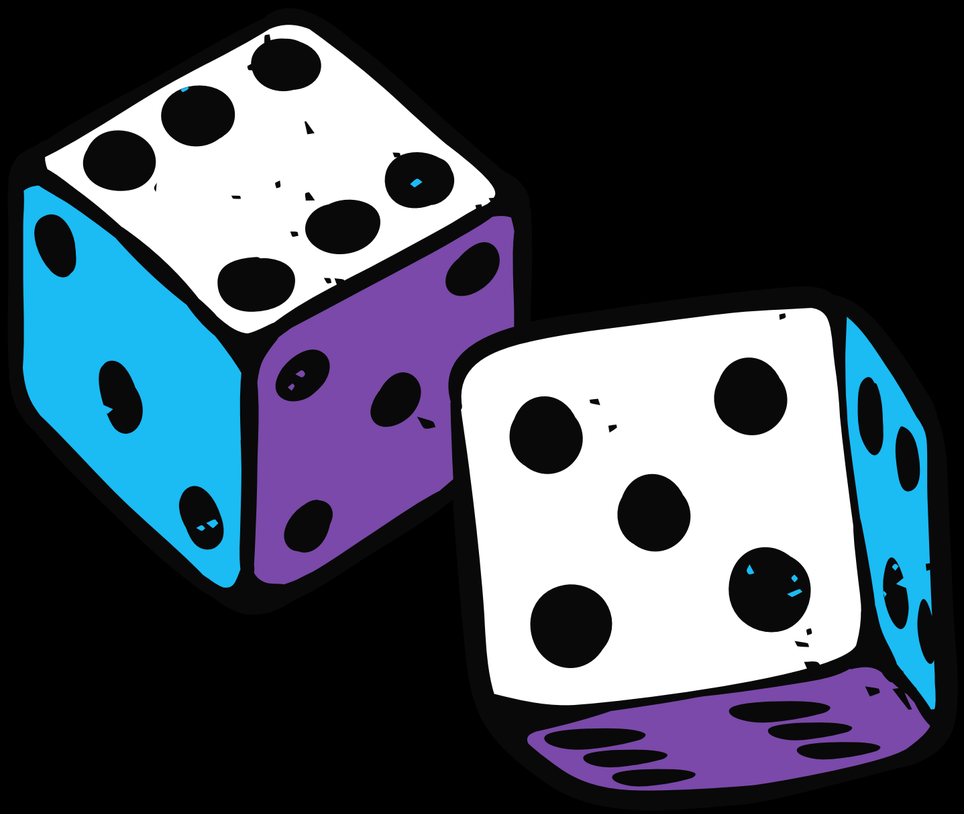
MPI LAB 8086 DIGITAL DICE



MPI LAB

a simulation of dice by using 8086

Our Group Members

ABHIRAM
B221025

APARNA
B221059

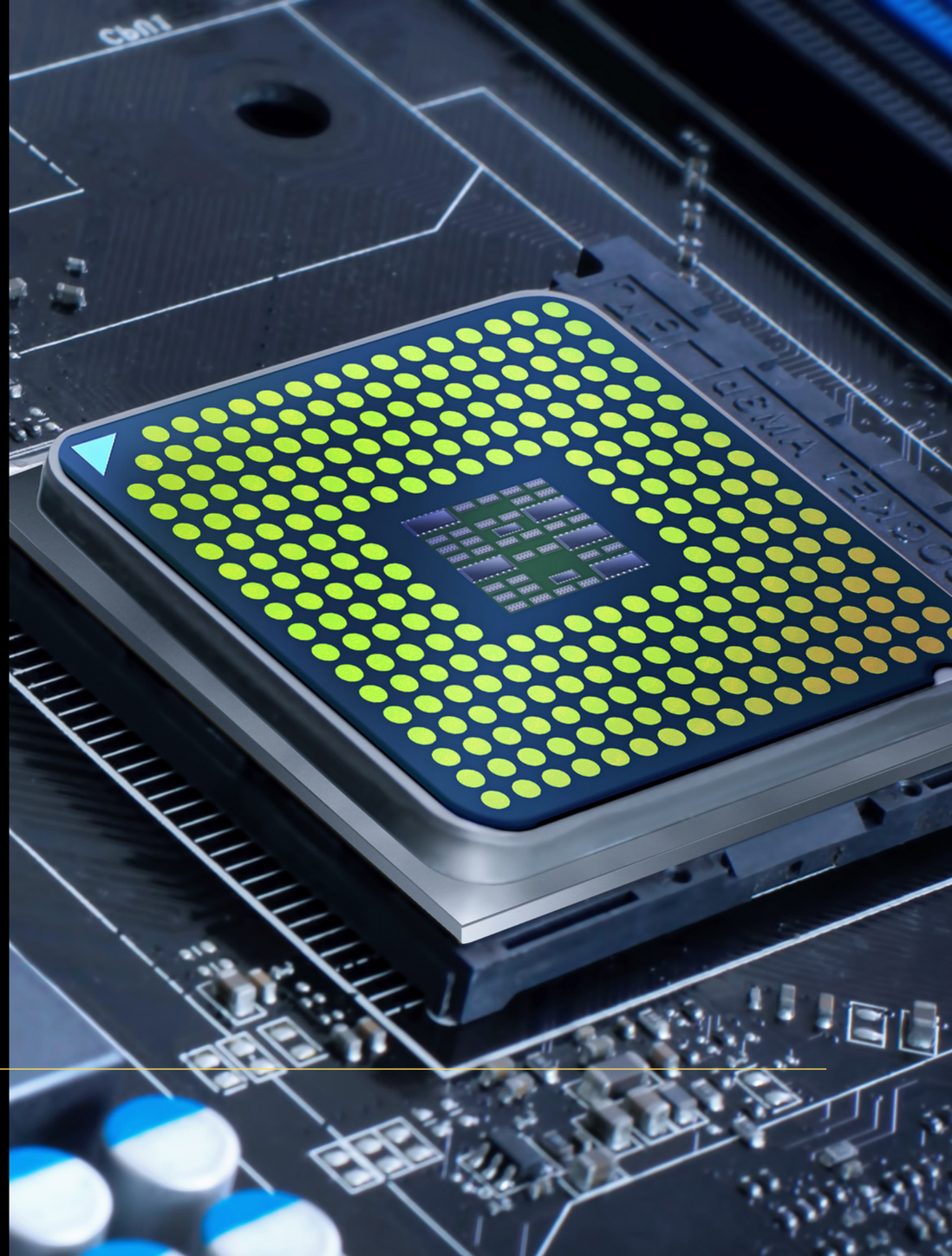
ANJAN
B221028

UDAY
B221064

8086 µP

8086 is a 16-bit microprocessor, meaning it processes data in 16-bit chunks. This architecture allowed for more efficient manipulation and processing of data compared to its 8-bit predecessors.

The 8086 features a 20-bit address bus, allowing it to address up to 2^20 (1 MB) of memory.This increased address space was a significant improvement over earlier processors, enabling more extensive and sophisticated applications.
Memory is segmeted.



