DELAY TIMER

```
#include"header.h"
void delay_ms(unsigned int ms)
{
	int a[]={15,60,30,15,15};
	T0PC=0;//STARTING VALUE
	T0PR=a[VPBDIV]*1000-1;//ending value
	T0TC=0;
	T0TCR=1;//start timer
	while(T0TC<ms);//count overflow
	T0TCR=0;//stop timer
}
void delay_sec(unsigned int sec)
{
	int a[]={15,60,30,15,15};
	T0PC=0;//STARTING VALUE
	T0PR=a[VPBDIV]*1000000-1;
	T0TC=0;
	T0TCR=1;
	while(T0TC<sec);
	T0TCR=0;
}
```

ADC DRIVER

```
#include"header.h"
void adc_init()
{
	PINSEL1|=0x15400000;
	ADCR=0X00200400;
}
#define DONE ((ADDR>>31)&1)
unsigned int adc_read(int ch_num)
{
	unsigned int result=0;
	ADCR|=(1<<ch_num);
	ADCR|=(1<<24);
	while(DONE==0);
	ADCR^=(1<<24);
	ADCR^=(1<<ch_num);
	result=(ADDR>>6)&0x03ff;
	return result;
}
```

ADC PROGRAM

```
#include"header.h"
main()
{
	int pot;
	float vtg;
	int temp;
	float vout,result;
	adc_init();
	lcd_init();
	while(1)
	{
		pot=adc_read(2);//potentiometer
		temp=adc_read(1);//temperature sensor
		vtg=(pot*3.3)/1023;//potentiometer
		vout=(temp*3.3)/1023;//temp sensor
		result=(vout-0.5)/0.01;//temp sensor
		lcd_cmd(0x80);
		lcd_string("pot vtg:");
		lcd_float(vtg);
		lcd_cmd(0xc0);
		lcd_string("temp:");
		lcd_float(result);
		delay_ms(200);
		lcd_cmd(0x01);
	}
}
```

LCD 4 BIT DRIVER

```
#include"header.h"
#define RS (1<<17)
#define RW (1<<18)
#define EN (1<<19)
void lcd_cmd(unsigned char cmd)
{
        unsigned int temp;
        IOCLR1=0X00FE0000;
        temp=(cmd&0xf0)<<16;
        IOSET1=temp;
        IOCLR1=RS;
        IOCLR1=RW;
        IOSET1=EN;
        delay_ms(2);
        IOCLR1=EN;

        IOCLR1=0X00FE0000;
        temp=(cmd&0x0f)<<20;
        IOSET1=temp;
        IOCLR1=RS;
        IOCLR1=RW;
        IOSET1=EN;
        delay_ms(2);
        IOCLR1=EN;
}
void lcd_data(unsigned char data)
{
        unsigned int temp;
        IOCLR1=0X00FE0000;
        temp=(data&0xf0)<<16;
        IOSET1=temp;
        IOSET1=RS;
        IOCLR1=RW;
        IOSET1=EN;
        delay_ms(2);
        IOCLR1=EN;

        IOCLR1=0X00FE0000;
        temp=(data&0x0f)<<20;
        IOSET1=temp;
        IOSET1=RS;
        IOCLR1=RW;
        IOSET1=EN;
        delay_ms(2);
        IOCLR1=EN;
}
```

```
void lcd_init(void)
{
	IODIR1|=0X00FE0000;
	PINSEL2=0;
	lcd_cmd(0x02);
	lcd_cmd(0x03);
	lcd_cmd(0x28);
	lcd_cmd(0x0e);
	lcd_cmd(0x01);
}
void lcd_string(const char *p)
{
	while(*p)
	{
		lcd_data(*p);
		p++;
	}
}
void lcd_integer(int num)
{
	int i;
	char a[10];
	if(num<0)
	{
		lcd_data('-');
		num=-num;
	}
	if(num==0)
		{
			lcd_data('0');
			return;
		}
	for(i=0;num;i++,num=num/10)
		a[i]=num%10+48;
	for(i--;i>=0;i--)
		lcd_data(a[i]);
}
```

```
void lcd_float(float f)
{
	int num;
	if(f<0)
	{
		lcd_data('-');
		f=-f;
	}
	if(f==0)
	{
		lcd_string("0.0");
		return;
	}
	num=f;
	lcd_integer(num);
	lcd_data('.');
	num=(f-num)*1000000;
	lcd_integer(num);
}
unsigned char a[]={0x0e,0x1f,0x15,0x1f,0x15,0x15,0x11,0x0e};
unsigned char a1[]={0x17,0x14,0x14,0x1f,0x05,0x05,0x1D,0x00};
void lcd_cgram(void)
{
	int i;
	lcd_cmd(0x40);
	for(i=0;i<8;i++)
		lcd_data(a[i]);
	lcd_cmd(0x48);
	for(i=0;i<8;i++)
		lcd_data(a1[i]);
}
```

