Elevator

```
#include <lpc214x.h>
#define LED_OFF (IO0SET = 1U << 31)
#define LED_ON (IO0CLR = 1U << 31)

void delay_ms(unsigned int j);
void elevator_run(void);

int main()
{
   // Set P0.16 to P0.23 and P0.31 as outputs
   IO0DIR = (1U << 31) | (0xFF << 16);
   // Set P1.24 as output
   IO1DIR = 1U << 24;

   // Indicate that the program is running
   LED_ON;

   // Run the elevator control system
   elevator_run();

   // Main loop
   while(1);
}
```

```
void elevator_run(void)
{
   int i, val;
   unsigned int counter;

   // Enable elevator section in the application board: 0 to enable
   IO1CLR = 1U << 24;
   // Set the elevator LED for the ground floor
   IO0CLR = 0x000F0000;

   do {
      // Clear all the latches *CLR
      IO0CLR = 0x00F00000;
      IO0SET = 0x00F00000;

      // Waiting for floor key
      do {
         // Wait for any lift/elevator key press
         counter = (IO1PIN >> 16) & 0x0000000F;
      } while (counter == 0x0F);

      if (counter == 0x0E) val = 3; // 1110 - floor 1 key pressed
      else if (counter == 0x0D) val = 6; // 1101 - floor 2 key pressed
      else if (counter == 0x0B) val = 8; // 1011 - floor 3 key pressed
      else if (counter == 0x07) val = 10; // 0111- floor 4 key pressed
      else val = 0; // Default value for safety
```

```
        // Elevator movement - UP
        for (i = 0; i < val; i++) {
            IO0CLR = 0x000F0000;
            IO0SET = (1U << (i + 16));
            delay_ms(250);
        }



        // Elevator movement - DOWN
        for (i = val - 1; i >= 0; i--) {
            IO0CLR = 0x000F0000;
            IO0SET = (1U << (i + 16));
            delay_ms(250);
        }
    } while (1);
}

void delay_ms(unsigned int j)
{
    unsigned int x, i;
    for (i = 0; i < j; i++) {
        for (x = 0; x < 1000;x++);
}
}
```

7 Segment

```
#include <lpc214x.h>
#define LED_OFF (IO0SET = 1U << 31)
#define LED_ON (IO0CLR = 1U << 31)
#define PLOCK 0x00000400

void delay_ms(unsigned int j);
void SystemInit(void);
unsigned char getAlphaCode(unsigned char alphachar);
void alphadisp7SEG(char *buf);



int main() {
   IO0DIR |= (1U << 31) | (1U << 19) | (1U << 20) | (1U << 30); // Set pins as
outputs
   LED_ON; // Indicate the program is running
   SystemInit();
   while(1) {
      alphadisp7SEG("fire ");
      delay_ms(500);
      alphadisp7SEG("help ");
      delay_ms(500);
   }
}
```

```
unsigned char getAlphaCode(unsigned char alphachar) {
    switch (alphachar) {
        case 'f': return 0x8E;
        case 'i': return 0xF9;
        case 'r': return 0xCE;
        case 'e': return 0x86;
        case 'h': return 0x89;
        case 'l': return 0xC7;
        case 'p': return 0x8C;
        case ' ': return 0xFF;
        default: return 0xFF;
    }
}
```

```
void alphadisp7SEG(char *buf) {
    unsigned char i, j, seg7_data, temp;
    for (i = 0; i < 5; i++) {
        seg7_data = getAlphaCode(buf[i]);
        for (j = 0; j < 8; j++) {
```

```
            temp = seg7_data & 0x80;
            if (temp == 0x80)
               IOSET0 |= 1U << 19;
            else
               IOCLR0 |= 1U << 19;

            IOSET0 |= 1U << 20;
            delay_ms(1);
            IOCLR0 |= 1U << 20;
