LAB 4

PARALLEL INTERFACE

1. INTRODUCTION

This lab gives information about MC6821 peripheral interface adapter. Read MC6821 datasheet which is in "Sinif Dosyalari" section. Study connections between MC6821 and MC6802. Learn memory map I/0.

2. PROGRAMMING 6821 PERIPHERAL INTERFACE ADAPTER

MC6821 has two bidirectional 8-bit buses. These two 8-bit buses can be programmed as input or output. MC6802 programs MC6821 using address and data buses. Figure-1 shows MC6821 connections on ITU-Training Kit. Leds are connected to PORT-A. Figure-2 shows LEDs' location.
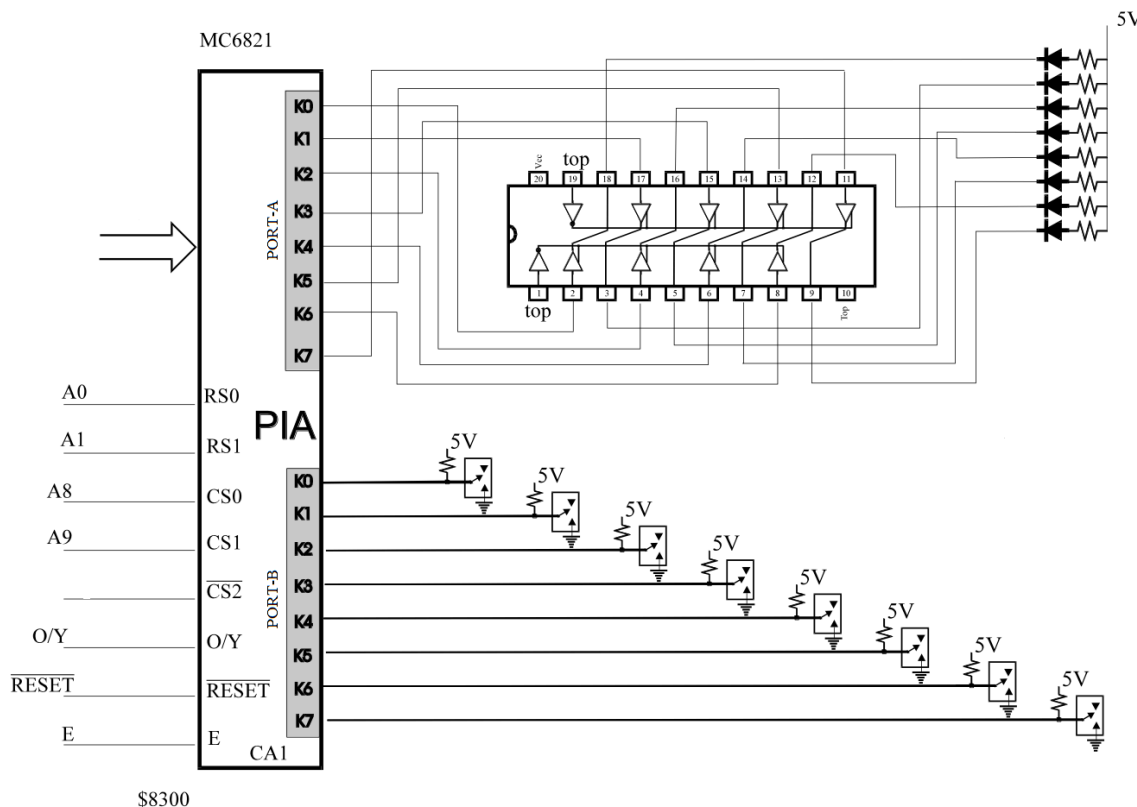

Figure-1

Address Table of MC6821

$8300	Port-A
$8300	Data Direction Register A
$8301	Control Register A
$8302	Port-B
$8302	Data Direction Register B
$8303	Control Register B

Note: If third bit of control register A is 0, $8300 points to data direction register. If it is 1, $8300 points to port-A value.

