Chapter 12

Parallel Interfacing

Expected Outcomes

- Identify the internal registers of MC6821
- Design the hardware interface for various I/O devices using MC6821
- Develop and write codes using MC6821

Parallel Interface/Timer

- I/O interface using a simple I/O device such as latch or buffer has its drawbacks
- One of them is the circuit design must be reconstructed if a user decide to change its I/O devices
- A dedicated peripheral such as the 68230 (PI/T - Parallel Interface/Timer)
  - Its primary function is to provide parallel interface
  - Its secondary function is a programmable timer
- It provides 4 modes of operation with various handshaking and buffering capabilities
  - Unidirectional 8-bits
  - Unidirectional 16-bits
  - Bidirectional 8-bits
  - Bidirectional 16-bits

Parallel Interface/Timer

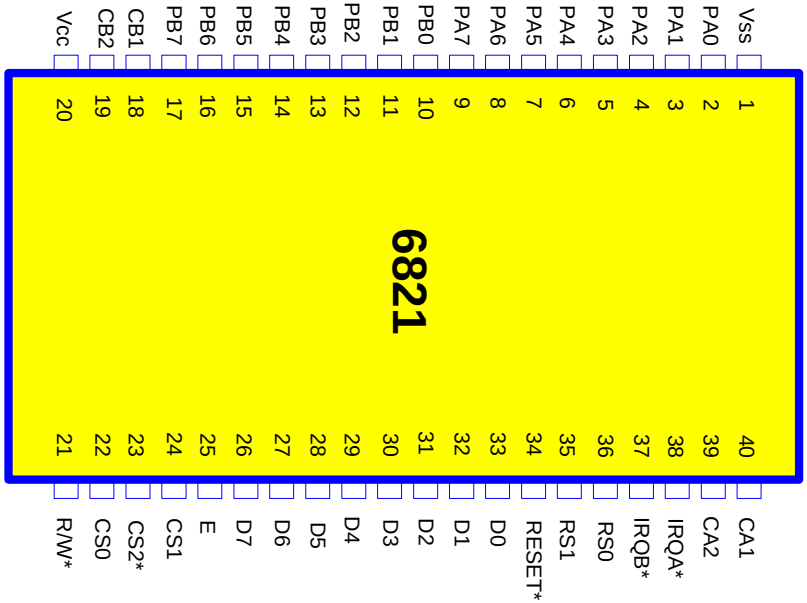
- The programmable timer provides a variety of services
  - Periodic interrupt generation
  - Square wave generator
  - Interrupt after time-out
  - Elapse time measurement
  - System watchdog
- Since, it is part of 68xxx family, the PI/T is expensive due to its capability to provide wider functions
- A simple 8-bit family can be used in certain cases to provide parallel I/O interface
- PIA 6821 peripheral interface adapter is introduced to give flexibility in I/O system

PIA 6821

- PIA was initially designed for use in 6800-based system
- However, it is widely used in 68000-based system due its flexibility and cost
- Since it is part of 6800 family, additional pin connections are needed to operate in asynchronous mode
  - The port A and B may be programmed individually to be an input or an output – allowing the software to adjust to a new system requirement
  - Additional signal lines may be programmed to allow interrupts and strobe signals for each port
  - Internal register to allow storage of data temporary thus allowing to interface with a slower devices such as printer

PIA 6821

- PIA has two port with each port may drive two TTL loads
- Each individual signal line (PA0 through PA7 and PB0 through PB7) can be programmed as an input or an output



