UNIT-II (Only programs)

1. LED interfacing programs.
2. Switch interfacing.
3. Seven Segment Display.
4. LCD.
5. Matrix Keypad.
6. Stepper motor.
7. DC motor.
8. Relay.
9. Opto-isolators.

} Digital Interfacing Programs

1. ADC channels.
2. LDR & Temperature sensors.
3. DAC for waveform generation.

} Analog interfacing Programs.

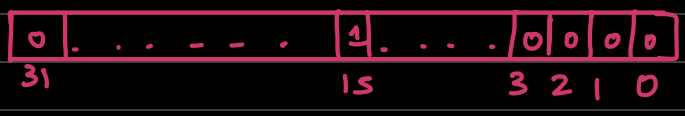
Digital Interfacing:

Programs on LED, switch interfacing:

1. Interface LED to a particular pin to on & off.
2. ~~~~~ ~~~~~ ~ & write a pgm to toggle.
3. generate asymmetric square wave at lowest 4 pins of port 0.
4. Interface LED & push button (press to on) switch. LED turns on/off when switch is pressed.
5. Interface 3 LEDs (red, green, yellow) ⇒ Traffic lights system implementation.
6. Blinking an LED on P1.24 to P1.31.

Note: there are many ways to configure a pin (one way was shown in last ques)

Ex: How to set P0.15?



We use port 0 so, we use IODIR0 and we shift '1' 15 times

```
IODIR0 = IODIR0 | (1<<15);
IOSET0 |= (1<<15);
```

We use 'OR' because we do not want all bits to be affected.
Then we set 1 on P0.15.

Ex: How to clear P1.16 ?

```
IODIR1 = IODIR1 | (1<<16);
IOCLR1 |= (1<<16);
```

Ex: set lower sixteen GPIO pins of P0 to 1?

```
IODIR0 = 0x0000FFFF
IOSET0 = 0x0000FFFF
```

F = 1111 hexadecimal value of F

0000 → all other pins are 0
FFFF → lower 16 are 1

Ex: interface an LED to P0.10 & turn ON & OFF the LED?

```
#include <lpc214x.h>
int main()
{
unsigned int x;
IODIR0 = 0xFFFFFFFF;   // make all the pins as outputs.
for(;;)                // ∞ loop
{
IOSET0 = 1 << 10;      // set port P0.10
for(x=0; x<30000; x++); // delay for ON time
IOCLR0 = 1 << 10;      // clear port P0.10
for(x=0; x<40000; x++); // delay for OFF time.
}
}
```

Ex: interface an LED to P0.31 & write a prog to toggle.

```
int main()
{
IODIR0 = 0x80000000;
do {
IOSET0 = 0x80000000;
IOCLR0 = 0x80000000;
} while(1);
}
void delay_ms(unsigned int val);   (above main)
delay_ms(10);
delay_ms(10);
```

```
void delay_ms (unsigned int val)
{
int i;
while (val--)
 for (i=0; i<1000; i++);
}
```

Ex: generate the asymmetric square wave at the lowest four pins of Port0.


```
#~~
int main (void)
{
unsigned int x;
IODIR0 = 0xFFFFFFFF;   // make all pins outputs.
for (;;)               // ∞ loop.
{
IOSET0 = 0x0000000F;   // sets 1st four bits as high. "00000000 00000000 00000000 00001111" high
for (x=0; x<10000; x++); // delay loop, pins remain high. (P0-P3)
IOCLR0 = 0x0000000F;   // clears 1st four bits
for (x=0; x<20000; x++); // delay loop, pins remain low.
 }
}
```

Ex: Interface an LED and Push Button (press to on) Switch. ⭐ Imp

⇒ Assume P0.31 connected to LED. Configure P0.31 as GPIO O/P, as LED is O/P device.
assume P0.14 connected to switch. Configure P0.14 as GPIO I/P, as switch is I/P device.

