Slide 1

ECE/CE 3720: Embedded System Design

Chris J. Myers

Lecture 17: Output LEDs and LCDs

Slide 2

Interfacing Multiple LEDs

(See Figure 8.29)

Slide 3

Single LED Interface

(See Figures 8.30 and 8.31)

Slide 4

Output Param. for Open-Collector/Emitter Gates

(See Tables 8.4 and 8.5)

Slide 5

Typical Voltage/Current Response of a LED

(See Figure 8.32 and Table 8.6)

Slide 6

Calculating the Resistor Value

(See Figure 8.33)

Slide 7

Seven-Segment LED Interfaces (Common-Cathode)

(See Figure 8.34)

Slide 8

Seven-Segment LED Interfaces (Common-Anode)

(See Figure 8.35)

Slide 9

Scanned Seven-Segment LED Interface

(See Figures 8.36 and 8.37, Table 8.7)

Slide 10

Circuit Used to Scan a LED Interface

(See Figure 8.38)

Slide 11

Software for Scanned LED Display

```
// PB7-PB0 output, 7 bit pattern
// PC2-PC0 output, selects LED digit
unsigned char code[3]; // binary codes
static unsigned char select[3]={4,2,1};
unsigned int index;    // 0,1,2
#define OC5F 0x08
void ritual(void) {
asm(" sei");     // make atomic
   index=0;
   DDRC=0xFF;    // outputs
   TMSK1|=OC5F;  // Arm OC5
   TFLG1=OC5F;   // clear OC5F
   TOC5=TCNT+10000;
asm(" cli"); }
```

Slide 12

Software for Scanned LED Display

```
#pragma interrupt_handler TOC5handler()
void TOC5handler(void){
   TFLG1=OC5F;      // Acknowledge
   TOC5=TOC5+10000; // every 5 ms
   PORTC=select[index]; // which LED?
   PORTB=code[index];   // enable
   if(++index==3) index=0;}
```

Slide 13

Scanned LED Interface Using Decoder

(See Figure 8.39)

Slide 14

Software for Multiplexed LED Display

```
unsigned int global; // 12 bit packed BCD
const struct LED
{  unsigned char enable; // select
   unsigned char shift;  // bits to shift
   const struct LED *Next; };  // Link
#typedef const struct LED LEDType;
#typedef LEDType * LEDPtr;
LEDType LEDTab[3]={
{  0x04, 8, &LEDTab[1] },  // Most sig
{  0x02, 4, &LEDTab[2] },
{  0x01, 0, &LEDTab[0] }}; // least sig
LEDPtr Pt;   // Points to current digit
#define OC5F 0x08
```

Slide 15

Software for Multiplexed LED Display

```
#pragma interrupt_handler TOC5handler()
void TOC5handler(void){
    TFLG1=OC5F;        // Acknowledge
    TOC5=TOC5+10000;  // every 5 ms
    PORTB=(Pt->enable)+(global>>(pt->shift))<<4);
    Pt=Pt->Next; }
void ritual(void) {
asm(" sei");        // make atomic
    global=0;
    TMSK1=OC5F;     // Arm OC5
    Pt=&LEDTab[0];
    TFLG1=OC5F;     // clear OC5F */
    TOC5=TCNT+10000;
asm(" cli"); }
```

Slide 16

Extensions to Multiple Digits

- Two issues to consider as number of digits is increased:
  1. Scan frequency - for display to “look” continuous, each digit must be updated faster than 60 Hz.
     - interrupt rate = 60 Hz × #digits
  2. Duty cycle - this decreases as digits added, so must increase instantaneous current.
     - instantaneous current = desired current × #digits
- Ratio of maximum instantaneous current to desired LED current determines maximum number of digits.

Slide 17

Integrated IC Interface for LED Digits

(See Figure 8.40)

Slide 18

Data Timing of Integrated LED Controller

(See Figures 8.41 and 8.42)

Slide 19

Software for Integrated LED Display

```
// PD3/MOSI = MC14489 DATA IN
// PD4/SCLK = MC14489 CLOCK IN
// PD5 (simple output) = MC14489 ENABLE
void ritual(void) {
    DDRD |= 0x38;  // outputs to MC14489
    SPCR=0x50;
    PORTD|= 0x20; // ENABLE=1
    PORTD&= 0xDF; // ENABLE=0
    SPDR=0x01;    // hex format
    while(SPSR&0x80)==0){};
    PORTD|=0x20;} // ENABLE=1
```

Slide 20

Software for Integrated LED Display

```
void LEDout(unsigned char data[3]){
// 24 bit packed BCD
    PORTD &= 0xDF;    // ENABLE=0
    SPDR = data[2];   // send MSbyte
    while(SPSR&0x80)==0){};
    SPDR = data[1];   // send middle byte
    while(SPSR&0x80)==0){};
    SPDR = data[0];   // send LSbyte
    while(SPSR&0x80)==0){};
    PORTD |= 0x20;}   // ENABLE=1
```

Slide 21

LCD Fundamentals

- LCD display consume less power than LED displays.
- LCDs are more flexible in sizes and shapes, allowing for combination of numbers, letters, words, and graphics.
- Uses liquid-crystal material that behaves as a capacitor.
- While LED converts electric power to emitted optical power, LCD uses AC voltage to change light reflectivity.
- Light energy is supplied by room or separate back light.
- Display controlled by altering reflectivity of each segment.
- Disadvantage is that they have slow response time, but typically fast enough for human perception.