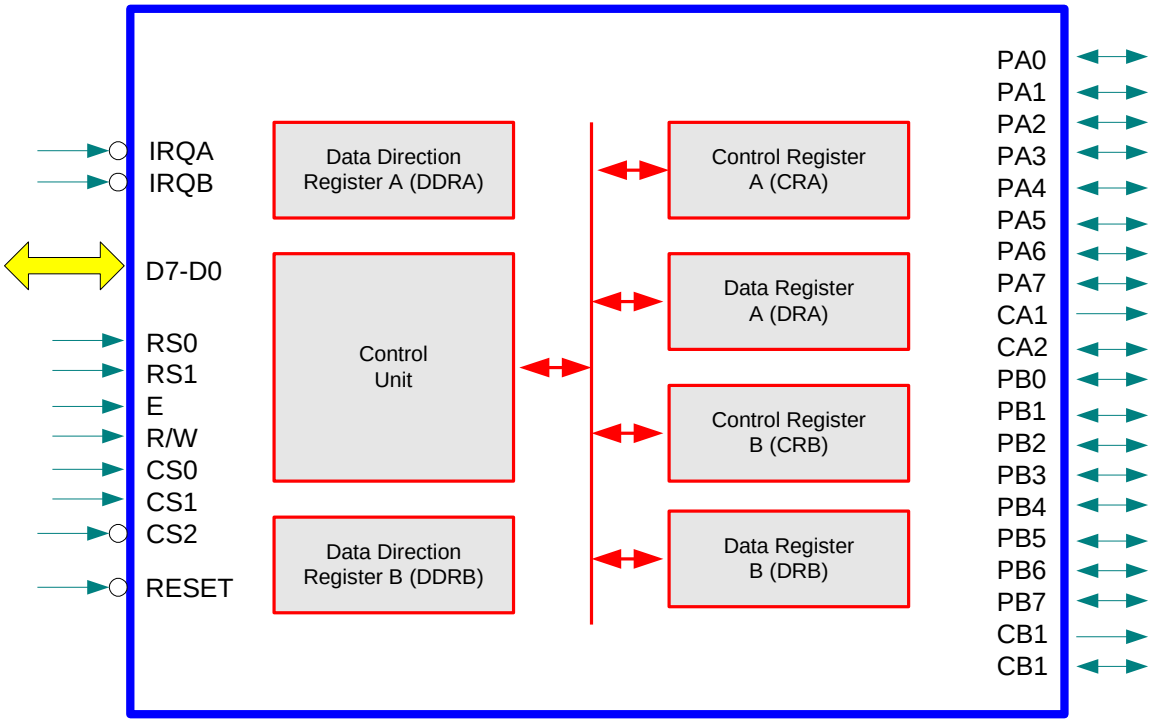
PIA 6821

- PIA has 6 internal registers that can be programmed to its need
- Each port has three registers

- **For Port A**
  - **Data Register A (DRA)**
    To transfer data in or out
  - **Data Direction Register A (DDRA)**
    Determine direction of each line
  - **Control Register A (CRA)**
    Control the operation of port A
- **For Port B**
  - **Data Register B (DRB)**
    To transfer data in or out
  - **Data Direction Register B (DDRB)**
    Determine direction of each line
  - **Control Register A (CRB)**
    Control the operation of port B

