Laboration 1

Group 5: Xu Fei, Qiu Yinan
November 19, 2012

1.Introduction

This report is mainly about the assignment of Lab1, which is required to write some application of AVR Butterfly. Source code with comments is in lab1.1c, lab1.2c, lab1.3c and lab1.4c files. All function is tested and excutable.

2.Answer the questions

Question 1: Try out your program for different start values for the prime number computations, for example 1, 5000, and 25000. How do these values affect the display of the 1 Hz blinking segment? How is the response time of the joystick affected? What would it take to ensure a stable blinking display and a responsive joystick, that are independent of the currently calculated prime number?

Answer 1: The more numbers need to be calculated in is_prime(), the more response time we need for the 1Hz blinking segment and the joystick. But because we used sqrt(i) in the loop of determine the prime, so we don’t need to calculate a lot of numbers, different start values can’t affect too much to the response time of blinking display and joystick. If we want to ensure a stable blinking display and a responsive joystick, we need to reduce the CPU utilization of function is_prime(), the more efficient of the function is_prime(), the more stable blinking display and the more responsive joystick.

Question 2: What is there to say about the structure of your combined program? Was some clarity of your individual solutions lost in the adaption process? Why wasn't it possible to simply call the original functions one after another in the main program? If you didn't modify is_prime() in your solution to part 4, what stopped you from doing that in order to obtain constant response times?

Answer 2: We don’t need to loop for n number when we decide n is a prime number or not, we only need to loop for $\sqrt{\mathrm{n}}$ numbers when we want to decide n is a prime number or not, the time complexity of function is_prime() is only $\log_2 n$. When n becoming bigger and bigger, the function will be more efficient.

3.Code:

Exercise 1: prime numbers and the LCD.

```
/*
 * lab1.c
 *
 * Created: 2012/11/15 22:14:39
 *  Author: xufei
 */
#include <avr/io.h>
#include <math.h>

#define beep_on PORTB |=1<<5
#define beep_off PORTB &=(~(1<<5))

void delay()
{
    TCNT0=0x00;
    TCCR0A=0x02;
    while(TCNT0!=0xb0);
    TCCR0A=0x00;
}



void beep()
{
    unsigned int length=50;
    unsigned char i;
    DDRB |=1<<5;
    for(i=0;i<length;i++)
    {
        beep_on;
        delay();
        beep_off;
        delay();
    }
    DDRB &=~(1<<5);
}

/*

         0       1       2       3       4       5       6       7       8       9
DR0      0001    1000    0001    0001    0000    0001    0001    0001    0001    0001
DR5      0101    0001    0001    0001    0101    0100    0100    0001    0101    0101
DR10     0101    0001    1110    1011    1011    1011    1111    0001    1111    1011
DR15     0001    0000    0001    0001    0000    0001    0001    0000    0001    0001
```

```
       1551   8110   11E1   11B1   05B0   14B1   14F1   1110   15F1   15B1

*/
void writeChar(char ch, int pos)
{
    int n1,n2,n3,n4;
    switch(ch)
    {
        case '0': n1=1; n2=5; n3=5; n4=1;break;
        case '1': n1=8; n2=1; n3=1; n4=0;break;
        case '2': n1=1; n2=1; n3=0xE; n4=1;break;
        case '3': n1=1; n2=1; n3=0xB; n4=1;break;
        case '4': n1=0; n2=5; n3=0xB; n4=0;break;
        case '5': n1=1; n2=4; n3=0xB; n4=1;break;
        case '6': n1=1; n2=4; n3=0xF; n4=1;break;
        case '7': n1=1; n2=1; n3=0x1; n4=0;break;
        case '8': n1=1; n2=5; n3=0xF; n4=1;break;
        case '9': n1=1; n2=5; n3=0xB; n4=1;break;
    }
    switch(pos)
    {
        case 0: LCDDR0  = LCDDR0&0xf0;
                LCDDR5  = LCDDR5&0xf0;
                LCDDR10 = LCDDR10&0xf0;
                LCDDR15 = LCDDR15&0xf0;
                LCDDR0  = LCDDR0|n1;
                LCDDR5  = LCDDR5|n2;
                LCDDR10 = LCDDR10|n3;
                LCDDR15 = LCDDR15|n4;
                 break;
        case 1: LCDDR0  = LCDDR0&0x0f;
                LCDDR5  = LCDDR5&0x0f;
                LCDDR10 = LCDDR10&0x0f;
                LCDDR15 = LCDDR15&0x0f;
                LCDDR0  = LCDDR0|(n1<<4);
                LCDDR5  = LCDDR5|(n2<<4);
                LCDDR10 = LCDDR10|(n3<<4);
                LCDDR15 = LCDDR15|(n4<<4);
                 break;
        case 2: LCDDR1  = LCDDR1&0xf0;
                LCDDR6  = LCDDR6&0xf0;
                LCDDR11 = LCDDR11&0xf0;
                LCDDR16 = LCDDR16&0xf0;
                LCDDR1  = LCDDR1|n1;
```

```
                 LCDDR6  = LCDDR6|n2;
                 LCDDR11 = LCDDR11|n3;
                 LCDDR16 = LCDDR16|n4;
                  break;
        case 3: LCDDR1  = LCDDR1&0x0f;
                LCDDR6  = LCDDR6&0x0f;
                LCDDR11 = LCDDR11&0x0f;
                LCDDR16 = LCDDR16&0x0f;
                 LCDDR1  = LCDDR1|(n1<<4);
                LCDDR6  = LCDDR6|(n2<<4);
                LCDDR11 = LCDDR11|(n3<<4);
                LCDDR16 = LCDDR16|(n4<<4);
                  break;
         case 4: LCDDR2  = LCDDR2&0xf0;
                 LCDDR7  = LCDDR7&0xf0;
                 LCDDR12 = LCDDR12&0xf0;
                 LCDDR17 = LCDDR17&0xf0;
                 LCDDR2  = LCDDR2|n1;
             LCDDR7 = LCDDR7|n2;
                 LCDDR12  = LCDDR12|n3;
                 LCDDR17 = LCDDR17|n4;
                  break;
         case 5: LCDDR2  = LCDDR2&0x0f;
                 LCDDR7  = LCDDR7&0x0f;
                 LCDDR12 = LCDDR12&0x0f;
                 LCDDR17 = LCDDR17&0x0f;
                 LCDDR2  = LCDDR2|(n1<<4);
                 LCDDR7 = LCDDR7|(n2<<4);
                 LCDDR12  = LCDDR12|(n3<<4);
                 LCDDR17 = LCDDR17|(n4<<4);
                  break;
    }
}

int longSize(long i)
{
    if (i%10==i)
    {
        return 1;
    }
    else if (i%100==i)
    {
        return 2;
    }
    else if (i%1000==i)
    {
        return 3;
    }
    else if (i%10000==i)
    {
        return 4;
    }
    else if (i%100000==i)
    {
        return 5;
    }
    else
    return 6;
}


void writeLong(long i)
{
    int position = 0;
    char number = ’0’;
    int num = 0;
    int numsize;
    numsize = longSize(i);
    int j;
    int temp=1;

    for (position=0; position<numsize; position++)
    {
       switch (position)
        {
            case 5: num = (i%1000000-i%100000)/100000; number = num+’0’;break;
            case 4: num = (i%100000-i%10000)/10000; number = num+’0’;break;
            case 3: num = (i%10000-i%1000)/1000; number = num+’0’;break;
            case 2: num = (i%1000-i%100)/100; number = num+’0’;break;
            case 1: num = (i%100-i%10)/10; number = num+’0’;break;
            case 0: num = i%10; number = num+’0’;break;
        }
        switch (position)
        {
            case 0:writeChar(number,5);break;
            case 1:writeChar(number,4);break;
            case 2:writeChar(number,3);break;
            case 3:writeChar(number,2);break;
            case 4:writeChar(number,1);break;
            case 5:writeChar(number,0);break;
        }

    }
}

int is_prime(long i)
{
    int loop;
    loop = sqrt(i);
    int j = 0;
    int amount = 0;
    for (j=2; j<=loop+1; j++)
    {
        if (i%j == 0)
        {
            amount++;
        }
    }
    if (amount != 0)
    {
        return 0;
    }
    else
    return 1;
}

void primes()
{

    long i = 0;
    for (i=2; i<1000000; i++)
    {
        if ((is_prime(i))==1)
        {
            writeLong(i);
        }
    }
}

int main(void)
{
        LCDCRA  = 0x80;
        LCDCRB  = 0xB7;
        primes();
}
```

