**2η Σειρά Ασκήσεων**

Ειρήνη Στρουμπάκου 03121183

Χαράλαμπος Παπαδόπουλος 03120199

1. Το πρόγραμμα σε γλώσσα assembly είναι το εξής

|  |
| --- |
| ;a  IN 10H  MVI A,00H ;Store the value 00H in A  LXI H,0900H ;Store the value 0900H in HL  START:  MOV M,A ;Store the (A) to (H)(L)  INX H  INR A  CPI 80H ;Check if it has reached 128, so it's greater than 127  JC START  ;b  LXI B,0000H ;Initialise the counter of ones  MVI D,00H ;Initialise the counter of 0 to 127 to 0 value  START2:  INR D ;Increase the counter of 0 to 127  MOV A,D ;Store each number we use in A  CPI 80H ;Check if it has reached 128, so it's greater than 127  JNC ENDING  MVI E,08H ;Initialize a counter for the shifts  JMP CHECK    CHECK:  DCR E ;Check if 7 shifts have been done  JZ START2 ;If yes go to the next number  RRC  JNC CHECK ;If the LSB was not 1 then continue the shifts  INX B ;Else increase the ones counter  JMP CHECK ;And continue with the shifts  ;c  MVI D,00H ;Initialize the counter  MVI A,00H ;Set the first number in 0  CHECK2:  INR A  CPI 80H ;Check if it has reached 128, so it's greater than 127  JNC ENDING  CPI 10H ;Check if greater than 10H  JC STOP ;If not, do not increase and go to STOP which jumps to CHECK  CPI 61H ;Check if smaller than 61H, to include 60H  JNC STOP ;If not, do not increase and go to STOP which jumps to CHECK  INR D ;If it passes all checks, increase counter D  JMP CHECK2  STOP:  JMP CHECK2    ENDING:  END |

2)

|  |
| --- |
| START:  IN 10H  LXI D,FFFFH ;set delay  MVI L,16H ;multiply delay by L  ;for 20 seconds L=16H and program virtual delay cursor at 2nd position  INPUT:  LDA 2000H ;first input  PUSH\_BUTTON:  ANI 01H ;check LSB  JNZ HIGH  LOW: ;LSB 0 subroutine  MOV A,C ;check counter  CPI 02H ;if counter=2 that means the previous state was HIGH  JZ LIGHTS\_ON  JNZ LOOP\_LOW  LOOP\_LOW: ;stay here while LSB doesn't change from 0  LDA 2000H  CPI 01H  JNZ LOOP\_LOW  INR C  JMP HIGH  HIGH:  MOV A,C  CPI 00H ;in case the first state was LSB=1. avoid tunging on the\_  JZ INPUT ;\_lights after just HIGH-LOW sequence  CPI 01H ;if counter=1 that means the previous state was LOW  JNZ LOOP\_HIGH  LOOP\_HIGH: ;stay here while LSB doesn't change from 1  LDA 2000H  CPI 01H  JZ LOOP\_HIGH  INR C  JMP LOW  LIGHTS\_ON:  MVI A,FFH  CMA  STA 3000H  MVI C,00H ;reset the counter for future inputs  JMP DELAY  DELAY:  DCR L  JNZ SUBDELAY  JMP END1  SUBDELAY:  CALL CHECK\_INPUT ;keep checking for PUSH sequence  DCX D  MOV A,D  ORA E  JNZ SUBDELAY  JMP DELAY  CHECK\_INPUT:  LDA 2000H  CPI 00H  JZ LOW2  JNZ HIGH2  ;same logic as before but now we need to return to the delay since we need to keep  ;counting down in between possible inputs.  HIGH2:  MOV A,C  CPI 01H ;if counter=1 that means the previous state was LOW  JZ HELP\_HIGH2  RET  HELP\_HIGH2:  INR C  RET  LOW2:  MOV A,C  CPI 00H ;if counter=0 that means the previous state was LOW  JZ HELP\_LOW2  CPI 02H ;if counter=2 that means the previous state was HIGH  JZ REFRESH\_DELAY  RET  HELP\_LOW2:  INR C  RET  REFRESH\_DELAY:  LXI D,FFFFH  MVI L,16H  MVI C,00H  JMP DELAY  END1:  MVI A,00H  CMA  STA 3000H ;turn off the lights  END |