SCROLLING LCD RIGHT

```
#include"header.h"
main()
{
	int i;
	lcd_init();
	while(1)
	{
		for(i=0;i<16;i++)
		{
			lcd_cmd(0x80+i);
			lcd_string("vector");
			delay_ms(500);
			lcd_cmd(0x01);
		}
	}
}
```

SCROLLING LCD LEFT

```
#include"header.h"
main()
{
	int i;
	lcd_init();
	while(1)
	{
		for(i=15;i>=0;i--)
		{
			lcd_cmd(0x80+i);
			lcd_string("vector");
			delay_ms(200);
			lcd_cmd(0x01);
		}
	}
}
```

CIRCULAR SCROLL

```
#include"header.h"
main()
{
	int i;
	lcd_init();
	lcd_cmd(0x0c);
	while(1)
	{
	for(i=0;i<16;i++)
	{
		lcd_cmd(0x80+i);
		lcd_string("vector	");
		lcd_cmd(0x80);
		lcd_string("vector	"+16-i);
		delay_ms(500);
		lcd_cmd(0x01);
	}
	}
}
```

FLASH 5 TIMES

```
#include"header.h"
main()
{
	int i;
	lcd_init();
	for(i=0;i<5;i++)
	{
		lcd_string("vector");
		delay_ms(200);
		lcd_cmd(0x01);
		delay_ms(200);
	}
}
```

LCD CGRAM

```
#include"header.h"
main()
{
        lcd_init();
        lcd_cgram();
        lcd_cmd(0x8f);
        lcd_data(0);
        lcd_cmd(0xc0);
        lcd_data(1);
}
```

LCD PRINT 0 TO 59

```
#include"header.h"
main()
{
        int i;
        lcd_init();
        while(1)
        {
                for(i=0;i<60;i++)
                {
                        lcd_cmd(0xc0);
                        lcd_data(i/10+48);
                        lcd_data(i%10+48);
                        delay_ms(200);
                        lcd_cmd(0x01);
                        delay_ms(200);
                }
        }
}
```

UART DRIVER

```
#include"header.h"
void uart0_init(unsigned int baud)
{
	int a[]={15,60,30,15,15};
	unsigned int pclk=a[VPBDIV]*1000000;
	unsigned int result=pclk/(16*baud);
	PINSEL0|=0X05;
	U0LCR=0X83;
	U0DLL=result&0xff;
	U0DLM=(result>>8)&0xff;
	U0LCR=0X03;
}
#define THRE ((U0LSR>>5)&1)
#define RDR (U0LSR&1)

void uart0_tx(unsigned char data)
{
	U0THR=data;
	while(THRE==0);
}
unsigned char uart0_rx(void)
{
	while(RDR==0);
	return U0RBR;
}
void uart0_tx_string(const char *p)
{
	while(*p)
	{
		U0THR=*p;
		while(THRE==0);
		p++;
	}
}
void uart0_rx_string(char *p,unsigned int max_size)
{
	int i;
	for(i=0;i<max_size;i++)
	{
		while(RDR==0);
		p[i]=U0RBR;
		if(p[i]=='\r')
			break;
	}
	p[i]='\0';
}
```

```
void uart0_tx_integer(int num)
{
	char a[10];
	int i;
	if(num<0)
	{
		uart0_tx('-');
		num=-num;
	}
	if(num==0)
	{
		uart0_tx('0');
		return;
	}
	for(i=0;num;num=num/10,i++)
		a[i]=num%10+48;
	for(--i;i>=0;i--)
		uart0_tx(a[i]);
}
void uart0_tx_float(float f)
{
	int num;
	if(f<0)
	{
		uart0_tx('-');
		f=-f;
	}
	if(f==0)
	{
		uart0_tx_string("0.0");
		return;
	}
	num=f;
	uart0_tx_integer(num);
	uart0_tx('.');
	num=(f-num)*1000000;
	uart0_tx_integer(num);
}
```

```
void uart0_tx_hex(int num)
{
	unsigned char temp;
	uart0_tx_string("0x");
	temp=(num&0xf0)>>4;
	if(temp>9)
		uart0_tx('A'+temp-10);
	else
		uart0_tx(temp+48);
	temp=num&0x0f;
	if(temp>9)
		uart0_tx('A'+temp-10);
	else
		uart0_tx(temp+48);
}
int uart0_rx_integer()
{
	int i,num1;
	char s[10];
	uart0_rx_string(s,10);
	if(s[0]=='-' || s[0]=='+')
		i=1;
	else
		i=0;
		for(i,num1=0;s[i];i++)
		{
			num1=num1*10+(s[i]-48);
		}
	if(s[0]=='-')
		num1=-num1;
	return num1;
}
```

```
float uart0_rx_float()
{
	int i,num1,l=0,num2;
	float result,num3;
	char s[10];
	uart0_rx_string(s,10);
	if(s[0]=='-' || s[0]=='+')
		i=1;
	else
		i=0;
	for(i,num1=0;s[i]&&s[i]!='.';i++)
	{
		num1=num1*10+(s[i]-48);
	}
	i++;
	for(i;s[i];i++)
	{
		num2=num2*10+(s[i]-48);
	}
	num3=num2;
	for(num2;num2;num2=num2/10)
		l++;
	for(i=0;i<l;i++)
		num3=num3/10;
	result=num1+num3;
	if(s[0]=='-')
		result=-result;
	return result;
}
```