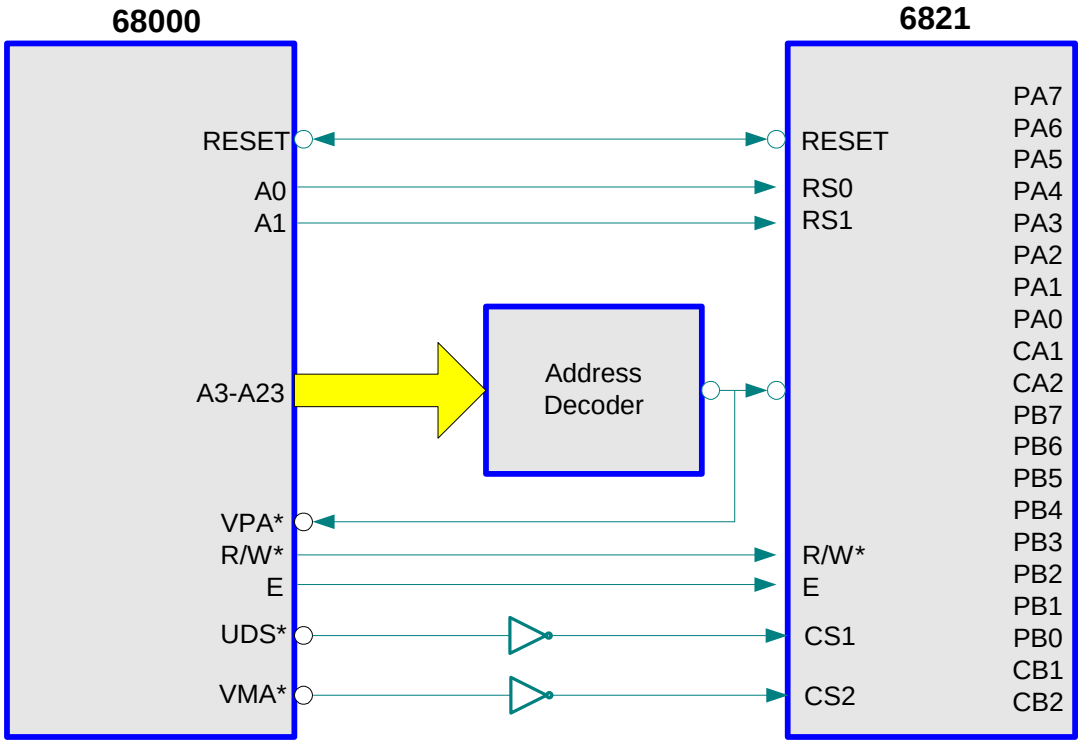
PIA 6821

- Internal structure of PIA consist of six registers and control unit



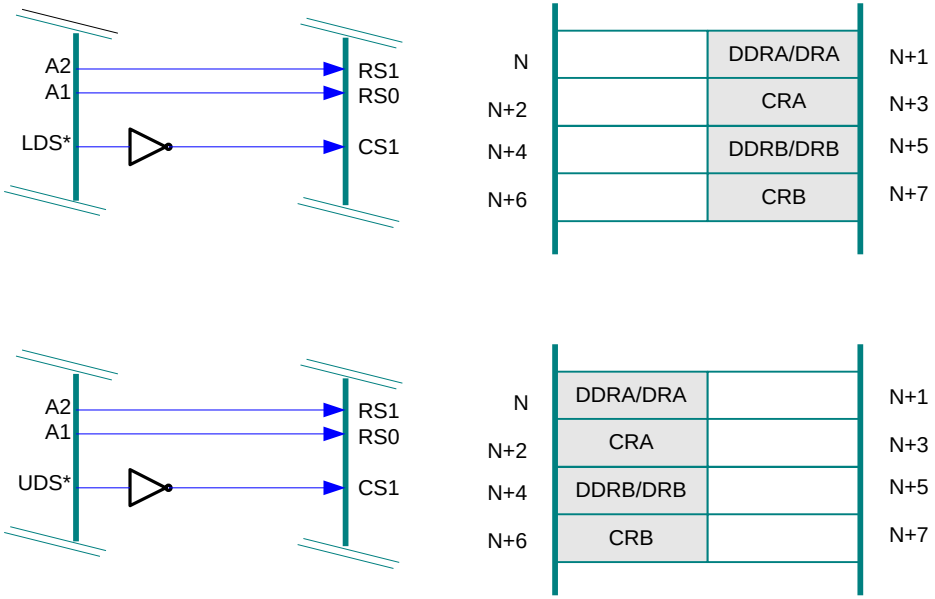
Interface With 68000

- Since 6821 is operating in synchronous mode, additional pin connections are required



PIA Registers

- The address of 6 internal registers of PIA depends on
  - Address decoder
  - Data Strobe connection (`UDS` or `LDS`)



PIA Registers

- Since there is 6 registers and only four locations, two registers are located in the same address
- To distinguish these registers, bit2 of control register is used

RS1	RS1	Register
0	0	Data Register A/Data Direction Reg. A
0	1	Control Register A
1	0	Data Register B/Data Direction Reg. B
1	1	Control Register B

RS1	RS0	CRA2	CRB2	Register
0	0	0	X	Data Direction Reg. A (DDRA)
0	0	1	X	Data Register A (DRA)
0	1	X	X	Control Register A (CRA)
1	0	X	0	Data Direction Reg. B (DDRB)
1	0	X	1	Data Register B (DRB)
1	1	X	X	Control Register B (CRB)

PIA Registers

Data Register A (DRA)

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
DRA7	DRA6	DRA5	DRA4	DRA3	DRA2	DRA1	DDRA0