Exercise 2: blinking and busy-waiting.

```
/*
* lab1.c
*
* Created: 2012/11/16 11:53:05
*  Author: xufei
*/


#include <avr/io.h>
static int symbool=1;

int main(void)
{
    TCNT1 = 0;
    LCDCRA  = 0x80;
    LCDCRB  = 0xB7;
    TCCR1B = 0x04;
    while (1)
    {
        if (TCNT1>=32767)
        {
            symbool = 1-symbool;
            if (symbool==0)
            {
                LCDDR18 = 0x01;
                TCNT1 = 0;
            }
            else
            {
                LCDDR18 = 0x00;
                TCNT1 = 0;
            }
        }

    }
}
```

Exercise 3: the joystick and busy-waiting.

```
/*
```

```
 * lab1.c
 *
 * Created: 2012/11/17 0:48:23
 *  Author: xufei
 */
#include <avr/io.h>

int main(void)
{
    LCDCRA  = 0x80;
    LCDCRB  = 0xB7;
    PORTB = PORTB|0x80;
    while (1)
    {
          if ((PINB&0x80)!=0x80)
          {
              LCDDR13 &= 0x00;
          }
          else
          LCDDR13 |= 0x01;
    }
}
```

Exercise 4: putting things together ...

```
/*
 * lab1.c
 *
 * Created: 2012/11/17 1:23:33
 *  Author: xufei
 */
#include <avr/io.h>
#include <math.h>
#define beep_on PORTB |=1<<5
#define beep_off PORTB &=(~(1<<5))
static int flag=1;

void delay()
{
    TCNT0=0x00;
    TCCR0A=0x02;
    while(TCNT0!=0xb0);
    TCCR0A=0x00;
}
```

```
void beep()
{
    unsigned int length=50;
    unsigned char i;
    DDRB |=1<<5;
    for(i=0;i<length;i++)
    {
        beep_on;
        delay();
        beep_off;
        delay();
    }
    DDRB &=~(1<<5);
}

/*

        0      1      2      3      4      5      6      7      8      9
DR0    0001   1000   0001   0001   0000   0001   0001   0001   0001   0001
DR5    0101   0001   0001   0001   0101   0100   0100   0001   0101   0101
DR10   0101   0001   1110   1011   1011   1011   1111   0001   1111   1011
DR15   0001   0000   0001   0001   0000   0001   0001   0000   0001   0001
       1551   8110   11E1   11B1   05B0   14B1   14F1   1110   15F1   15B1

*/
void writeChar(char ch, int pos)
{
    int n1,n2,n3,n4;
    switch(ch)
    {
        case '0': n1=1; n2=5; n3=5; n4=1;break;
        case '1': n1=8; n2=1; n3=1; n4=0;break;
        case '2': n1=1; n2=1; n3=0xE; n4=1;break;
        case '3': n1=1; n2=1; n3=0xB; n4=1;break;
        case '4': n1=0; n2=5; n3=0xB; n4=0;break;
        case '5': n1=1; n2=4; n3=0xB; n4=1;break;
        case '6': n1=1; n2=4; n3=0xF; n4=1;break;
        case '7': n1=1; n2=1; n3=0x1; n4=0;break;
        case '8': n1=1; n2=5; n3=0xF; n4=1;break;
        case '9': n1=1; n2=5; n3=0xB; n4=1;break;
    }
    switch(pos)
    {
```

```
 case 0: LCDDR0  = LCDDR0&0xf0;
         LCDDR5  = LCDDR5&0xf0;
         LCDDR10 = LCDDR10&0xf0;
         LCDDR15 = LCDDR15&0xf0;
         LCDDR0  = LCDDR0|n1;
         LCDDR5  = LCDDR5|n2;
         LCDDR10 = LCDDR10|n3;
         LCDDR15 = LCDDR15|n4;
          break;
 case 1: LCDDR0  = LCDDR0&0x0f;
         LCDDR5  = LCDDR5&0x0f;
         LCDDR10 = LCDDR10&0x0f;
         LCDDR15 = LCDDR15&0x0f;
         LCDDR0  = LCDDR0|(n1<<4);
         LCDDR5  = LCDDR5|(n2<<4);
         LCDDR10 = LCDDR10|(n3<<4);
         LCDDR15 = LCDDR15|(n4<<4);
          break;
 case 2: LCDDR1  = LCDDR1&0xf0;
         LCDDR6  = LCDDR6&0xf0;
         LCDDR11 = LCDDR11&0xf0;
         LCDDR16 = LCDDR16&0xf0;
         LCDDR1  = LCDDR1|n1;
         LCDDR6  = LCDDR6|n2;