CALCULATOR USING UART

```
#include"header.h"
main()
{
	char s[10];
	int num1,num2,num3,i,c;
	uart0_init(9600);
	while(1)
	{
		uart0_tx_string("\r\nenter expression:");
		uart0_rx_string(s,10);
		for(i=0,num1=0;(s[i]>='0'&&s[i]<='9');i++)
			num1=num1*10+(s[i]-48);
		c=i;
		for(++i;s[i];i++)
			num2=num2*10+(s[i]-48);
		switch(s[c])
		{
			case '+':num3=num1+num2;uart0_tx_integer(num3);break;
			case '-':num3=num1-num2;uart0_tx_integer(num3);break;
			case '*':num3=num1*num2;uart0_tx_integer(num3);break;
			case '/':num3=num1/num2;uart0_tx_integer(num3);break;
		}
		num1=num2=num3=0;
	}
}
```

LED PROJECT ON UART

```
#include"header.h"
#define LED1 (1<<17)
#define LED2 (1<<18)
main()
{
	char temp;
	IODIR0=LED1|LED2;
	IOSET0=LED1|LED2;
	uart0_init(9600);
	while(1)
	{
		uart0_tx_string("\r\nenter option:\r\na for led1 on\r\nb for led1 off\r\nc for led2 on\r\
nd for led2 off\r\n");
		temp=uart0_rx();
		uart0_tx(temp);
		switch(temp)
		{
			case 'a':IOCLR0|=LED1;uart0_tx_string("\r\n led1 on");break;
			case 'b':IOSET0|=LED1;uart0_tx_string("\r\n led1 off");break;
			case 'c':IOCLR0|=LED2;uart0_tx_string("\r\n led2 on");break;
			case 'd':IOSET0|=LED2;uart0_tx_string("\r\n led2 off");break;
			default :uart0_tx_string("\r\nunknown option");
		}
	}
}
```

PRINT ASCII AND HEX IN UART

```
#include"header.h"
main()
{
	char temp;
	uart0_init(9600);
	while(1)
	{
		uart0_tx_string("\r\nenter ascii:");
		temp=uart0_rx();
		uart0_tx(temp);
		while(uart0_rx()!='\r');
			uart0_tx_string("\r\n ascii:");
			uart0_tx_integer(temp);
			uart0_tx_string("\r\n hex value:");
			uart0_tx_hex(temp);
	}
}
```

PRIME OR NOT

```
#include"header.h"
#include<string.h>
int prime(int num);
main()
{
	int num;
	uart0_init(9600);
	while(1)
	{
		uart0_tx_string("\r\nenter num:");
		num=uart0_rx_integer();
		uart0_tx_integer(num);
		if(prime(num))
			uart0_tx_string("\r\nprime number");
		else
			uart0_tx_string("\r\nnon-prime number");
	}
}
int prime(int num)
{
	int i;
	for(i=2;i<num;i++)
	{
		if(num%i==0)
			break;
	}
	if(num==i)
		return 1;
	else
		return 0;
}
```