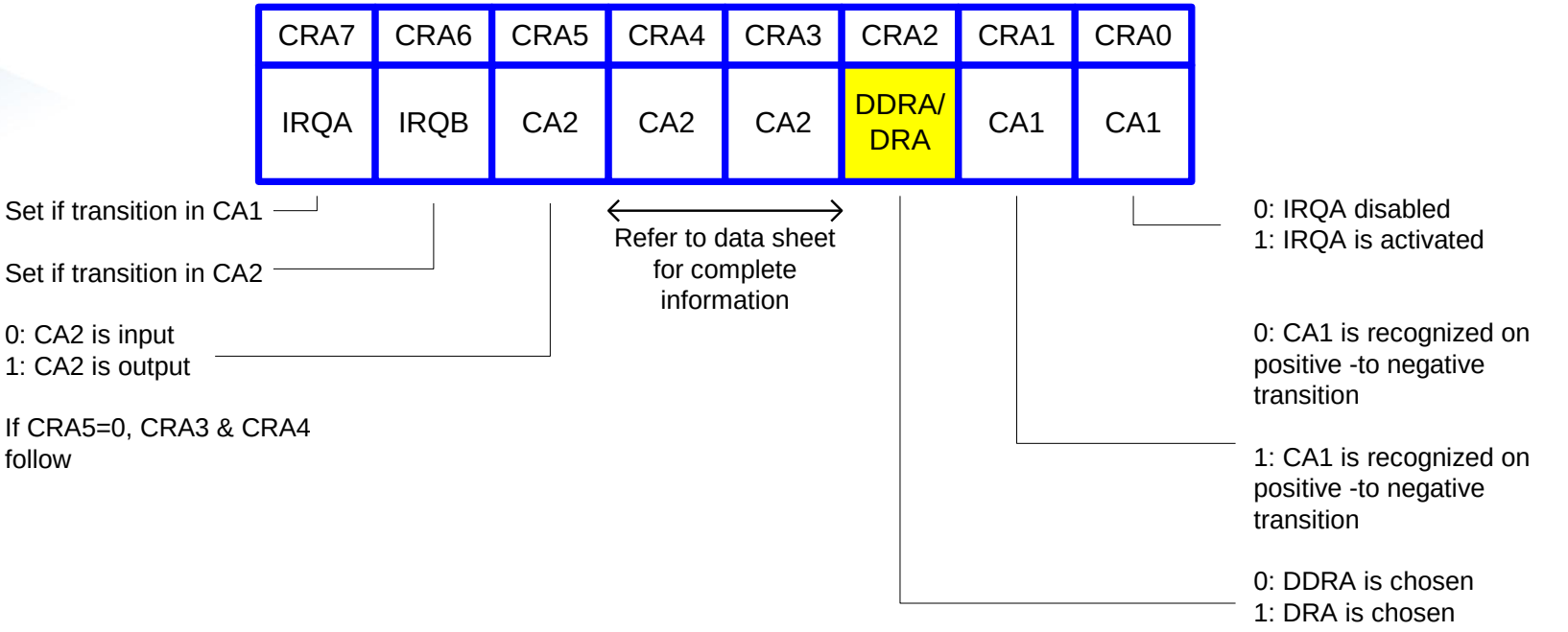
Data Direction Register A (DDRA)

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
DDRA7	DDRA6	DDRA5	DDRA4	DDRA3	DDRA2	DDRA1	DDRA0

Logic 1 in each DDRAi causes the signal line PAi to become **output line**
Logic 0 in each DDRAi causes the signal line PAi to become **input line**

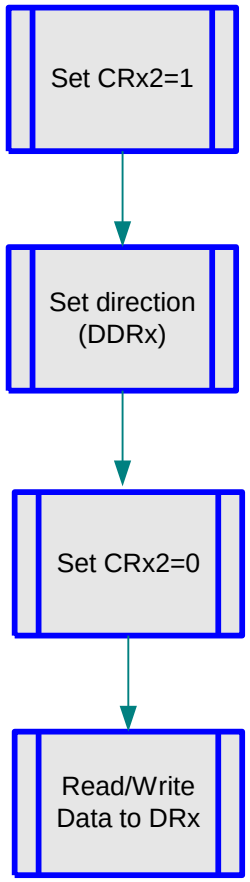
For example:
If DDRA = $4C, PA6, PA3 and PA2 are output and others are input

Control Register



Initialize PIA

- Initializing PIA is the most crucial steps in programming a system using this I/O device
- Followings are brief procedure to initialize PIA for simple I/O (Assuming port A)
  - Fill bit 2 of `CRA` with 0 to access `DDRA`
  - Fill `DDRA` with proper value to determine the role of each signal line of the port A (1 for output and 0 for input)
  - Fill bit 2 of `CRA` with 1 to access `DRA`
  - Write or read data using `DRA`



