Technical Notes on using Analog Devices' DSP components and development tools
Phone: (800) ANALOG-D, FAX: (781) 461-3010, EMAIL: dsp.support@analog.com, FTP: ftp.analog.com, WEB: www.analog.com/dsp

Initializing DSP System & Control Registers From C and C++

By Dan Ledger
Submitted 11/6/01

This application note will present techniques for setting up memory-mapped and non memory-mapped control and system registers on ADI DSPs from C or C++.

Memory Mapped Control Registers

The memory mapped control registers can be written and read by simply de-referencing a C pointer. De-referencing a C pointer simply means accessing the memory which the pointer is currently pointing to. For example, to place a value of 0xffff at address 0x10, the following notation is used:

```
* (int *) 0x10 = 0xffff;
```

The first '*' is the de-reference syntax. This is followed by, '(int *)', which is called a **cast** and tells the compiler how to handle the data following it. The * (int *) notation is the key to reading and writing explicit addresses in memory.

VisualDSP provides an include file for each processor which holds memory-mapped control register addresses and bit fields. By including this file in a .c file, you can take advantage of all of these preprocessor variables when setting up control registers. These files are typically named *defxxxx.h* where *xxxx* is the processor type and can be found in the include directory.

```
#include <def21065l.h>
```

Copy a value of 0x10 into address 0x0:

```
* (int *) 0x0 = 0x10 ;
```

Copy a value of 0x10 into the IOCTL register:

```
* (int *) IOCTL = 0x10 ;
```

Set the SDPSS bit in the IOCTL register:

```
* (int *) IOCTL |= SDPSS;
```

Set multiple bits in the IOCTL register:

```
* (int *) IOCTL |= SDPSS|SDBN2|SDBS0;
```

Clear the SDPSS bit in the IOCTL register:

```
* (int *) IOCTL &= ~SDPSS;
```

Clear multiple bits in the IOCTL register:

```
* (int *) IOCTL &= (~SDPSS) & (~SDBN2) & (~SDBS0);
```

Because the logical operations are all happening in the preprocessor (i.e. at compile time), the code generated by the compiler to initialize memory-mapped registers is just as efficient as implementing it in straight assembly language!

Non-Memory Mapped Control Registers

Control and system registers which are not memory-mapped, like the MODE2 register on the SHARC® or the IMASK register on a 218x part, require the assistance of in-lined assembly when working in C. Fortunately, there are some nice macros provided in the include files provided with VisualDSP® which can help out. These functions can be found in the sysreg.h file located in the include directory and contain the following C prototypes :

sysreg_read(r)
sysreg_write(r, val)
sysreg_bit_clr(r, bits)
sysreg_bit_set(r, bits)
sysreg_bit_tgl(r, bits)
sysreg_bit_tst(r, bits)
sysreg_tst(r, bits)

In these functions, 'r' is the enumerated register name and 'val' & 'bits' are either the direct value or the bit-fields to be set, cleared, etc.

For example, to setup the MODE2 register on the SHARC, in-lined assembly can be used like so:

```
asm("bit clr mode2 FLG2O | FLG1O | FLG0O; ");
```

Or, a sysreg function can be used:

```
sysreg_bit_clr(sysreg_MODE2, FLG2O | FLG1O | FLG0O);
```

The enumerations for the control registers are specified at the bottom of sysreg.h located in the include directory.

When using inline-assembly to setup registers, it is important to include the appropriate .h file for the assembly pre-processor as the C file and the in-lined assembly are preprocessed separately. By performing this declaration at the top of a C file, you are guaranteed that both your C and assembly preprocessor variable references will be resolved.

```
#include <def21065l.h>
asm("#include <def21065l.h>");
```