SINGLE DIGIT CALCULATOR

```
#include"header.h"
main()
{
	int i,result;
	char s[4];
	uart0_init(9600);
	while(1)
	{
		uart0_tx_string("\r\nenter expression:");
		uart0_rx_string(s,4);
		result=s[0]-48;
		if(s[1]=='+')
			result+=(s[2]-48);
		if(s[1]=='-')
			result-=(s[2]-48);
		if(s[1]=='*')
			result*=(s[2]-48);
		if(s[1]=='/')
			result/=(s[2]-48);

		uart0_tx_integer(result);
	}
}
```

UART0 INTERRUPT DRIVER

```
#include"header.h"
extern unsigned char temp;
void uart0_handler(void)__irq
{
	int r=U0IIR;//identifies uart0 interrupt(rx int or tx int)
	if(r&4)//if data recieved
	{
		temp=U0RBR;//rx data
		U0THR=temp;//loop back
	}
	if(r&2)//if data transmitted
	{

	}
	VICVectAddr=0;//to finish isr execution
}
void config_vic(void)
{
	VICIntSelect=0;//all irq interrupt
	VICVectCntl0=6|(1<<5);//int num 6 (uart0) to slot 0 and enable slot 0
	VICVectAddr0=(int)uart0_handler;//store uart0 isr to slot 0
	VICIntEnable|=(1<<6);//enable uart0 interrupt
}
void en_uart0_interrupt(void)
{
	U0IER=0x3;//en tx and rx int
}
```

LED INTERRUPTS PROGRAMS USING UART INTERRUPT

```
#include"header.h"
#define LED1 (1<<17)
#define LED2 (1<<18)
#define LED3 (1<<19)
#define sw ((IOPIN0>>14)&1)
#define sw1 ((IOPIN0>>14)&1)
#define sw2 ((IOPIN0>>15)&1)
#define sw3 ((IOPIN0>>16)&1)
unsigned char temp;
main()
{
	int i,j,flag;
	uart0_init(9600);
	config_vic();
	en_uart0_interrupt();
	IODIR0=0XFF;
	IOCLR0=0XFF;
	IODIR0|=(7<<17);
	while(1)
	{
		while(temp=='a')//8 led blink
		{
			IOSET0=0XFF;
			delay_ms(50);
			IOCLR0=0XFF;
			delay_ms(50);
		}
		while(temp=='b')//1 led blink
		{
			IOSET0=0X1;
			delay_ms(50);
			IOCLR0=0X1;
			delay_ms(50);
		}
		while(temp=='c')//led shift left
		{
			for(i=0;i<8;i++)
			{
				IOSET0=1<<i;
				delay_ms(50);
				IOCLR0=1<<i;
				delay_ms(50);
			}
		}
```

```
while(temp=='d')//led shift right
{
	for(i=7;i>=0;i--)
	{
		IOSET0=1<<i;
		delay_ms(50);
		IOCLR0=1<<i;
		delay_ms(50);
	}
}
while(temp=='e')//car indicator right
{
	IOCLR0=0XFF;
	for(i=7;i>=0;i--)
	{
		IOSET0|=1<<i;
		delay_ms(50);
	}
	for(i=7;i>=0;i--)
	{
		IOCLR0|=1<<i;
		delay_ms(50);
	}
}
while(temp=='f')//car indicator left
{
	IOCLR0=0XFF;
	for(i=0;i<8;i++)
	{
		IOSET0|=1<<i;
		delay_ms(50);
	}
	for(i=0;i<8;i++)
	{
		IOCLR0|=1<<i;
		delay_ms(50);
	}
}
while(temp=='g')//binary print 0 to 7
{
	IOCLR0=7<<17;
	for(i=0;i<8;i++)
	{
		IOSET0=i<<17;
		delay_ms(50);
		IOCLR0=i<<17;
		delay_ms(50);
	}
}
```

```
while(temp=='h')//conversion and diversion
{
	IOCLR0=0XFF;
	for(i=0,j=7;i<j;i++,j--)
	{
		IOSET0|=(1<<i)|(1<<j);
		delay_ms(50);
	}
	for(i,j;i>=0;i--,j++)
	{
		IOCLR0|=(1<<i)|(1<<j);
		delay_ms(50);
	}
}
while(temp=='i')//while swwitch is pressend led blink and released led off
{
	if(sw==0)
	{
		IOSET0=LED1;
		delay_ms(50);
		IOCLR0=LED1;
		delay_ms(50);
	}
	else
		IOCLR0=LED1;
}
while(temp=='j')//toggle led when switch is pressed
{
	flag=0;
	if(sw==0)
		flag=!flag;
	if(flag)
		IOSET0=LED1;
	else
		IOCLR0=LED1;
}
while(temp=='k')//binary print using switch
{
	IOSET0=i<<17;
	while(sw==0)
	{
		while(sw==0);
		IOCLR0=i<<17;
		i++;
	}
	if(i>7)
		i=0;
}
```

```
		while(temp=='l')//3 switch 3 led
		{
			if(sw1==0)
				IOSET0=LED1;
			if(sw2==0)
				IOSET0=LED2;
			if(sw3==0)
				IOSET0=LED3;
			else
				IOCLR0=LED1|LED2|LED3;
		}
	}
}
```