            seg7_data <<= 1;
         }
      }

      IOSET0 |= 1U << 30;
      delay_ms(1);
      IOCLR0 |= 1U << 30;
}

void SystemInit(void) {
      PLL0CON = 0x01;
      PLL0CFG = 0x24;
      PLL0FEED = 0xAA;
      PLL0FEED = 0x55;
      while (!(PLL0STAT & PLOCK)) { }
      PLL0CON = 0x03;
      PLL0FEED = 0xAA;
```

```
    PLL0FEED = 0x55;
    VPBDIV = 0x01;
}

void delay_ms(unsigned int j) {
    unsigned int x, i;
    for (i = 0; i < j; i++) {
        for (x = 0; x < 10000; x++);
    }
}
```

Stepper Motor

```
#include <lpc214x.h>

#define LED_ON (IO0CLR = 1U << 31)
#define LED_OFF (IO0SET = 1U << 31)
#define PLOCK 0x00000400

void delay_ms(unsigned int j);
void SystemInit(void);

int main() {
    unsigned int no_of_steps_clk = 100, no_of_steps_aclk = 100;

    IO0DIR |= (1U << 31) | 0x00FF0000; // Set P0.16 to P0.23 as outputs
    LED_ON;
```

```
delay_ms(500);
LED_OFF;

SystemInit();

// Clockwise rotation
while (no_of_steps_clk--) {
   IO0CLR = 0x000F0000;
   IO0SET = 0x00010000; // First winding
   delay_ms(10);

   IO0CLR = 0x000F0000;
   IO0SET = 0x00020000; // Second winding
   delay_ms(10);

   IO0CLR = 0x000F0000;
   IO0SET = 0x00040000; // Third winding
   delay_ms(10);

   IO0CLR = 0x000F0000;
   IO0SET = 0x00080000; // Fourth winding
   delay_ms(10);
}

// Anti-clockwise rotation
while (no_of_steps_aclk--) {
```

```
        IO0CLR = 0x000F0000;
        IO0SET = 0x00080000; // Fourth winding
        delay_ms(10);

        IO0CLR = 0x000F0000;
        IO0SET = 0x00040000; // Third winding
        delay_ms(10);

        IO0CLR = 0x000F0000;
        IO0SET = 0x00020000; // Second winding
        delay_ms(10);

        IO0CLR = 0x000F0000;
        IO0SET = 0x00010000; // First winding
        delay_ms(10);
    }

    IO0CLR = 0x00FF0000; // Turn off all motor windings
    while (1); // Keep the program running indefinitely
}

void delay_ms(unsigned int j) {
    for (unsigned int i = 0; i < j; i++) {
        for (unsigned int x = 0; x < 10000; x++);
    }
}
```

```
void SystemInit(void) {
   PLL0CON = 0x01;
   PLL0CFG = 0x24;
   PLL0FEED = 0xAA;
   PLL0FEED = 0x55;
   while (!(PLL0STAT & PLOCK)) { }
   PLL0CON = 0x03;
   PLL0FEED = 0xAA;
   PLL0FEED = 0x55;
   VPBDIV = 0x01;
}
```

DAC

```
#include <lpc214x.h>
#define PLOCK 0x00000400
#define LED_OFF (IO0SET = 1U << 31)
#define LED_ON (IO0CLR = 1U << 31)
#define SW2 (!(IO0PIN & (1U << 14)))
#define SW3 (!(IO0PIN & (1U << 15)))
#define SW4 (!(IO1PIN & (1U << 18)))
#define SW5 (!(IO1PIN & (1U << 19)))
#define SW6 (!(IO1PIN & (1U << 20)))

void SystemInit(void);
void delay_ms(unsigned int j);
```

```
short int sine_table[] = {
   512+0, 512+53, 512+106, 512+158, 512+208, 512+256, 512+300, 512+342,
512+380, 512+413,
   512+442, 512+467, 512+486, 512+503, 512+510, 512+511,
   512+510, 512+503, 512+486, 512+467, 512+442, 512+413, 512+380,
512+342, 512+300, 512+256, 512+208, 512+158, 512+106, 512+53, 512+0,
   512-53, 512-106, 512-158, 512-208, 512-256, 512-300, 512-342, 512-380,
512-413, 512-442, 512-467, 512-486, 512-503, 512-510, 512-511,
   512-510, 512-503, 512-486, 512-467, 512-442, 512-413, 512-380, 512-342,
512-300, 512-256, 512-208, 512-158, 512-106, 512-53
};
short int sine_rect_table[] = {