Slide 22

Basic Idea of a Liquid Crystal Interface

(See Figures 8.43, 8.44, and 8.45)

Slide 23

Direct Interface of a LCD

(See Figure 8.46)

Slide 24

Helper Function for a Simple LCD Display

```
void LCDOutDigit(unsigned char position,
                 unsigned char data) {
  // position is 0x80, 0x40, 0x20, or 0x10
  // and data is the BCD digit
  // set BCD digit on A-D inputs of the MC14543B
  PORTB=0x0F&data;
  // toggle one of the LD inputs high
  PORTB|=position;
  // LD=0, latch digit into MC14543B
  PORTB=0x0F&data;}
```

Slide 25

Software Interface for a Simple LCD Display

```
void LCDOutNum(unsigned int data){
  unsigned int digit,num,i;
  unsigned char pos;
  // data should be unsigned from 0 to 9999
  num=min(data,9999);
  pos=0x10;   // position of first digit (ones)
  for(i=0;i<4;i++){
    // next BCD digit 0 to 9
    digit=num%10; num=num/10;
    LCDOutDigit(pos,digit); pos=pos<<1;}}
```

Slide 26

Latched Interface of a LCD

(See Figure 8.47)

Slide 27

Artwork for 8-Segment LCD Digits

(See Figures 8.48 and 8.49)

Slide 28

LCD Timing

(See Figures 8.50 and 8.51)

Slide 29

LCD Timing (cont)

(See Figures 8.52 and 8.53)

Slide 30

Interface of a 48-Segment LCD Display

(See Figure 8.54)

Slide 31

Bit-Banged Interface to a Scanned LCD Display

```
void LCDOut (unsigned char *pt) {
  unsigned int i;
  unsigned char mask;
  for(i=0;i<6;i++){
    // look at bits 7,6,5,4,3,2,1,0
    for(mask=0x80;mask;mask=mask>>1){
      if((*pt)&mask) PORTB=1;
      else PORTB=0; // Serial data of the MC145000
      PORTB|=2; // toggle the serial clock high
      PORTB&=0xFD;}  // then low
      pt++; }}
```

Slide 32

SPI Interface to a Scanned LCD Display

```
// PD3/MOSI = MC145000 DATA IN
// PD4/SCLK = MC145000 CLOCK IN
void ritual(void) {
    DDRD |= 0x18;  // outputs to MC145000
    SPCR=0x50; }
void LCDout(unsigned char data[6]){
unsigned int j;
  for(j=5; j>=0 ; j--){
    SPDR = data[j];   // Msbyte first
    while(SPSR&0x80)==0){};}}
```

Slide 33

Interface of a HD44780 LCD Controller

(See Figure 8.55 and Table 8.8)

Slide 34

Private Functions for LCD Display

```
// 1 by 16 char LCD Display (HD44780)
#define LCDdata    1 // PB0=RS=1
#define LCDcsr     0 // PB0=RS=0
#define LCDread    2 // PB1=R/W=1
#define LCDwrite   0 // PB1=R/W=0
#define LCDenable  4 // PB2=E=1
#define LCDdisable 0 // PB2=E=0
void LCDcycwait(unsigned short cycles){
    TOC5=TCNT+cycles; // 500ns cycles
    TFLG1 = 0x08;     // clear C5F
    while((TFLG1&0x08)==0){};}
```

Slide 35

Public Functions for LCD Display

```
void LCDinit(void){
    DDRC=0xFF;
    LCDputcsr(0x06);
// I/D=1 Increment, S=0 nodisplayshift
    LCDputcsr(0x0C);   // D=1 displayon,
// C=0 cursoroff, B=0 blinkoff
    LCDputcsr(0x14);   /
/ S/C=0 cursormove, R/L=0 shiftright
    LCDputcsr(0x30);
// DL=1 8bit, N=0 1 line, F=0 5by7dots
    LCDclear();}    // clear display
```

Slide 36

Public Functions for LCD Display

```
void LCDputchar(unsigned short letter){
// letter is ASCII code
    PORTC=letter;
    PORTB=LCDdisable+LCDwrite+LCDdata;
    PORTB=LCDenable+LCDwrite+LCDdata;
// E goes 0,1
    PORTB=LCDdisable+LCDwrite+LCDdata;
// E goes 1,0
    LCDcycwait(80);} // 40 us wait
```

Slide 37

Public Functions for LCD Display

```
void LCDputcsr(unsigned short command){
    PORTC=command;
    PORTB=LCDdisable+LCDwrite+LCDcsr;
    PORTB=LCDenable+LCDwrite+LCDcsr;
// E goes 0,1
    PORTB=LCDdisable+LCDwrite+LCDcsr;
// E goes 1,0
    LCDcycwait(80);} // 40 us wait
void LCDclear(void){
    LCDputcsr(0x01);   // Clear Display
    LCDcycwait(3280);  // 1.64ms wait
    LCDputcsr(0x02);   // Cursor to home
    LCDcycwait(3280);} // 1.64ms wait
```