Initialize PIA

- **Example** : Initialize PIA assuming port A is an output port and port B is an input port

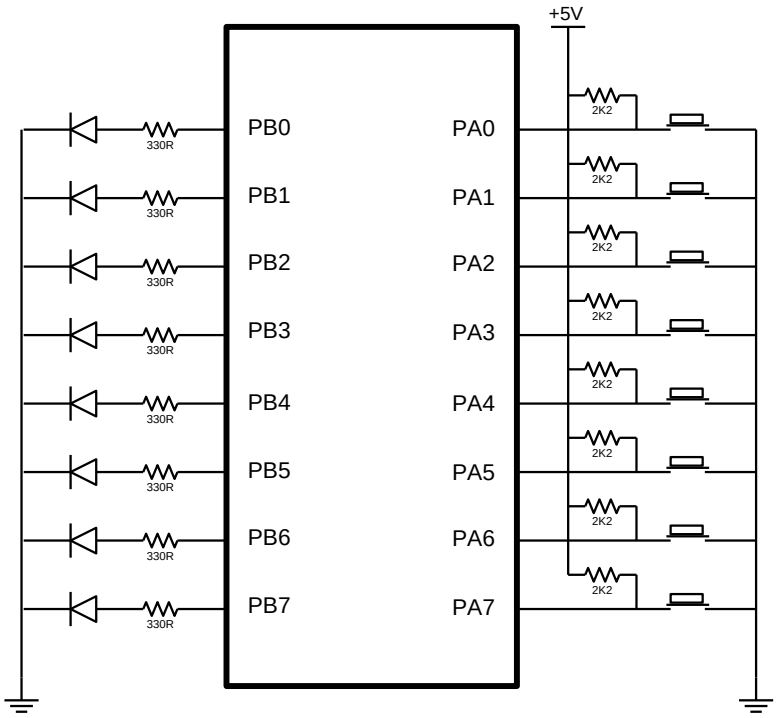
```
; Initialize Port A
MOVE.B #0,CRA
MOVE.B #$FF,DDRA
MOVE.B #4,CRA

; Initialize Port B
MOVE.B #0,CRB
MOVE.B #0,DDRB
MOVE.B #4,CRB
```

More Examples..

- Write a program to switch on the LED as the respective switch is pressed assuming all ports are initialized

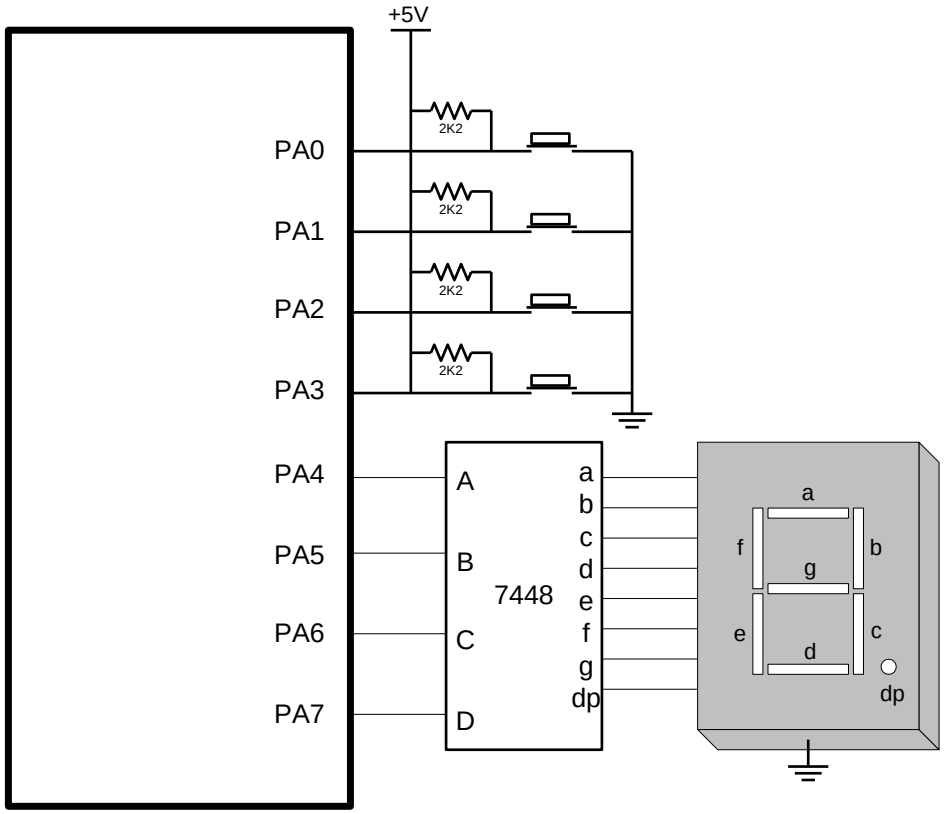
```
START   MOVE.B DRA,D0
        NOT.B D0
        MOVE.B DRB
        BRA START
```



