
#define PI 3.14
#define E 2.72
#ifdef ASMDEFS /*IF DEFINED, CREATE .asg DIRECTIVES*/
#define ASM_ASG(sym) asm("\t.asg\t" VAL(sym) ","
                                                #sym)
#define VAL(sym) #sym
ASM_ASG(PI);
ASM_ASG(E);
#endif /*ASMDEFS*/
```

Figure 1. Input file `defs.h`

The output is the file `defs.asm`, which contains `.asg` directives for your symbols.

```
; ... <compiler-generated header stuff> ...
.asg 3.14,PI
.asg 2.72,E
```

Figure 2. Output file `defs.asm`

You can then `.include` this file in your assembly modules. The same technique can be used to create `.set` directives rather than `.asg`.

Here's how it works: The ASM_ASG macros in `defs.h` expand to `asm` statements containing the `.asg` directives. The trick is in generating both the name and the value of the argument symbol. ASM_ASG accomplishes this with ANSI C's new stringize operator, `#`. The last expression in ASM_ASG's definition, `#sym`, simply makes a string out of the argument without expanding it. Thus, `#PI` becomes "PI". The second expression in ASM_ASG's definition calls another macro, VAL, which, in turn, stringizes its argument. But in passing `sym` to VAL, PI is expanded (to 3.14), so VAL returns "3.14". The result:

```
asm("\t.asg\t" "3.14" "," "PI")
```

In ANSI C, adjacent strings are concatenated, so this compiles down to a simple `.asg` directive in `defs.asm`.