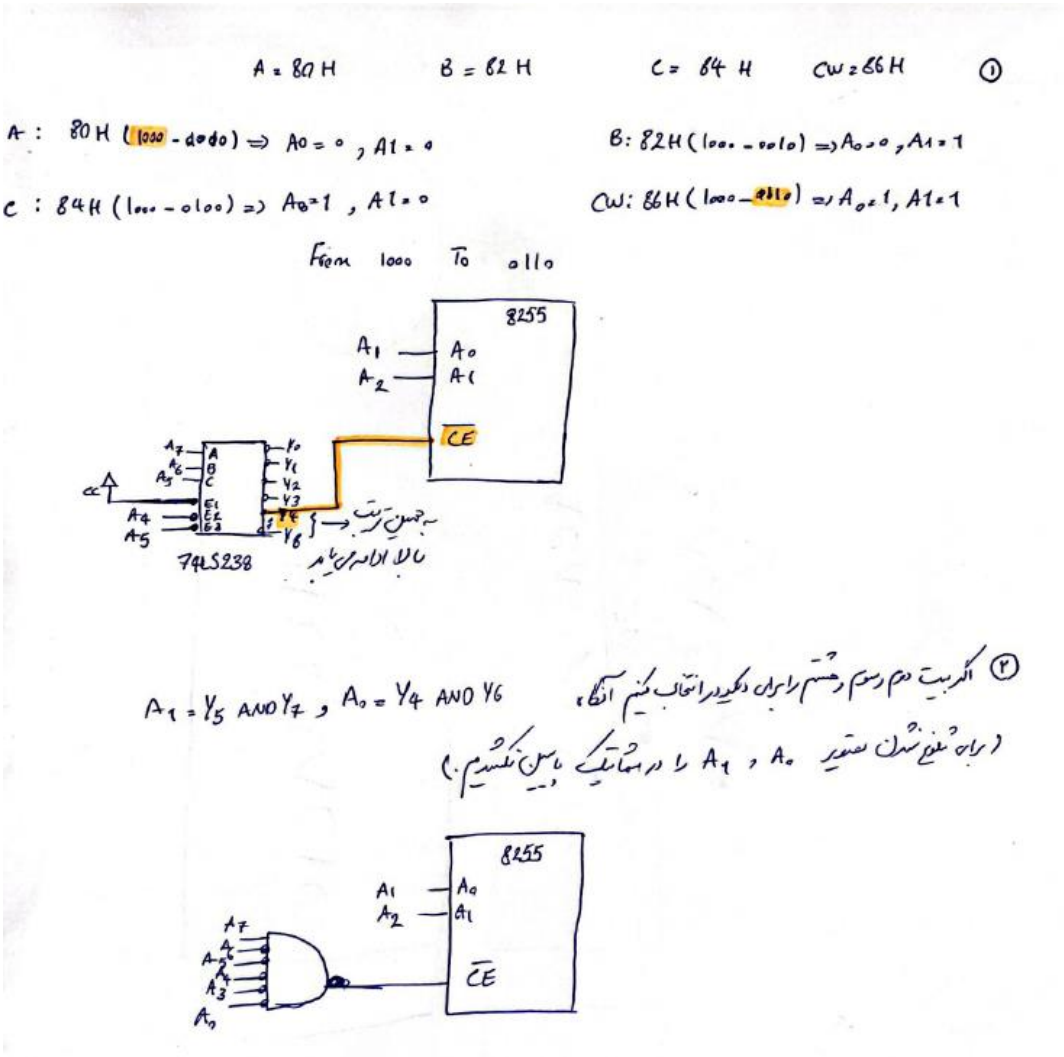
گزارش‌کار آزمایش دوم – آزمایشگاه ریزپردازنده