   512+0, 512+53, 512+106, 512+158, 512+208, 512+256, 512+300, 512+342,
512+380, 512+413,
   512+442, 512+467, 512+486, 512+503, 512+510, 512+511,
   512+510, 512+503, 512+486, 512+467, 512+442, 512+413, 512+380,
512+342, 512+300, 512+256, 512+208, 512+158, 512+106, 512+53, 512+0
};

int main() {
   short int value;
   unsigned int i = 0;

   SystemInit();
   PINSEL1 |= 0x00080000; // P0.25 as DAC output
   IO0DIR |= (1U << 31) | 0x00FF0000; // P0.16 to P0.23 as outputs
```

```
while (1) {
    if (SW2) { // Sine wave
        for (i = 0; i < 60; i++) {
            value = sine_table[i];
            DACR = (1 << 16) | (value << 6);
            delay_ms(1);
        }
    } else if (SW3) { // Rectified sine wave
        for (i = 0; i < 30; i++) {
            value = sine_rect_table[i];
            DACR = (1 << 16) | (value << 6);
            delay_ms(1);
        }
    } else if (SW4) { // Triangular wave
        for (value = 0; value < 1024; value++) {
            DACR = (1 << 16) | (value << 6);
        }
        for (value = 1023; value >= 0; value--) {
            DACR = (1 << 16) | (value << 6);
        }
    } else if (SW5) { // Sawtooth wave
        for (value = 0; value < 1024; value++) {
            DACR = (1 << 16) | (value << 6);
        }
    } else if (SW6) { // Square wave
        value = 1023;
```

```
            DACR = (1 << 16) | (value << 6);
            delay_ms(1);
            value = 0;
            DACR = (1 << 16) | (value << 6);
            delay_ms(1);
        } else { // Default to 3.3V DC
            value = 1023;
            DACR = (1 << 16) | (value << 6);
        }
    }
}

void SystemInit(void) {
    PLL0CON = 0x01;
    PLL0CFG = 0x24;
    PLL0FEED = 0xAA;
    PLL0FEED = 0x55;
    while (!(PLL0STAT & PLOCK)) {}
    PLL0CON = 0x03;
    PLL0FEED = 0xAA;
    PLL0FEED = 0x55;
}
void delay_ms(unsigned int j) {
    for (unsigned int i = 0; i < j; i++) {
        for (unsigned int x = 0; x < 10000; x++);
    }
```

}

Keyboard

```
#include <lpc214x.h>
#define LED_OFF (IO0SET = 1U << 31)
#define LED_ON  (IO0CLR = 1U << 31)
#define PLOCK 0x00000400
#define ROW0 (1U << 16)
#define ROW1 (1U << 17)
#define ROW2 (1U << 18)
#define ROW3 (1U << 19)

#define COL0 (IO1PIN & (1U << 19))
#define COL1 (IO1PIN & (1U << 18))
#define COL2 (IO1PIN & (1U << 17))
#define COL3 (IO1PIN & (1U << 16))

unsigned char lookup_table[4][4] = {
   {'0', '1', '2', '3'},
   {'4', '5', '6', '7'},
   {'8', '9', 'a', 'b'},
   {'c', 'd', 'e', 'f'}
};

void uart_init(void) {
   PINSEL0 |= 0x00000005; // P0.0 & P0.1 are TXD0 & RXD0
```

```
    U0LCR = 0x83;   // 8 bits, no Parity, 1 Stop bit
    U0DLM = 0; U0DLL = 8; // 115200 baud rate
    U0LCR = 0x03;   // DLAB = 0
    U0FCR = 0x07;   // Enable and reset TX and RX FIFO
}
void SystemInit(void) {
    PLL0CON = 0x01;
    PLL0CFG = 0x24;
    PLL0FEED = 0xAA;
    PLL0FEED = 0x55;
    while (!(PLL0STAT & PLOCK));
    PLL0CON = 0x03;
    PLL0FEED = 0xAA;
    PLL0FEED = 0x55;
    VPBDIV = 0x01; // PCLK same as CCLK (60MHz)
}

void delay_ms(unsigned int j) {
    unsigned int x, i;
    for (i = 0; i < j; i++) {
        for (x = 0; x < 10000;x++);
}
}

int main() {
    unsigned char rowsel, colsel;
```

```