MPI LAB

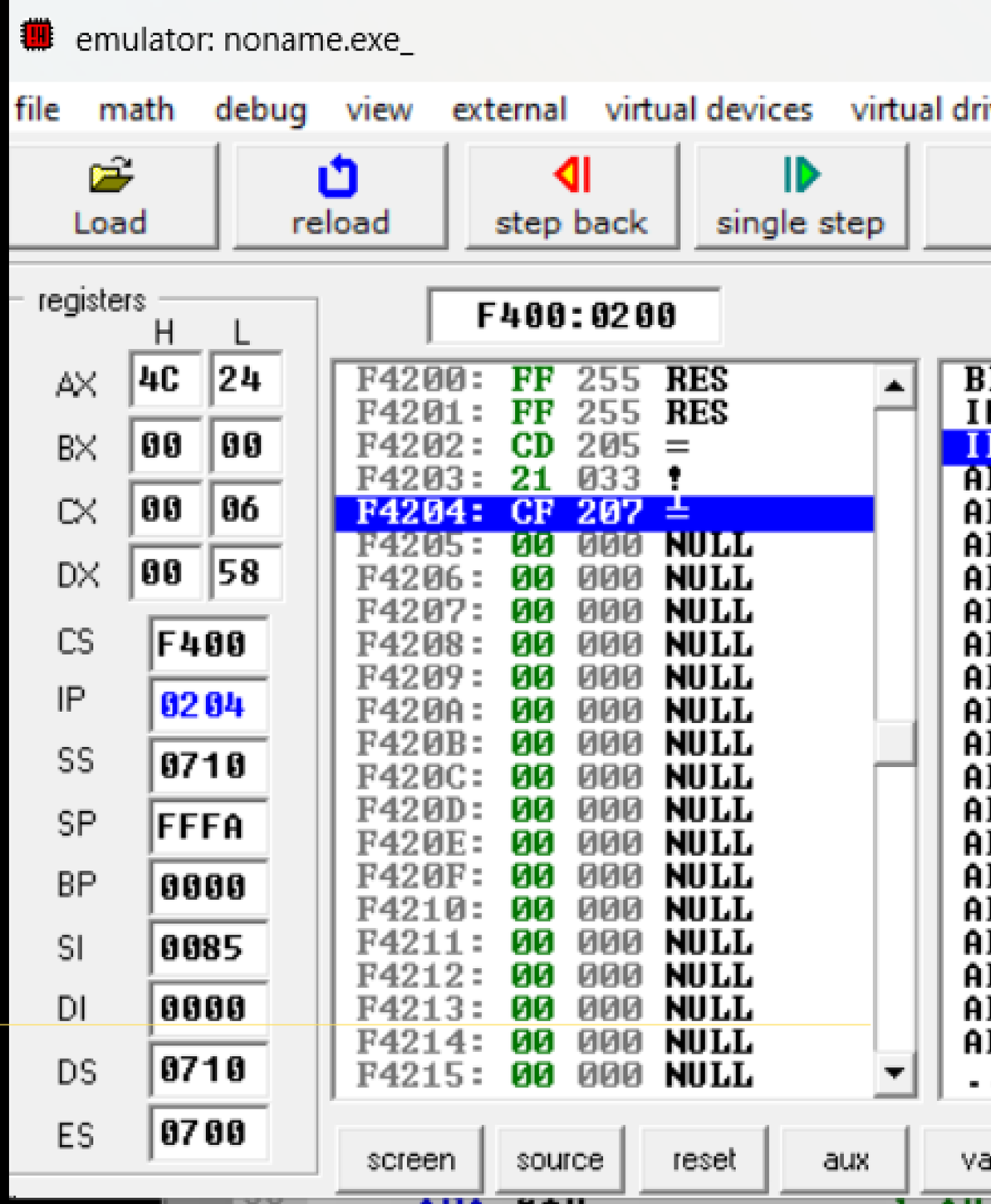
group 6

Digital Dice

```
DATA SEGMENT
        COUNT DB 0
       CHOICE DB ?
        RINT DB ?
       MSG1 DB 10,13,"ENTER 1-ROLL THE DICE 0 -TO STOP$"
       MSG2 DB 10,13,"ENTER YOUR CHOICE$"
       MSG3 DB 10,13,"RANDOM NUMBER GENERATED IS $"
       MSG4 DB 10,13,"-------EXIT !!!!------$"
      MSG5 DB 10,13,"BINGO$"
      REC DB 100 DUP(0)
DATA ENDS
```

MPI LAB

group 6



Code Segment

```
START:
   MOV AX,DATA
MOV DS,AX
LEA SI,REC
INC SI
INC SI
INC SI
   UP:
LEA DX,MSG1      ;To display message MSG1
MOV AH,09H
INT 21H
LEA DX,MSG2      ;To display message MSG2
MOV AH,09H
INT 21H
MOV AH,01H       ;To read the choice
INT 21H
MOV CHOICE,AL     ; To Store the value of choice
CMP AL,'0'       ; Compare the entered choice with ASCII value of 0
JE EXIT          ; If equal to 0 then Exit ( Terminate the program )
LEA DX,MSG3      ; To display message MSG3
MOV AH,09H
INT 21H
   MOV AH,2CH
   INT 21H          ; Interrupt used to get the system time
   MOV AX,DX
   MOV DX,0
   MOV CX,6
   DIV CX           ; Divide with 6 in order to get remainder between 0 to 5
   ADD DL,'0'       ; Add the ASCII value 30 to convert value to ASCII
   ADD DL,1    ; add 1 to make it display between 1 to 6 to simulate rolling of dice.
   MOV [SI],DL
   INC SI
   MOV AH,02H       ;To display the Random value generated
   INT 21H
  CMP DL,'6'
  JNE NEXT1
  INC COUNT;
NEXT1:
   MOV BL,[SI]
   CMP BL,[SI-1]
   JNE EX
   CMP BL,[SI-2]
   JE NEXT
   EX:
```

Code Segment

```
  CMP COUNT,3
  JMP UP          ; Continue till the user enters 0

NEXT:
  LEA DX,MSG5
  MOV AH,09H
  INT 21H
  JMP EXIT
EXIT:
  LEA DX,MSG4     ; To display message MSG1
  MOV AH,09H
  INT 21H
  MOV AH,4C H
  INT 21H
CODE ENDS
END START
```