3)

i)

|  |
| --- |
| LOOP1:  MVI B,00H ;Initialize the counter of shifts  MVI C,09H ;Initialize another counter to find if the input is 0  LDA 2000H  START:  INR B ;For each shift, we increase our counter  DCR C ;We decrease the initialized to 9 counter and check if it has become 0  JZ ENDING1 ;If yes, A=0 and we go to ending1  RRC ;We make a right shift till we find 1  JNC START ;If we don’t find 1, loop  FOUND1:  MVI A,01H  DCR B ;We want to store 2^(B-1)  MAKE:  RLC ;We move A left till we reach the desirable bit position  DCR B ;We decrease the shift counter for each left shift  JNZ MAKE ;If the shifts counter doesn’t reach 0, loop  JMP ENDING2 ;If we find 1, got to ending2  ENDING1:  MVI A,00H  CMA  STA 3000H  JMP LOOP1 ;The program doesn’t stop  ENDING2:  CMA  STA 3000H  JMP LOOP1  END |

ii)

|  |
| --- |
| LOOP1:  CALL KIND  CPI 00H ;If 0, go to ending1  JZ ENDING1  CPI 09H  JZ ENDING1 ;If 9, go to ending1  MOV B,A ;Load in B the given number A  MVI A,01H  DCR B  JZ NUM1 ;If B-1 is 0, then the given number is 1  MAKE:  DCR B ;Decrease B till it reaches 0, so Ι have made all the  JZ ENDING2 ;necessary shifts  RLC ;Move left and add 1, so I create  ADI 01H ;my number in a form 00111  JMP MAKE  NUM1:  MVI A,00H ;If given number is 1, turn on all LEDs  JMP ENDING2  ENDING1:  MVI A,00H ;For 0 or 9 turn off all LEDs  CMA  STA 3000H  JMP LOOP1  ENDING2:  STA 3000H  JMP LOOP1  END |

iii)