         LCDDR11 = LCDDR11|n3;
         LCDDR16 = LCDDR16|n4;
          break;
case 3: LCDDR1  = LCDDR1&0x0f;
        LCDDR6  = LCDDR6&0x0f;
        LCDDR11 = LCDDR11&0x0f;
        LCDDR16 = LCDDR16&0x0f;
         LCDDR1  = LCDDR1|(n1<<4);
        LCDDR6  = LCDDR6|(n2<<4);
        LCDDR11 = LCDDR11|(n3<<4);
        LCDDR16 = LCDDR16|(n4<<4);
          break;
 case 4: LCDDR2  = LCDDR2&0xf0;
         LCDDR7  = LCDDR7&0xf0;
         LCDDR12 = LCDDR12&0xf0;
         LCDDR17 = LCDDR17&0xf0;
         LCDDR2  = LCDDR2|n1;
     LCDDR7 = LCDDR7|n2;
         LCDDR12  = LCDDR12|n3;
         LCDDR17 = LCDDR17|n4;
```

```
                break;
        case 5: LCDDR2  = LCDDR2&0x0f;
               LCDDR7  = LCDDR7&0x0f;
               LCDDR12 = LCDDR12&0x0f;
               LCDDR17 = LCDDR17&0x0f;
               LCDDR2  = LCDDR2|(n1<<4);
               LCDDR7 = LCDDR7|(n2<<4);
               LCDDR12  = LCDDR12|(n3<<4);
               LCDDR17 = LCDDR17|(n4<<4);
                break;
    }
}

int longSize(long i)
{
    if (i%10==i)
    {
        return 1;
    }
    else if (i%100==i)
    {
        return 2;
    }
    else if (i%1000==i)
    {
        return 3;
    }
    else if (i%10000==i)
    {
        return 4;
    }
    else if (i%100000==i)
    {
        return 5;
    }
    else
    return 6;
}


void writeLong(long i)
{
    int position = 0;
    char number = ’0’;
    int num = 0;
    int numsize;
    numsize = longSize(i);
    int j;
    int temp=1;

    for (position=0; position<numsize; position++)
    {
        switch (position)
         {
             case 5: num = (i%1000000-i%100000)/100000; number = num+’0’;break;
             case 4: num = (i%100000-i%10000)/10000; number = num+’0’;break;
             case 3: num = (i%10000-i%1000)/1000; number = num+’0’;break;
             case 2: num = (i%1000-i%100)/100; number = num+’0’;break;
             case 1: num = (i%100-i%10)/10; number = num+’0’;break;
             case 0: num = i%10; number = num+’0’;break;
         }
         switch (position)
         {
             case 0:writeChar(number,5);break;
             case 1:writeChar(number,4);break;
             case 2:writeChar(number,3);break;
             case 3:writeChar(number,2);break;
             case 4:writeChar(number,1);break;
             case 5:writeChar(number,0);break;
         }

    }
}

int is_prime(long i)
{
    int loop;
    loop = sqrt(i);
    int j = 0;
    int amount = 0;
    for (j=2; j<=loop+1; j++)
    {
         if (i%j == 0)
         {
             amount++;
         }
    }
    if (amount != 0)
    {
        return 0;
    }
    else
    return 1;
}

void primes()
{
    long i = 0;
    for (i=2; i<1000000; i++)
    {
        button();
        blink();
        if ((is_prime(i))==1)
        {
            writeLong(i);
        }
    }
}

void blink()
{
    if (TCNT1>=32767)
    {
        flag = 1-flag;
        if (flag==0)
        {
            LCDDR18 = 0x01;
            TCNT1 = 0;
        }
        else
        {
            LCDDR18 = 0x00;
            TCNT1 = 0;
         }
    }
}

void button()
{
    if ((PINB&0x80)!=0x80)
    {
        LCDDR13 = LCDDR13&0xfe;
    }
    else
    LCDDR13 = LCDDR13|0x01;
}

int main(void)
{
    TCNT1 = 0;
    LCDCRA  = 0x80;
    LCDCRB  = 0xB7;
    TCCR1B = 0x04;
    PORTB = PORTB|0x80;
    primes();
}
```