More Examples..

- Write a program to display a pressed number on seven segment display

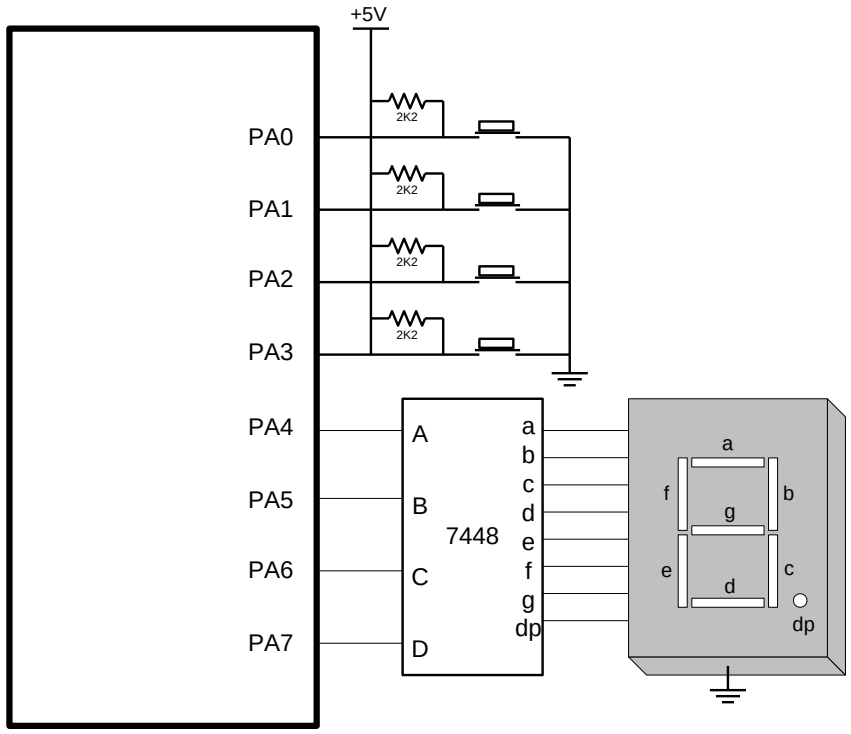
```
        CLR.B  CRA
        MOVE.B #$F0,DDRA
        MOVE.B #4,CRA
LOOP    MOVE.B DRA,D0
        ASL.B  #4,D0
        MOVE.B D0,DRA
        BRA    LOOP
```



More Examples..

- Write a program to display an upward counter (0,1,2..8,9,0…) continuously

```
MAIN CLR.B  D0
LOOP MOVE.B D0,DRA
     BSR    DELAY
     ADDQ.B #1,D0
     CMPI.B #10,D0
     BNE    LOOP
     BRA    MAIN
```



Self-Test

- **Exercise**

  Write as simple program to initialize port A where PA0-PA3 are output and PA4-PA7 are input

- **Exercise**

  If port B is connected to 7-segment display, write a program to initialize port B. Then, write a program to display downward counter on port B

- **Exercise**

  If PB0 is connected to a low current speaker, write a program to generate continuous tone with frequency of 1kHz to the speaker

Self-Test

Exercise

Write a program to meet following requirements,

- If switch 1 is pressed, the segment will display upward counter
- If switch 2 is pressed, the segment will display downward counter
- If switch 3 is pressed, the segment will display random value continuously
- If switch 4 is pressed, the program is terminated