Reentrant example

```
//ReEntrant.c
//This example has a reentrant subprogram that is used by both
// the main program and the interrupt service routine. The
// subprogram is marked "reentrant" to make this work correctly.
// To make code reentrant the parameters must be saved on a stack –
//   not in registers or memory as is usually done.  To do this
//   the compiler simulates a software stack.  You can see this
//   in the disassembly window.  The stack is at ?C_IBP
// _?ComplementBitP3:
//     USING  0
//                 ; SOURCE LINE # 32
//     DEC    ?C_IBP       ;IBP is simulated stack pointer
//     MOV    R0,?C_IBP
//     MOV    @R0,AR7
//     DEC    ?C_IBP
// ;  {unsigned char x;

#include<t89c51ac2.h>
void ComplementBitP3(unsigned char bNum) reentrant;
void main(void)
 {CKCON = 0x01; // x2 mode
  TMOD = 0x01; //Timer 0 mode = not gated, internal clock, 16-bit mode
  //For fosc = 29.4MHz in x2 mode timer is clocked at 29.4Mhz/6 = 4.9MHz so
  // period is 1/4.9Mhz = .20408 usec. To get 1 msec we need 1000/.20408
  // = 4900 counts. 65536 - 4900 = 60636 = 0xECDC.
  TH0 = 0xEC; //Timer 0 set to D9B8 - 55736
  TL0 = 0xDC;
  TR0 = 1;
  ET0 = 1; //Timer 0 interrupt enable
  EA = 1; //Global interrupt enable
  while(1)
  ComplementBitP3(0);
 }
//
void T0Int() interrupt 1
 {ComplementBitP3(2); //bit P3.2 to 1
  TR0 = 0; //Turn timer 0 off
  TH0 = 0xEC; //Timer 1 set to D9B8 = 55736
  TL0 = 0xDC;
  TR0 = 1; //Turn timer 0 on
 }
//
void ComplementBitP3(unsigned char bNum) reentrant
 {unsigned char x;
  x = 1;
  while(bNum > 0)
   {x = x << 1;
    bNum = bNum >> 1;
   }
  P3 = P3 ^ x;
 }
```