|  |
| --- |
| IN 10H  LOOP1:  MVI A,10H  LXI D,0B00H ;I create 4 empty spaces, because I only  STA 0B00H ;want to show 2 numbers  STA 0B01H  STA 0B02H  STA 0B03H  MVI A,FEH ;The 0 line has code 11111110  STA 2800H  LDA 1800H  ANI 07H ;I want to keep the last 3 digits  CPI 06H ;If 110 INSTR STEP  JZ INSTR\_STEP  CPI 05H ;If 101 FETCH PC  JZ FETCH\_PC  MVI A,FDH ;The 1 line has code 11111101  STA 2800H  LDA 1800H  ANI 07H ;I want to keep the last 3 digits  CPI 06H ;If 110 RUN  JZ RUNS  CPI 05H ;If 101 FETCH REG  JZ FETCH\_REG  CPI 03H ;If 011 FETCH ADDRS  JZ FETCH\_ADDRS  MVI A,FBH ;The 2 line has code 11111011  STA 2800H  LDA 1800H  ANI 07H ;I want to keep the last 3 digits  CPI 06H ;If 110 ZERO  JZ ZERO  CPI 05H ;If 101 STORE/INCR  JZ STORE\_INCR  CPI 03H ;if 011 DECR  JZ DECR  MVI A,F7H ;The 3 line has code 11110111  STA 2800H  LDA 1800H  ANI 07H ;I want to keep the last 3 digits  CPI 06H ;If 110 ONE  JZ ONE  CPI 05H ;If 101 TWO  JZ TWO  CPI 03H ;If 011 THREE  JZ THREE  MVI A,EFH ; The 4 line has code 11101111  STA 2800H  LDA 1800H  ANI 07H ;I want to keep the last 3 digits  CPI 06H ;If 110 FOUR  JZ FOUR  CPI 05H ;If 101 FIVE  JZ FIVE  CPI 03H ;If 011 SIX  JZ SIX  MVI A,DFH ;The 5 line has code 11011111  STA 2800H  LDA 1800H  ANI 07H ;I want to keep the last 3 digits  CPI 06H ;If 110 SEVEN  JZ SEVEN  CPI 05H ;If 101 EIGHT  JZ EIGHT  CPI 03H ;If 011 NINE  JZ NINE  MVI A,BFH ;The 6 line has code 10111111  STA 2800H  LDA 1800H  ANI 07H ;I want to keep the last 3 digits  CPI 06H ;If 110 A  JZ BUT\_A  CPI 05H ;If 101 B  JZ BUT\_B  CPI 03H ;If 011 C  JZ BUT\_C  MVI A,7FH ;The 7 line has code 01111111  STA 2800H  LDA 1800H  ANI 07H ;I want to keep the last 3 digits  CPI 06H ;If 110 D  JZ BUT\_D  CPI 05H ;If 101 E  JZ BUT\_E  CPI 03H ;If 011 F  JZ BUT\_F  JMP LOOP1  INSTR\_STEP:  MVI A,06H ;We first store the last digit of the 2 digits number  STA 0B04H  MVI A,08H ;We then store in the next position the second  STA 0B05H ;digit of the 2 digits number  JMP ENDING  FETCH\_PC:  MVI A,05H ;We first store the last digit of the 2 digits number  STA 0B04H  MVI A,08H ;We then store in the next position the second  STA 0B05H ;digit of the 2 digits number  JMP ENDING  RUNS:  MVI A,04H ;We first store the last digit of the 2 digits number  STA 0B04H  MVI A,08H ;We then store in the next position the second  STA 0B05H ;digit of the 2 digits number  JMP ENDING  FETCH\_REG:  MVI A,00H ;We first store the last digit of the 2 digits number  STA 0B04H  MVI A,08H ;We then store in the next position the second  STA 0B05H ;digit of the 2 digits number  JMP ENDING  FETCH\_ADDRS:  MVI A,02H ;We first store the last digit of the 2 digits number  STA 0B04H  MVI A,08H ;We then store in the next position the second  STA 0B05H ;digit of the 2 digits number  JMP ENDING  ZERO:  MVI A,00H ;We first store the last digit of the 2 digits number  STA 0B04H  STA 0B05H ;We then store in the next position the second  ;digit of the 2 digits number  JMP ENDING  STORE\_INCR:  MVI A,03H ;We first store the last digit of the 2 digits number  STA 0B04H  MVI A,08H ;We then store in the next position the second  STA 0B05H ;digit of the 2 digits number  JMP ENDING  DECR:  MVI A,01H ;We first store the last digit of the 2 digits number  STA 0B04H  MVI A,08H ;We then store in the next position the second  STA 0B05H ;digit of the 2 digits number  JMP ENDING  ONE:  MVI A,01H ;We first store the last digit of the 2 digits number  STA 0B04H  MVI A,00H ;We then store in the next position the second  STA 0B05H  JMP ENDING  TWO:  MVI A,02H ;We first store the last digit of the 2 digits number  STA 0B04H  MVI A,00H ;We then store in the next position the second  STA 0B05H ;digit of the 2 digits number  JMP ENDING  THREE:  MVI A,03H ;We first store the last digit of the 2 digits number  STA 0B04H  MVI A,00H ;We then store in the next position the second  STA 0B05H ;digit of the 2 digits number  JMP ENDING  FOUR:  MVI A,04H ;We first store the last digit of the 2 digits number  STA 0B04H  MVI A,00H ;We then store in the next position the second  STA 0B05H ;digit of the 2 digits number  JMP ENDING  FIVE:  MVI A,05H ;We first store the last digit of the 2 digits number  STA 0B04H  MVI A,00H ;We then store in the next position the second  STA 0B05H ;digit of the 2 digits number  JMP ENDING  SIX:  MVI A,06H ;We first store the last digit of the 2 digits number  STA 0B04H  MVI A,00H ;We then store in the next position the second  STA 0B05H ;digit of the 2 digits number  JMP ENDING  SEVEN:  MVI A,07H ;We first store the last digit of the 2 digits number  STA 0B04H  MVI A,00H ;We then store in the next position the second  STA 0B05H ;digit of the 2 digits number  JMP ENDING  EIGHT:  MVI A,08H ;We first store the last digit of the 2 digits number  STA 0B04H  MVI A,00H ;We then store in the next position the second  STA 0B05H ;digit of the 2 digits number  JMP ENDING  NINE:  MVI A,09H ;We first store the last digit of the 2 digits number  STA 0B04H  MVI A,00H ;We then store in the next position the second  STA 0B05H ;digit of the 2 digits number  JMP ENDING  BUT\_A:  MVI A,0AH ;We first store the last digit of the 2 digits number  STA 0B04H  MVI A,00H ;We then store in the next position the second  STA 0B05H ;digit of the 2 digits number  JMP ENDING  BUT\_B:  MVI A,0BH ;We first store the last digit of the 2 digits number  STA 0B04H  MVI A,00H ;We then store in the next position the second  STA 0B05H ;digit of the 2 digits number  JMP ENDING  BUT\_C:  MVI A,0CH ;We first store the last digit of the 2 digits number  STA 0B04H  MVI A,00H ;We then store in the next position the second  STA 0B05H ;digit of the 2 digits number  JMP ENDING  BUT\_D:  MVI A,0DH ;We first store the last digit of the 2 digits number  STA 0B04H  MVI A,00H ;We then store in the next position the second  STA 0B05H ;digit of the 2 digits number  JMP ENDING  BUT\_E:  MVI A,0EH ;We first store the last digit of the 2 digits number  STA 0B04H  MVI A,00H ;We then store in the next position the second  STA 0B05H ;digit of the 2 digits number  JMP ENDING  BUT\_F:  MVI A,0FH ;We first store the last digit of the 2 digits number  STA 0B04H  MVI A,00H ;We then store in the next position the second  STA 0B05H ;digit of the 2 digits number  JMP ENDING  ENDING:  CALL STDM ;We portray the number we want in  CALL DCD ;the screen above the keyboard  JMP LOOP1  END |