⇒ LED off ⇒ 1
LED on ⇒ 0
} LED is common anode

⇒ switch pressed, P0.14 receives 0
switch not pressed P0.14 receives 1

```
#include <lpc214x.h>
void delay (unsigned int count);
int main (void)
{                                  → 0x80000000
IODIR0 |= (1<<31);                 // configure P0.31 as O/P (for LED)
IODIR0 &= ~(1<<14);                // Configure P0.14 as I/P (for switch) by clearing P0.14
while (1)
{
if ((IOPIN0 & (1<<14)) ==0)        // check if switch is pressed (P0.14 = 0)
{
 IOCLR0 = (1<<31);                 // turn ON LED (P0.31 = 0)
```

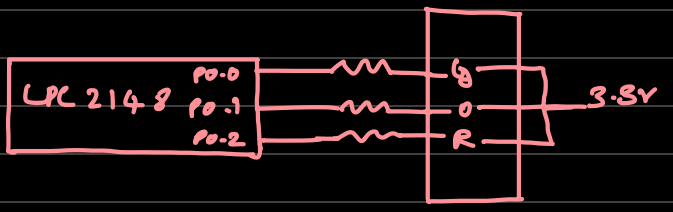
```
delay (100000);
IOSET = (1<<31);          // turn off LED (P0.31 set to 1)
delay (100000);
}
else                      // turn off LED if switch not pressed.
{
IOSET = (1<<31);          // led off (P0.31 set to 1)
}
}
}

Void delay (unsigned int count)
{
unsigned int i, x;
for(i=0; i<count ; i++) {
  for (x=0 ; x<1000; x++);
} }
```

} why 2 loops?

Note : 'OR' and 'AND' are used to not affect the other bits in the register.
To set a bit, we use 'OR' operation. To set O/P high without altering other bits
To Clr a bit, we use 'AND' op with 'NOT' operator. To set a pin low without altering other bits.

Ex: Interface 3 LEDs (Red, Yellow, green) to LPC2148 & write embedded C Program to simulate traffic light system.

⇒ P0.0 ⇒ o/p for RED LED.
P0.1 ⇒ o/p for yellow LED
P0.2 ⇒ o/p for green LED.




```
#include <lpc214x.h>
void delay (unsigned int count)
int main (void)
{
IODIR0 |= (1<<0);         // P0.0 o/p for RED
IODIR0 |= (1<<1);         // P0.1 o/p for YELLOW
IODIR0 |= (1<<2);         // P0.2 o/p for GREEN.
while (1)
{
IOSET0 = (1<<2);          // green on, others off
IOCLR0 = (1<<1);
IOCLR0 = (1<<0);
delay (300000);
IOCLR0 = (1<<2);          // yellow on, others off
IOSET0 = (1<<1);
IOCLR0 = (1<<0);
delay (100000);
```

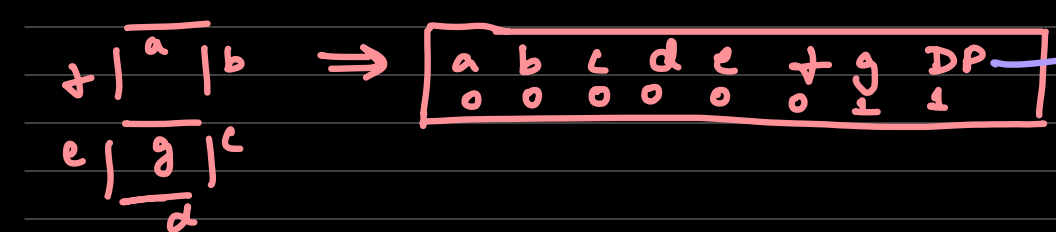
```
IOCLR0 = (1<<2);          // red on, others off.
IOSET0 = (1<<1);
IOCLR0 = (1<<0);
delay (300000);
}
}
Void delay (unsigned int count)
{
unsigned int i, x;
for(i=0; i<count; i++)
{
 for (x=0; x<count; x++)
 }
 }
```

Q: Blinking LED from P1.24 to P1.31

```
#include <lpc214x.h>
void delay (unsigned int count)
int main ()
{
IODIR1 |= (1<<24);        (or)   IODIR1 |= 0xFF000000;   (31-24)
while (1)
{
IOSET1 = 0xFF000000;
delay (1000);
IOCLR1 = 0xFF000000;
delay (1000);
}
void delay (unsigned int count)
{
unsigned int i, x;
for(i=0; i<count; i++)
{
 for (x=0; x<1000; x++)
  }
 }
```

Seven Segment Interface : Imp

⇒ Common anode seven segment display.