نگار هنرور صدیقیان-99243076

امین احسانی مهر-99243009

سوالات تحلیلی

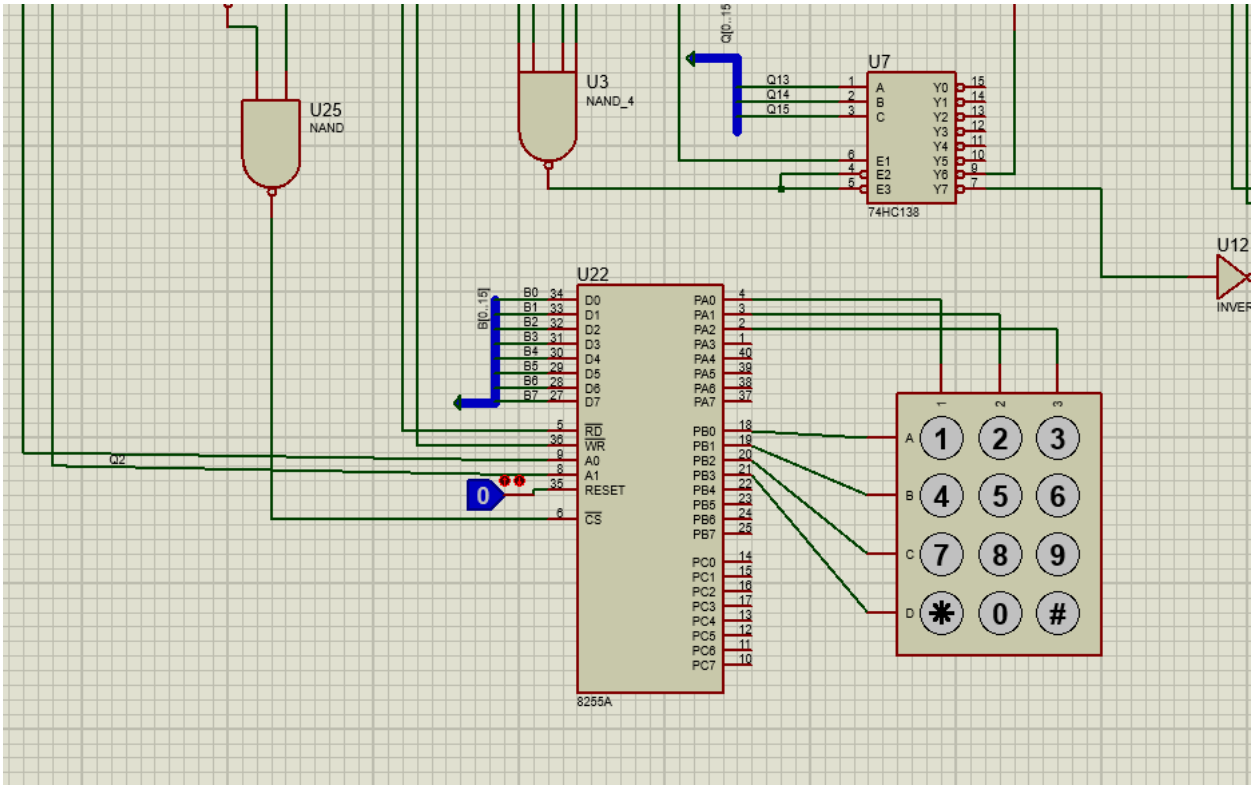


گزارش کد

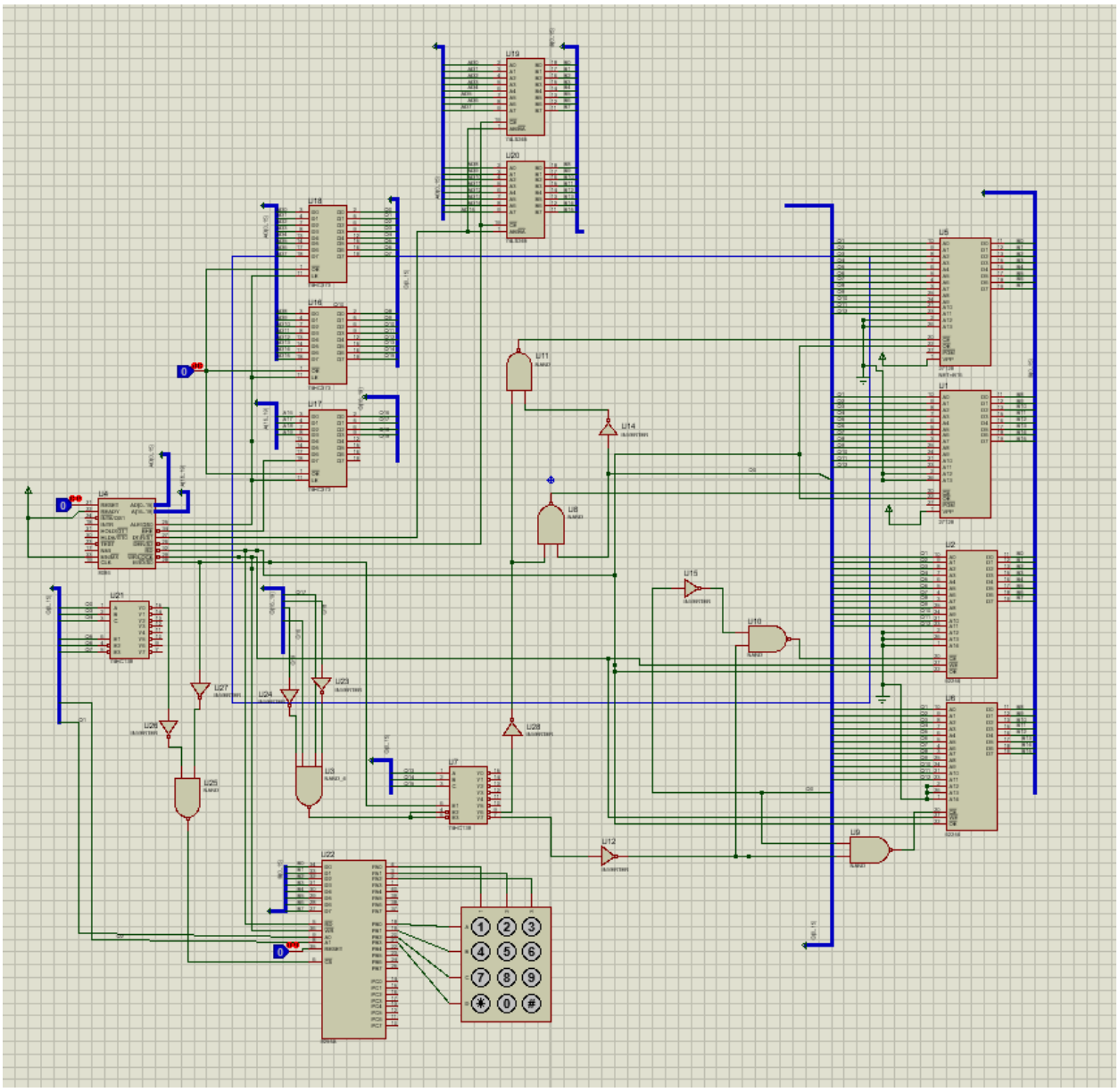
بخش B

سوالات 1 و 2

بخش b مشابه بخش a بوده و در اینجا برخی جزئیات به آن اضافه می‌شود. در اینجا یک بخش keypad اضافه می‌شود.



کاربرد این keypad به این صورت است که اگر عدد انتخاب شده روی آن عددی زوج بود مقداری را از cpy خوانده و اگر عدد مقداری فرد بود از invcpy مقدار بخواند و عملیات‌های مربوط به آن را یعنی کپی کردن N عدد اول حافظه rom به صورت مستقیم و یا معکوس به ابتدای حافظه رم است را انجام دهد.در اینجا یک پورت I/O در میکروپروسسور داریم که آن را گرفته و به کمک یک دیکودر معکوس مقادیر روی پورت را با یکدیگر nand میکنیم



اگر مقدار حاصل صفر شد بخش مربوطه در پردازنده 8255 را فعال کرده و ورودی ها را بخواند. این ورودی‌ها به این صورت هستند که از پورت a ورودی گرفته و از پورت b خروجی را .