4)

|  |
| --- |
| START:  IN 10H  LDA 2000H  MVI C,00H  MOV B,A  LOOP1:  MOV A,C  CPI 00H  JZ AND0  CPI 01H  JZ AND1  CPI 02H  JZ XOR0  CPI 03H  JZ XOR1    AND0:  MOV A,B  ANI 01H ;B0  MOV D,A  MOV A,B  ANI 02H ;A0  RRC ;we bring A0 and B0 on the same position to compare  ANA D ;AND between A0 and B0  MOV D,A ;store the result in register D  INR C  JMP LOOP1  AND1:  MOV A,B  ANI 04H ;B1  MOV E,A  MOV A,B  ANI 08H ;A1  RRC ;we bring A1 and B1 on the same position to compare  ANA E  RRC ;X1 output  MOV E,A ;store the result in register E  MOV A,D ;second level OR gate comparison  RLC ;we bring D and E on the same position  ORA E ;OR between D and E  RRC ;move the result to output X0  MOV D,A ;store the result in register D  INR C  JMP LOOP1  XOR0:  MOV A,B  ANI 10H ;B2  MOV H,A  MOV A,B  ANI 20H ;A2  RRC ;we bring A2 and B2 on the same position to compare  XRA H  RRC  RRC ;X2 output  MOV H,A ;store the result in register H  INR C  JMP LOOP1  XOR1:  MOV A,B  ANI 40H ;B3  MOV L,A  MOV A,B  ANI 80H ;A3  RRC ;we bring A3 and B3 on the same position to compare  XRA L  RRC  RRC  RRC ;X3 output  MOV L,A ;store the result in register L  MOV A,H ;second level XOR comparison  RLC ;we bring H and L on the same position  ORA L ;OR between H and L  RRC ;move the result to output X2  MOV H,A ;store the result in register H  END:  MOV A,D  ORA E  ORA H  ORA L  CMA  STA 3000H  JMP START  END |