C. Watchdog Timer Configuration

The Watchdog Timer (WDT) is used to generate a variety of output signals after a user programmable count. The WDT is suitable for the use in the prevention of system lock-up, such as when software becomes trapped in a deadlock. Under these sorts of circumstances, the timer will count to zero and the selected outputs will be driven.

Under normal circumstance, you will need to restart the WDT at regular intervals before the timer counts to zero.

Sample Code

```
//---------------------------------------------------------------------------
//
// THIS CODE AND INFORMATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY
// KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
// IMPLIED WARRANTIES OF MERCHANTABILITY AND/OR FITNESS FOR A PARTICULAR
// PURPOSE.
//
//---------------------------------------------------------------------------
#include <dos.h>
#include <conio.h>
#include <stdio.h>
#include <stdlib.h>
#include "F81964.H"
//---------------------------------------------------------------------------
int main (int argc, char *argv[]); void EnableWDT(int);
void DisableWDT(void);
//---------------------------------------------------------------------------
int main (int argc, char *argv[])
{
	unsigned char bBuf;
	unsigned char bTime;
	char **endptr;

	char SIO;
	printf("Fintek 81866 watch dog program\n");
	SIO = Init_F81964();
	if (SIO == 0)
	{
		printf("Can not detect Fintek 81866, program abort.\n");
		return(1);
	}//if (SIO == 0)

	if (argc != 2)
	{
		printf(" Parameter incorrect!!\n");
		return (1);
	}

	bTime = strtol (argv[1], endptr, 10);
```

```
        printf("System will reset after %d seconds\n", bTime);

        if (bTime)
        {       EnableWDT(bTime); }
        else
        {       DisableWDT();  }
        return 0;
}
//---------------------------------------------------------------------------
void EnableWDT(int interval)
{
        unsigned char bBuf;

        bBuf = Get_F81964_Reg(0x2B);
        bBuf &= (~0x20);
        Set_F81964_Reg(0x2B, bBuf);             //Enable WDTO

        Set_F81964_LD(0x07);                    //switch to logic device 7
        Set_F81964_Reg(0x30, 0x01);             //enable timer

        bBuf = Get_F81964_Reg(0xF5);
        bBuf &= (~0x0F);
        bBuf |= 0x52;
        Set_F81964_Reg(0xF5, bBuf);             //count mode is second
        Set_F81964_Reg(0xF6, interval);         //set timer
        bBuf = Get_F81964_Reg(0xFA);
        bBuf |= 0x01;
        Set_F81964_Reg(0xFA, bBuf);             //enable WDTO output

        bBuf = Get_F81964_Reg(0xF5);
        bBuf |= 0x20;
        Set_F81964_Reg(0xF5, bBuf);             //start counting
}
//---------------------------------------------------------------------------
void DisableWDT(void)
{
        unsigned char bBuf;
        Set_F81964_LD(0x07);                    //switch to logic device 7
        bBuf = Get_F81964_Reg(0xFA);
        bBuf &= ~0x01;
        Set_F81964_Reg(0xFA, bBuf);             //disable WDTO output

        bBuf = Get_F81964_Reg(0xF5);
        bBuf &= ~0x20;
        bBuf |= 0x40;
        Set_F81964_Reg(0xF5, bBuf);             //disable WDT
}
//---------------------------------------------------------------------------
```

```
//---------------------------------------------------------------------------
//
// THIS CODE AND INFORMATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY
// KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
// IMPLIED WARRANTIES OF MERCHANTABILITY AND/OR FITNESS FOR A
PARTICULAR
// PURPOSE.
//
//---------------------------------------------------------------------------
#include "F81964.H"
#include <dos.h>
//---------------------------------------------------------------------------
unsigned int F81964_BASE; void Unlock_F81964 (void); void Lock_F81964 (void);
//---------------------------------------------------------------------------
unsigned int Init_F81964(void)
{
	unsigned int result;
	unsigned char ucDid;

	F81964_BASE = 0x4E;
	result = F81964_BASE;

	ucDid = Get_F81964_Reg(0x20);
	if (ucDid == 0x07)			//Fintek 81866
	{	goto Init_Finish;}

	F81964_BASE = 0x2E;
	result = F81964_BASE;

	ucDid = Get_F81964_Reg(0x20);
	if (ucDid == 0x07)			//Fintek 81866
	{	goto Init_Finish;}

	F81964_BASE = 0x00;
	result = F81964_BASE;

Init_Finish:
	return (result);
}
//---------------------------------------------------------------------------
void Unlock_F81964 (void)
{
	outportb(F81964_INDEX_PORT, F81964_UNLOCK);
	outportb(F81964_INDEX_PORT, F81964_UNLOCK);
}
//---------------------------------------------------------------------------
void Lock_F81964 (void)
{
	outportb(F81964_INDEX_PORT, F81964_LOCK);
}
//---------------------------------------------------------------------------
void Set_F81964_LD( unsigned char LD)
{
	Unlock_F81964();
```

```
        outportb(F81964_INDEX_PORT, F81964_REG_LD);
        outportb(F81964_DATA_PORT, LD); Lock_F81964();
}
//-----------------------------------------------------------------------------
void Set_F81964_Reg( unsigned char REG, unsigned char DATA)
{
        Unlock_F81964();
        outportb(F81964_INDEX_PORT, REG);
        outportb(F81964_DATA_PORT, DATA);
        Lock_F81964();
}
//-----------------------------------------------------------------------------
unsigned char Get_F81964_Reg(unsigned char REG)
{
        unsigned char Result;
        Unlock_F81964();
        outportb(F81964_INDEX_PORT, REG);
        Result = inportb(F81964_DATA_PORT);
        Lock_F81964();
        return Result;
}
//-----------------------------------------------------------------------------
```

```
//-----------------------------------------------------------------------------
//
// THIS CODE AND INFORMATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY
// KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
// IMPLIED WARRANTIES OF MERCHANTABILITY AND/OR FITNESS FOR A
PARTICULAR
// PURPOSE.
//
//-----------------------------------------------------------------------------
#ifndef         F81964_H
#define         F81964_H     1
//-----------------------------------------------------------------------------
#define     F81964_INDEX_PORT      (F81964_BASE)
#define     F81964_DATA_PORT       (F81964_BASE+1)
//-----------------------------------------------------------------------------
#define     F81964_REG_LD      0x07
//-----------------------------------------------------------------------------
#define F81964_UNLOCK 0x87
#define     F81964_LOCK  0xAA
//-----------------------------------------------------------------------------
unsigned int Init_F81964(void);
void Set_F81964_LD( unsigned char);
void Set_F81964_Reg( unsigned char, unsigned char); unsigned char
Get_F81964_Reg( unsigned char);
//-----------------------------------------------------------------------------
#endif //     F81964_H
```