TMS320 DSP
Number 8
DESIGNER'S NOTEBOOK
TEXAS INSTRUMENTS

Bit-reversed Addressing in C on the 'C3x

Contributed by Tim Grady

Design Problem Suppose a C programmer wanted to take advantage of bit-reversed addressing. The C compiler does not support it. How does the programmer embed assembly language statements into the C code to do this?

Solution An example of how to do this is shown in Figure 1.

```
#define N 16
int x[N] = { 0,8,4,12,2,10,6,14,1,9,5,13,3,11,7,15 };
int y[N] = { 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0 };
/* int bitrev(int m, int n); */

void main()
{
   int i;
          asm("     PUSH    AR5");
          asm("     PUSH    AR0");
          asm("     LDI     8,IR0          ; initialize ir0 to 1/2 n");
          asm("     LDI     @CONST+0,AR5   ; AR5 <- address of x[]  ");
          asm("     LDI     @CONST+1,AR0   ; AR0 <- address of y[]  ");

   for ( i=0; i<N; i++ )
       {
          /* y[bitrev(i,N) ] = x[i]; */
          asm("     LDI     *AR5++(IR0)b, R0");
          asm("     STI     R0,*AR0++");
       }
   asm("   pop  ar0");
   asm("   pop  ar5");
}
          /* These statements place x and y in .bss and make their addresses
              available via the CONST table. */

asm("     .bss       CONST,2    ");
asm("     .sect      \".cinit\" ");
asm("     .word      2,CONST    ");
asm("     .word      _x         ");
asm("     .word      _y         ");
```

Figure 1. Solution example