توضیح کد

در اینجا CPY و INVCPY مشابه با آنچه در بخش A تعریف شد مجددا پیاده‌سازی شده‌اند اما تفاوتی که در مقایسه با بخش A در اینجا وجود دارد این است که آدرس‌های شروع تغییریافته‌اند.

```
CPY     PROC    FAR

    MOV AX, 3c00H
    MOV ES, AX
    MOV AX, 3e00H
    MOV DS, AX


    MOV BX, 20 ; this contains n
    INC BX
    MOV SI,BX
    cpyloop:
        DEC SI
        CMP SI, 0 ; n = n-1 , check if n >= 0

        MOV AL, ES:[SI]
        MOV [SI], AL

        JNZ cpyloop

   RET
CPY     ENDP
```

در اینجا یک پراسس به نام TWO_BANK داریم که برای بانک‌های زوج و فرد در نظر گرفته شده‌اند.

```
TWO_BANK PROC FAR

        ; SET DATA SEGMENT OF RAM
        MOV AX, 3E00H
        MOV DS, AX
        MOV DI, 42
        MOV DX, 9876H
        MOV [DI+1], DX

        RET
```

یک پراسس به نام PHONE_read داریم که در اینجا اطلاعات مربوط به تراشه 8225 را ست میکنیم و در ادامه در این پراسس چگونگی تقسیم داده‌ها و ذخیره‌سازی آن‌ها در بانک‌های زوج و فرد پیاده‌سازی می‌شود؛در مراحلی که با لیبل con1,con2,con3 مشخص شده‌اند بررسی میکند که از میان کلیدهای ممکن کدام کلید انتخاب شده‌است و بسته به عدد فشرده شده جامپ به بخش مربوطه کد انجام می‌شود،

```
75  PHONE_READ PROC FAR
76          ; CONFIG 8255
77          MOV AL, 10000010B
78          OUT 26H, AL
79
80          MOV BL, 11111111B
81          CON1:
82                  MOV AL, 6
83                  OUT 20H, AL
84                  IN AL, 22H
85                  XOR AL, BL
86                  CMP AL, 0
87                  JE CON2
88                  ; CHECK EVEN OR ODD
89                  CMP AL, 2
90                  JNZ ODD1
91                  JMP EVEN1
92          CON2:
93                  MOV AL, 5
94                  OUT 20H, AL
95                  IN AL, 22H
96                  XOR AL, BL
97                  CMP AL, 0
98                  JE CON3
99                  ; CHECK EVEN OR ODD
100                 CMP AL, 2
101                 JNZ EVEN1
102                 JMP ODD1
```

سپس در ادامه در بخش ODD1 و EVEN1 باقی‌مانده عدد نسبت به ۲ را بررسی میکند که ۱ شده یا صفر شده تا با توجه به آن مجددا به CPY که مربوط به زوج‌ها بوده و INVCPY که مربوط به اعداد فرد است جامپ کند.

```
103         CON3:
104                 MOV AL, 3
105                 OUT 20H, AL
106                 IN AL, 22H
107                 XOR AL, BL
108                 CMP AL, 0
109                 JE CON1
110                 ; CHECK EVEN OR ODD
111                 CMP AL, 2
112                 JNZ ODD1
113                 JMP EVEN1
114         ODD1:
115                 MOV DX, 1
116                 RET
117         EVEN1:
118                 MOV DX, 0
119                 RET
120
121 PHONE_READ ENDP
```