LED INTERRUPT BLINKING USING UART INTERRUPT INTERRUPT

```
#define LED1 (1<<17)
#define LED2 (1<<18)
unsigned char temp;
main(){
	uart0_init(9600);
	config_vic();
	en_uart0_interrupt();
	IODIR0|=LED1|LED2;
	IOSET0|=LED1|LED2;
	while(1)
	{
		while(temp=='a')
		{
			IOCLR0=LED1;
			delay_ms(500);
			IOSET0=LED1;
			delay_ms(500);
		}
		while(temp=='b')
		{
			IOCLR0=LED2;
			delay_ms(500);
			IOSET0=LED2;
			delay_ms(500);
		}
	}
}
```

LCD INTERRUPT SCROLLING USING UART INTERRUPT

```
#include"header.h"
unsigned char temp;
main()
{
	int i,pot;
	float vtg;
	uart0_init(9600);
	lcd_init();
	adc_init();
	config_vic();
	en_uart0_interrupt();
	while(1)
	{
		while(temp=='a')
		{
			for(i=0;i<16;i++)
			{
				lcd_cmd(0xc0);
				lcd_string("right");
				lcd_cmd(0x80+i);
				lcd_string("vector");
				delay_ms(200);
				lcd_cmd(0x01);
			}
		}
		while(temp=='b')
		{
			for(i=15;i>=0;i--)
			{
				lcd_cmd(0xc0);
				lcd_string("left");
				lcd_cmd(0x80+i);
				lcd_string("vector");
				delay_ms(200);
				lcd_cmd(0x01);
			}
		}
		while(temp=='c')
		{
			for(i=0;i<15;i++)
			{
				lcd_cmd(0xc0);
				lcd_string("circular right");
				lcd_cmd(0x80+i);
				lcd_string("vector	");
				lcd_cmd(0x80);
				lcd_string("vector	"+16-i);
				delay_ms(200);
				lcd_cmd(0x01);
			}
		}
```

```
		while(temp=='d')
		{
			for(i=15;i>=0;i--)
			{
				lcd_cmd(0xc0);
				lcd_string("circular left");
				lcd_cmd(0x80+i);
				lcd_string("vector	");
				lcd_cmd(0x80);
				lcd_string("vector	"+16-i);
				delay_ms(200);
				lcd_cmd(0x01);
			}
		}
		while(temp=='e')
		{
			pot=adc_read(2);
			vtg=(pot*3.3)/1023;
			lcd_cmd(0x80);
			lcd_string("vtg:");
			lcd_cmd(0xc0);
			lcd_float(vtg);
			delay_ms(200);
			lcd_cmd(0x01);
		}
	}
}
```

ADC INTERRUPT USING UART INTERRUPT

```
#include"header.h"
unsigned char temp;
unsigned int count2;
main()
{
	unsigned int pot;
	float vtg,temp1;
	uart0_init(9600);
	lcd_init();
	adc_init();
	config_vic();
	en_uart0_interrupt();
	while(1)
	{
		while(temp=='a')
		{
			pot=adc_read(2);
			vtg=pot*3.3/1023;
			lcd_cmd(0x80);
			lcd_string("vtg:");
			lcd_cmd(0xc0);
			lcd_float(vtg);
		}
		while(temp=='b')
		{
			pot=adc_read(1);
			vtg=pot*3.3/1023;
			temp1=(vtg-0.5)/0.01;
			lcd_cmd(0x80);
			lcd_string("temp:");
			lcd_cmd(0xc0);
			lcd_float(temp1);
		}
	}
}
```