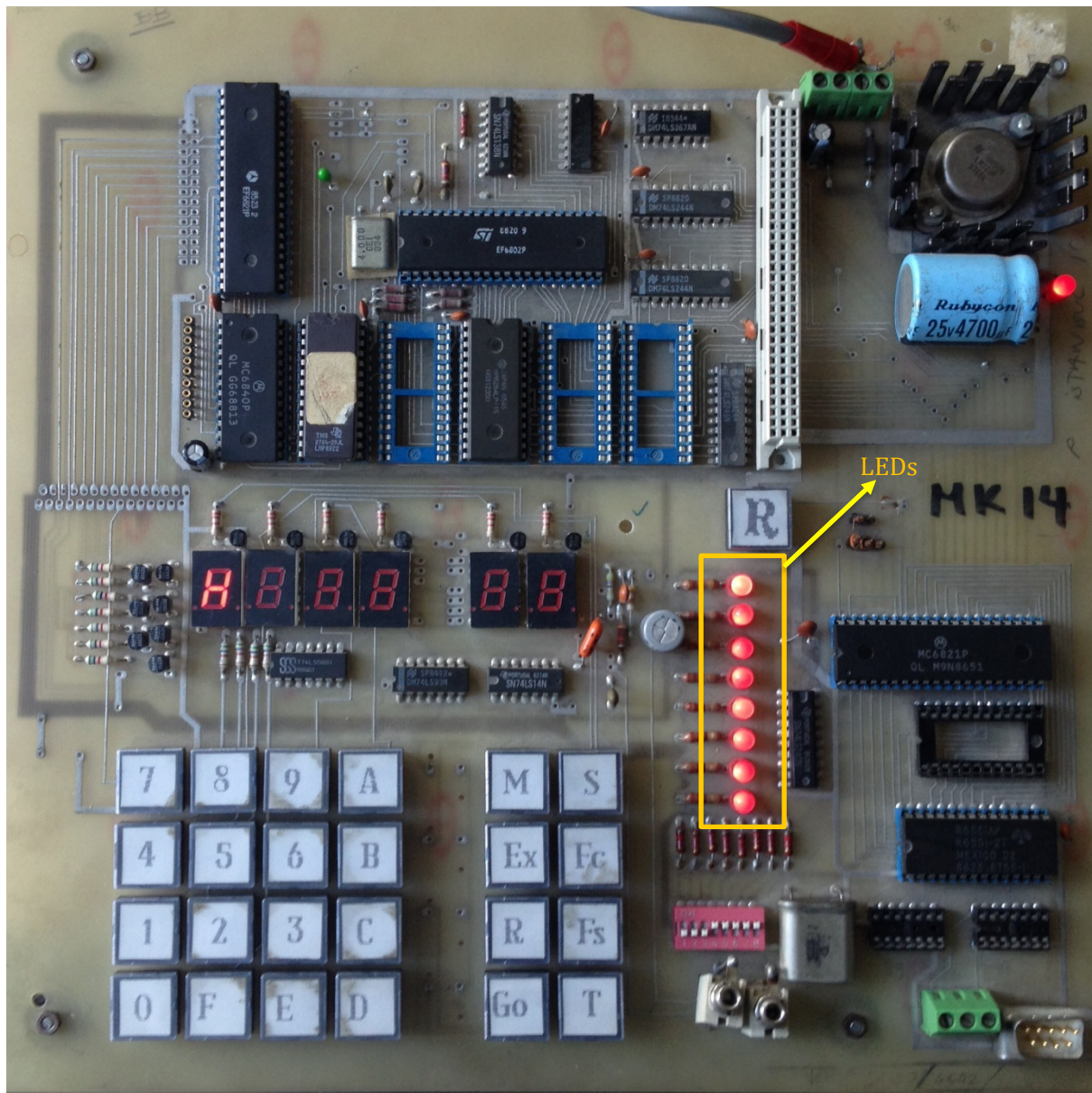You should write 0 or 1 to data direction register to set the port as input or output.

Figure-2

Run the following code on ITU-Training Kit. Report what you observe.

Simple LED Code

```
LDAA #$00
STAA $8301    //third bit of control register A is 0.
              //Now, $8300 points data direction register.
LDAA #$FF
STAA $8300    //Port-A is completely output.


LDAA #$04
STAA $8301    //third bit of control register A is 1


LDAA #$55
STAA $8300    //led0-2-4-6 are on, led1-3-5-7 is off.
SWI
```

3. EXPERIMENT

Write the assembly code for the following algorithm. Then, run it on ITU-TRAINING Kit. This code simple Knight Rider Led code. (see Knigth Rider Led on youtube.com) You cannot simulate LED code on simulator. But, simulator converts your assembly codes into machine codes. Understand the delay subroutine. Why do we count high amount?

```c
int leds[]={0x01, 0x02, 0x04, 0x08, 0x10, 0x20, 0x40, 0x80};
int i=0;
int main(){
      initPortA();
      while(1)
         for(i=0; i<8; i++){
            writePortA(leds[i]);
            delay();                 //approximately 1 second.
            }
}
```

Your code looks like following structure

```asm
        .org $4000

        LDAA #$00
        STAA $8301   //third bit of control register A is 0.
                     //Now, $8300 points data direction register.
        LDAA #$FF
        STAA $8300   //Port-A is completely output.
        LDAA #$04
        STAA $8301   //third bit of control register A is 1
        LDAA #$00
        STAA $8300   //initially all leds are off.

loop  …
        … //your algorithm
        …
        JSR delay
        BRA loop


delay   LDX #$0000
cnt     INX
        CPX #$FFFF
        BEQ term
        BRA cnt
term    RTS
```

In your report, answers and simulator results of section 2 and 3 should be stated.