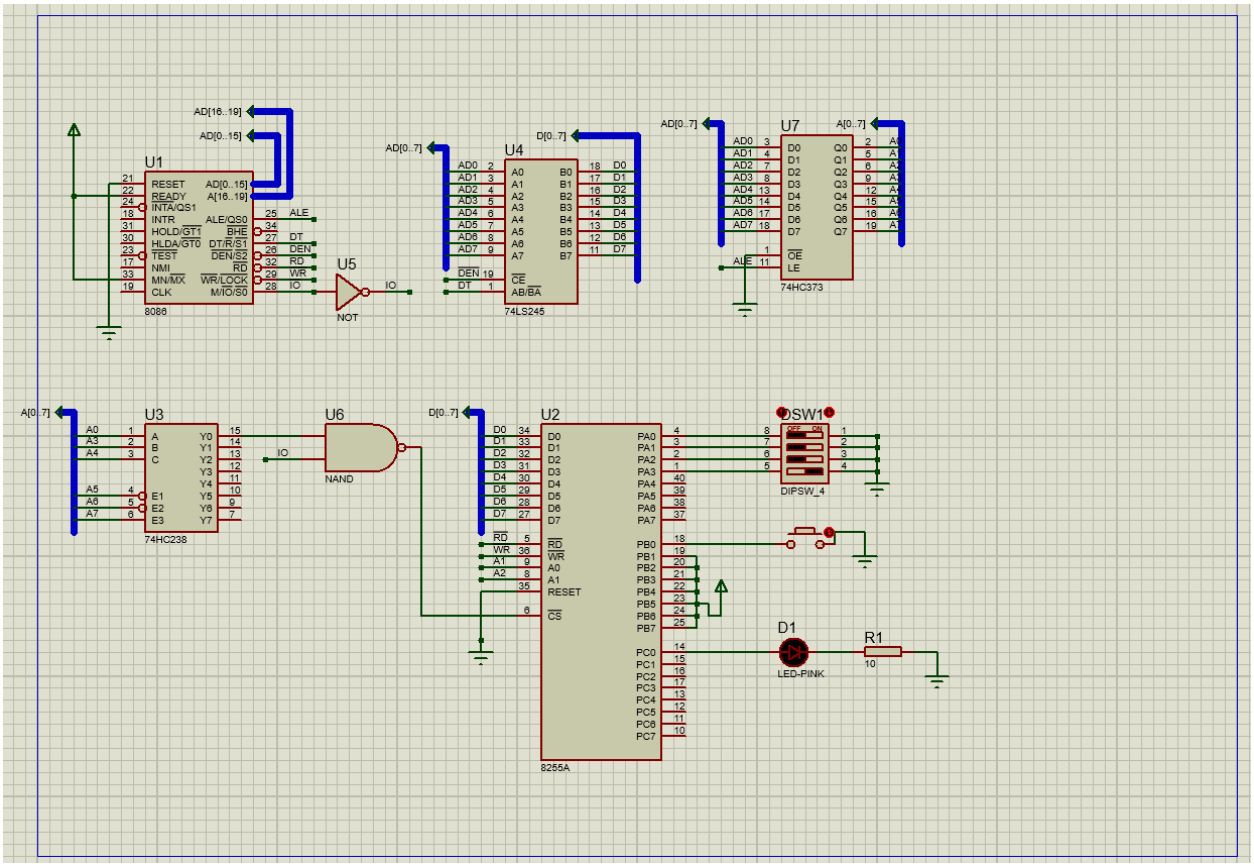
سوال 3

در اینجا به دنبال طراحی سیستمی با عملکرد فوق هستیم:

کاربر عدد N بین 0 تا 15 را از طریق یک dip switch تنظیم می‌کند و سپس دکمه start را می‌زند.ما به دنبال این هستیم که پس از فشردن این دکمه یک LED به تعداد دفعات N و با سرعت یکبار در ثانیه چشمک بزند.

برای پیاده‌سازی عملیات گفته شده در بالا یک لچ داریم؛داده‌های این لچ به پورت‌های a0 و a1 از تراشه 8255 متصل هستند و بقیه داده‌های آن به دیکودر وصل میشوند.سپس یک data transceiver داریم که وظیفه آن بررسی این است که اگر bus حاوی دیتا بود آن دیتا را به تراشه 8255 منتقل کرده تا عملیات‌های مورد نیاز توسط این تراشه روی آن انجام شود.عدد مورد نظر N را از دیکودر انتخاب می‌کنیم سپس مقدار آن را با مقدار پورت I/O ، عملیات nand میکنیم و خروجی آن به chip select تراشه 8255 منتقل می‌شود.