There are two types of seven segment displays:

① common anode (all 0s, g=1) ⇒ active low

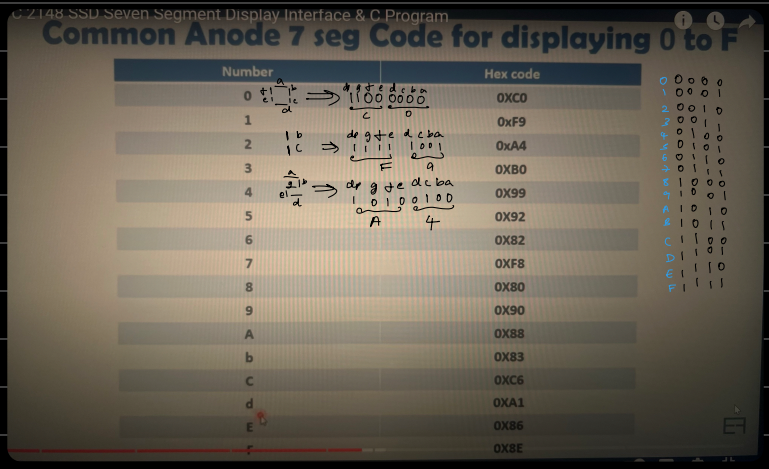
② common cathode (all 1s, g=0) ⇒ active high

Ex: To generate 3; DP g f e d c b a = 1 0 1 1 0 0 0 0 } 8 bits required

```
#include <lpc21xx.h>
#define LEDOFF (IOSET0 = 1U<<8)
#define LEDON (IOCLR0 = 1U<<8)
Void delay (unsigned int count);
Unsigned char seven segment codes [10] = {0xC0, 0xF9 ..... all values}; // Seven segment codes.

int main() {
unsigned char num;
IODIR0 |= 1U <<8 | 0x000000FF ;        8 pins
// we have to make P0.0 to P0.7 as O/P pins.
LEDON;
num=1; }

// to display 1 on the 7 segment display
                0000 0001

IOCLR0 = ~(seven segment code [num]);
while(1);
}

// to turn the lights of 1 on, we can use IOSET but it is not possible as we need a pattern to turn off the unwanted letters/numbers too. So, we do conversion from decimal to seven segment code.

Void delay (unsigned int count)
{
int i;
for (i=0; i<count; i++);
}
```

↳ 2 for loops

// Lookup table:

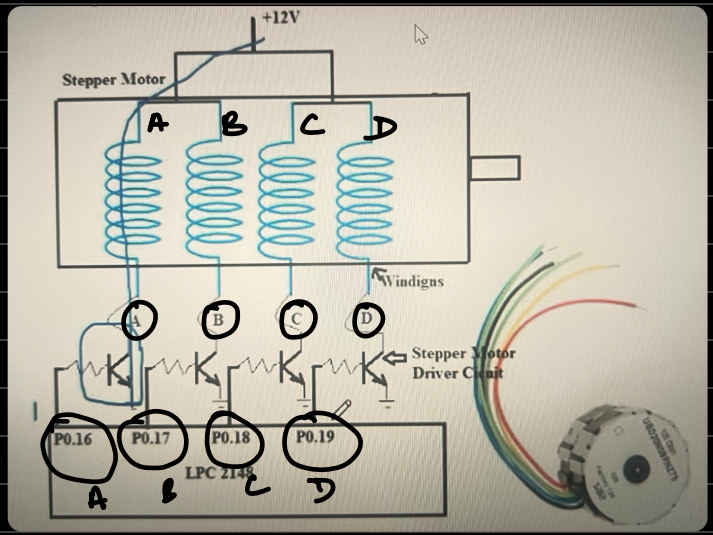


Stepper Motor interfacing: Imp

⇒ It is a mechanical device that is digitally controlled to rotate the motor in steps (each step can be controlled).