    IO0DIR |= 1U << 31 | 0x00FF0000; // Set P0.16 to P0.23 as outputs for LEDs and row selection
    //IO1DIR &= ~(0xF << 16); // Set P1.16 to P1.19 as inputs for columns

    SystemInit();
    uart_init();

    LED_ON; // Turn on LED for testing
    delay_ms(500);
    LED_OFF; // Turn off LED
    delay_ms(500);

    while (1) {
        for (rowsel = 0; rowsel < 4; rowsel++) {
            IO0SET = 0x000F0000; // Disable all rows
            switch (rowsel) {
                case 0: IO0CLR = ROW0; break;
                case 1: IO0CLR = ROW1; break;
                case 2: IO0CLR = ROW2; break;
                case 3: IO0CLR = ROW3; break;
            }

            delay_ms(1); // Debounce delay

            if (COL0 == 0) colsel = 0;
            else if (COL1 == 0) colsel = 1;
            else if (COL2 == 0) colsel = 2;
```

```
        else if (COL3 == 0) colsel = 3;
        else continue;

        delay_ms(50); // Debounce delay
        while (COL0 == 0 || COL1 == 0 || COL2 == 0 || COL3 == 0); // Wait for
key release
        delay_ms(50); // Additional debounce delay

        IO0SET = 0x000F0000; // Disable all rows

        U0THR = lookup_table[rowsel][colsel]; // Send key over UART
        break; // Exit the row scan loop
      }
   }
}
```

DC Motor

```
#include <lpc214x.h>

#define LED_OFF (IO0SET = 1U << 31)
#define LED_ON  (IO0CLR = 1U << 31)
#define PLOCK 0x00000400

void delay_ms(unsigned int j);
void SystemInit(void);
```

```
void runDCMotor(int direction, int dutycycle);
unsigned int adc(int no, int ch);

int main() {
   int dig_val;

   IO0DIR |= (1U << 31) | (1U << 30); // Set P0.31 (LED) and P0.30 (Motor Control) as outputs
   LED_ON;
   delay_ms(500);
   LED_OFF;

   SystemInit();

   while (1) {
      dig_val = adc(1, 2) / 10;
      if (dig_val > 100) dig_val = 100;
      runDCMotor(1, dig_val); // Run the motor with the calculated duty cycle
   }
}

void runDCMotor(int direction, int dutycycle) {
   IO0DIR |= (1U << 28); // Set P0.28 as output for direction control
   PINSEL0 |= (2 << 18); // Set P0.9 as PWM6 output

   if (direction == 1)
      IO0SET = (1 << 28); // Set P0.28 high for anti-clockwise
```

```
    else
        IO0CLR = (1 << 28); // Set P0.28 low for clockwise

    PWMPCR = (1 << 14); // Enable PWM6
    PWMMR0 = 1000; // Set PWM period
    PWMMR6 = (1000 * dutycycle) / 100; // Set PWM duty cycle
    PWMTCR = 0x09; // Enable PWM and the timer
    PWMLER = 0x40; // Load PWM match register 6 value
}

unsigned int adc(int no, int ch) {
    unsigned int val;
    AD1CR = 0x00200600 | (1 << ch); // Select channel and set clock
    AD1CR |= (1 << 24); // Start conversion
    while (!(AD1GDR & (1U << 31))); // Wait for conversion to complete
    val = AD1GDR;

    return (val >> 6) & 0x03FF; // Return the 10-bit ADC value
}

void SystemInit(void) {
    PLL0CON = 0x01;
    PLL0CFG = 0x24;
    PLL0FEED = 0xAA;
    PLL0FEED = 0x55;
    while (!(PLL0STAT & PLOCK));
```

```
    PLL0CON = 0x03;
    PLL0FEED = 0xAA;
    PLL0FEED = 0x55;
    VPBDIV = 0x01; // PCLK is same as CCLK (60 MHz)
}

void delay_ms(unsigned int j) {
    unsigned int x, i;
    for (i = 0; i < j; i++) {
        for (x = 0; x < 10000; x++);
    }
}
```