در ادامه یک dip switch داریم که مشابه آنچه گفته شد عدد ورودی N روی این dip switch انتخاب و تنظیم میشود. یک button به نام start داریم که این دکمه ورودی (مقدار N) را از روی dip switch می‌خواند. در نهایت یک LED داریم که خروجی به واسطه دفعات روشن و خاموش شدن این LED به نمایش گذاشته می‌شود.



توضیح کد

در ابتدا لازم است که یک تابع DELAY بنویسیم ، کاربرد این تابع برای این است که تاخیر لازم میان هربار چشمک زدن چراغ LED ایجاد شود و چراغ با سرعت یکبار در ثانیه چشمک بزند.

```
11  .MODEL SMALL
12  .STACK 64
13  .DATA
14  .CODE
15  DELAY PROC
16
17      PUSH CX
18
19      MOV CX, 9999H
20      DELAY_:
21          CMP CX, 0
22          LOOP DELAY_
23
24      POP CX
25
26      RET
27
28  DELAY ENDP
```

در ادامه یک تابع MAIN داریم که در این تابع دیتاسگمنت را در ابتدا روی AX ست میکنیم و مقدار رجیستر CX که شمارنده تعداد دفعات چشمک زدن چراغ است را در ابتدا برابر صفر قرار میدهیم تا جلوتر به تعداد موردنظر مقداردهی شود. سپس مجددا یک کانفیگ از تراشه 8225 داشته و در ادامه وارد لیبل READ_IN میشویم. در این لیبل ابتدا چک میکنیم که ببینم این دکمه‌ای که فشار داده‌ایم روشن است یا خاموش؛ اگر خاموش بود مجددا جامپ میکنیم به بخش READ_IN و تا زمانی که کلید را نزده ایم این کار را تکرار میکند و اگر روشن بود و مقدار AL صفر نبود مقداری که روی dip switch قرار دارد را میخوانیم و به CL منتقل میکنیم و به تعداد آن لامپ را خاموش روشن میکنیم،در ادامه در بخش TURN_ON مقدار AL که مربوط به روشن شدن LED است را برابر یک کرده و مقدار AL را به 8225 منتقل میکنیم و در ادامه مقدار رجیستر AL را به DL منتقل کرده و برای تامین سرعت مورد نظر تابعDELAY را صدا میکنیم. سپس مقدار رجیستر AL را صفر کرده، تابع DELAY را فراخوانده و مقدار CX که تعداد دفعات روشن شدن چراغ LED ما و درنتیجه دفعات اجرای حلقه را نشان می‌دهد را بررسی میکنیم که صفر شده است یا نه و تا زمانی که صفر نشده لیبل TURN_ON را به صورت حلقه تکرار میکنیم.

در انتها مجددا به لیبل READ_IN برمیگردیم تا از روی dip switch مقدار جدید خوانده شده و فرآیند فوق مجددا تکرار شود.

```
30  MAIN PROC FAR
31
32      MOV AX, @DATA
33      MOV DS, AX
34
35      MOV CX, 0
36
37      ; CONFIG THE 8255
38      MOV AL, 92H
39      OUT 86H, AL
40
41  READ_IN:
42          IN AL, 82H
43          NOT AL
44          CMP AL, 0
45          JE READ_IN
46          IN AL, 80H
47          NOT AL
48          MOV CL, AL
49  TURN_ON:
50          MOV AL, 1H
51          OUT 84H, AL
52          MOV DL, AL
53          CALL DELAY
54          MOV AL, 0H
55          OUT 84H, AL
56          CALL DELAY
57          CMP CX, 0
58          LOOP TURN_ON
59
60      JMP READ_IN
61
62  MAIN ENDP
63      END MAIN
```