⇒ By moving by a fixed amt of degrees.

```
#~
void delay( unsigned int count);
int main()
{
IODIR0 |= (1U<<0 | 1U<<1 | 1U <<2);      // we have used 16, 17, 18, 19 as the o/p pins.
while(1)
{
IODIR |= 0x000F0000;                      // Make P.16 to P.23 as o/p & assign no. of anticlockwise & clockwise steps.
int clockwisesteps = 100;
int anticlockwisesteps =100;
while(1)
{
do                                        // There are 4 windings. Each time we have to energize 1 winding & turn off the other 3 windings
  {
  IO0CLR0 = 0x000F0000;
  IO SET0 = 0x00010000;                   } winding A
  delay(10);
   if(--clockwisesteps ==0)
    break;
  IO0CLR0 = 0x000F0000;
  IO SET0 = 0x00020000;                   } winding B
  delay(10);
   if(--clockwisesteps ==0)
    break;
  IO0CLR0 = 0x000F0000;
  IO SET0 = 0x00040000;                   } winding C
  delay(10);
   if(--clockwisesteps ==0)
    break;
  IO0CLR0 = 0x000F0000;
  IO SET0 = 0x00080000;                   } winding D
  delay(10);
   if(--clockwisesteps ==0)
    break;
}
```



//clockwise step by step, until each winding reaches 0 (stops rotating).

```
while (1);
do
{
    IO0CLR0 = 0x000F0000;
    IO0SET0 = 0x00080000;
    delay (10);
    if (--anticlockwisesteps == 0)
        break;
    IO0CLR0 = 0x000F0000;
    IO0SET0 = 0x00040000;
    delay (10);
    if (--anticlockwisesteps == 0)
        break;
    IO0CLR0 = 0x000F0000;
    IO0SET0 = 0x00020000;
    delay (10);
    if (--anticlockwisesteps == 0)
        break;
    IO0CLR0 = 0x000F0000;
    IO0SET0 = 0x00010000;
    delay (10);
    if (--anticlockwisesteps == 0)
        break;
}
while (1);
}
}
```

// anticlockwise steps.

winding D

winding C

winding B

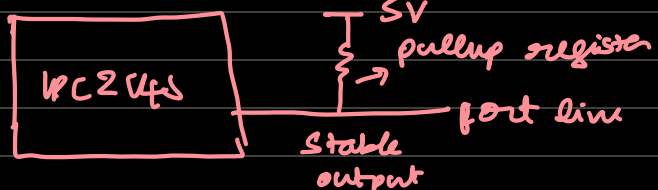
winding A

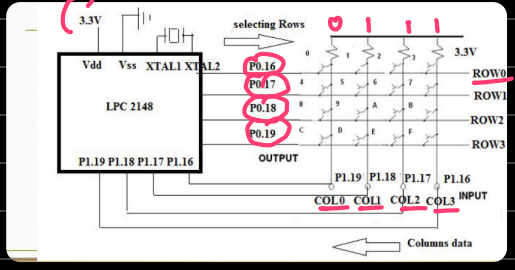
⇒ Total no. of steps for 1 revolution = 200 steps (200 teeth shaft.

⇒ step angle = $\frac{360^\circ}{200} = 1.8^\circ$

⇒ 2 phases, 4 windings stepper motor is used

Matrix keyboard:

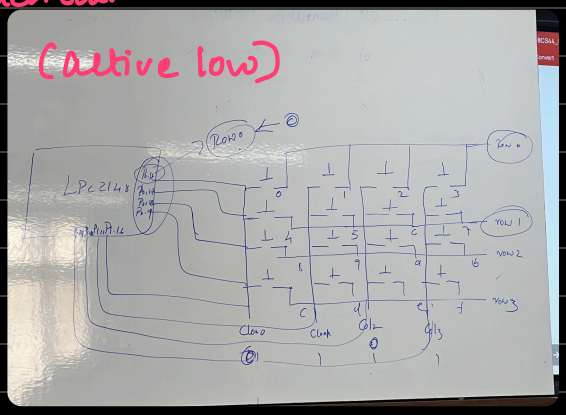




⇒ 4×4 matrix keyboard (16 keys) (lookup table)

⇒ when key is pressed port line is grounded. ⇒ 0
when key is not pressed, we receive 1

// 4 x 4 keyboard
// columns & rows are pulled up to 5V, if not we receive 1 on columns.
// method: check for '0' in each corresponding col of each row.
// rows: P0.16 P0.17 18 19 (port 0)
// col: P1.16 P1.17 18 9 (port 1)




```
#~~~~~
#define LEDOFF (IOSET0 = 1U<<31)
#~~~~~ ON (IOCLR = 1U<<31)
#define COL0 (IOPIN1 & 1<<16)
#~~~~~~~~~~~~ ~<<17)
#~~~~~ ~~~~~~~~~~~~ <<18)
#~~~~~ ~~~~~~~~~~~~ <<19)
unsigned char lookuptable[4][4] = {{'0','1','2','3'
                                   '4','5','6','7'
                                   '8','9','A','B'
                                   'C','D','E','F'}};
UART_init(); //ini UART0 port.

int main()
{

IODIR0 |= 1U<<31 | 0x00FF0000;  //Set 16 to 23 high
 {
 while(1)
 {
 rowsel = 0;
 IOSET0 = 0x000F0000;
 IOCLR0 = 1<<16;
 if (col0==0)
    colsel = 0;
    break;
 if(col1 ==0)
    colsel=1;
    break;
 if (col2==0)
    colsel=2;
    break;
 if(col3==0)
    colsel = 3;
    break;
```

Check for key press in row 0

→ sets row 0 ⇒ low
→ other rows ⇒ high

checks keypress in each col. If it is, breaks out of loop & stores true col no.

```
rowsel =1;
IOSET0 = 0x000F0000;
IOCLR0 = 1<<17;
if (col0 ==0)
   colsel =0;
   break;
if(col1 ==0)
   colsel=1;
   break;
if (col2 ==0)
   colsel=2;
   break;
if(col3 ==0)
   colsel =3;
   break;
rowsel =2;
IOSET0 = 0x000F0000;
IOCLR0 = 1<<18;
if (col0 ==0)
   colsel =0;
   break;
if(col1 ==0)
   colsel=1;
   break;
if (col2 ==0)
   colsel=2;
   break;
if(col3 ==0)
   colsel =3;
   break;
rowsel =3;
IOSET0 = 0x000F0000;
IOCLR0 = 1<<19;
if (col0 ==0)
   colsel =0;
   break;
if(col1 ==0)
   colsel=1;
   break;
if (col2 ==0)
   colsel=2;
   break;
if(col3 ==0) 3;
```

row 1

row 2

row 3

```
col_sel = 3;
break;
}
delay(50);                                  // delay for debouncing
while (col0 == 0 || col1 == 0 || col2 == 0);   // waits until all col are high, indicating key release.
LOSET0 = 0x00DF000;                         // sets all rows high to disable them.
UDTHR = lookuptable[row_sel-1][col_sel];    // sends char corresponding pressed key via UART.
}
while(1);
}
```

(after detecting keypress, code waits to debounce the key)

(+ delay fun^)

(similarly for 3x3, use 3 pins : 16, 17, 18, 3 rows & 3 col)

LCD interfacing:

⇒ There are many types of LCDs.

⇒ 16 X 2 LCD displays 16 characters per line & there are 2 lines / rows.

⇒ has two registers: Command reg (stores command instructions given to the data reg.