| **AACS3064 Computer Systems Architecture** |
| --- |

1. **Course Details**

Faculty : FOCS

Course : AACS3064 Computer System Architecture

Programme(s) : DCS2 [**KL**], DFT2[**KL**], DDS2[**JH**], DFT2[**JH**]

Sem / A.Y : June 2021 sem

Credit value : 4 (RM178 x 4 credits = RM712)

Contact hours : 2L, 1T, 2P per week

Course Learning Outcomes (CLOs)

CLO1: Apply arithmetic operations, assembly language and basic organisation of digital computers based on the concepts of computer architecture. (C3, PLO2)

CLO2: Demonstrate effective team skill in constructing a program using assembly language (A2 ,PLO4)

CLO3: Construct a program using assembly language. (P3, PLO3)

----------------------------------------------------------------------------------------------------------------------------

1. **Teaching staff**

Lecturer : Ms. Choy Lai Fun, [choylf@tarc.edu.my](mailto:choylf@tarc.edu.my)

Tutor : Mr. Wong Hon Yoon, [wonghy@tarc.edu.my](mailto:wonghy@tarc.edu.my)

Ms. Michelle Lim Sern Mi, [smlim@tarc.edu.my](mailto:smlim@tarc.edu.my)

Mr. Calvin Lim Seng Wah, [P3433@tarc.edu.my](mailto:P3433@tarc.edu.my)

Mr. Cho Wee Hing, [P3347@tarc.edu.my](mailto:P3347@tarc.edu.my)

Mr. Chong Jing Wen, [P3583@tarc.edu.my](mailto:P3583@tarc.edu.my)

Mr. Loh Kiean Nyah, [P2537@tarc.edu.my](mailto:P2537@tarc.edu.my)

----------------------------------------------------------------------------------------------------------------------------

1. **Assessment mode**
2. In course spec

Component Contribution Remarks Schedule

Continuous Assessment 50% Test + Assignment = 100m Week 12

Final Assessment 50% 4Qs at 25m each = 100m Exam period

1. Approved Assessment due to MCO

Component Contribution Remarks Schedule

Continuous Assessment 50% Assignment = 100m Week 12

Final Assessment 50% 4Qs at 25m each = 100m Exam period

----------------------------------------------------------------------------------------------------------------------------

1. **Topics covered**

Chapter 1: Numbering System

Chapter 2: Representing Numerical Data

Chapter 3: Addressing Data in Memory & Segment

Chapter 4: Computer Architecture

Chapter 5: Machine Execution

Chapter 6: Assembling, linking and executing program

Chapter 7: Assembly Language Fundamental I

Chapter 8: Assembly Language Fundamental II

Chapter 9: Conditional Processing

Chapter 10: Keyboard & Screen Processing

Chapter 11: I/O Facilities

Chapter 12: Performance Enhancement

----------------------------------------------------------------------------------------------------------------------------

1. **Tools**

Dosbox 0.74

MS VSCode

----------------------------------------------------------------------------------------------------------------------------

1. **Makegood Opportunity**

* Threshold for coursework: 50 marks
* To be eligible for the make-good test, students have to :
  + Demonstrate serious efforts by participating in classes, attending test(s), doing practical questions, and submitting assignments.
  + Posses satisfactory attendance for classes.
  + Score at between 40% - 49% on their coursework marks.
* Prior to Make Good: Students involved will be informed during the class / via Google Classroom
* During Make Good: Complete the tasks(s)/test assigned within the stipulated time frame
* Post Make Good: To pass the coursework, students MUST have to obtain at least 50% from the make-good test.
  + The highest mark obtainable after a make-good is the pass mark, i.e. 50.
  + Students who do not pass their coursework would be required to REPEAT the Course.

----------------------------------------------------------------------------------------------------------------------------

1. **Attendance**

* Students' attendance will be marked based on students' attendance during the classes via Google G Suite, inclusive of students' participation.
* Student with poor attendance / participation may be barred from final examination / final

online assessment / e-assessment. As a result, the student is required to REPEAT the course when it is offered. (Note: You may check Intranet > News and Notices > Schedule for Main, Resit and Repeat Courses for Bachelor Degree Programmes, for more info.)

----------------------------------------------------------------------------------------------------------------------------

1. **E-learning Support**

* Via Intranet
  + Link: https://sites.google.com/student.tarc.edu.my/elearningstudent/home
* Via Student Info Centre (SIC)
  + Link: https://sites.google.com/tarc.edu.my/focs-students/home

----------------------------------------------------------------------------------------------------------------------------

1. **Make Appointment for Consultation hour**

* Via email
* Via google classroom
* Via google chat

| **Lesson Discussion** |
| --- |

Place your question here if any.

| **28/6/2021, Monday, 10 -12pm (Replacement class for Week 1)** | | |
| --- | --- | --- |
| Programme  (e.g.: DCS/DDS/DFT) | Student’s ID | Question  (Insert your question here.) |
| DCS | KhongHX | When it comes to converting from decimal to binary/octal/hexadecimal, can we write it on a paper or on a drawing program, take a picture and upload instead of typing it like what miss is doing?  Yeah, can we do it in a paper first, then upload it as our answer? Thanks  → division / working is required. |
| DCS | ChongDM | Miss, the participation form needs to submit what ans? I was late, sorry :  → gform |
| **29/6/2021, Tuesday, 4-6pm (Replacement class for Week 1)** | | |
| DFT | ChongJY | So no mid-term test?  The answer can be in a text file?  We need to include workings in ok  → 21 base 5 to dec  = (2 x 51)+(1 x 50)  → Coursework is AssignmentIon  → quiz (L) == participation  → answer for participation can be in the tex file / .doc /docx / pdf /jpeg  So we will be assessed from base 2-16?  ---> Possible to include other bases to assess student’s understanding (9/7/2021)  16+9 is like when 6+9 = 15(exceeded 9), hence 15 -10, and 1+1 = 25? Same goes to BCD, when 7+7 = 14(exceeded 9), so 14-10=4, 1+1=2, then 2+1 = 3?  14(1110)-10(1010)=4(0100)  → Yes. You are right. Same concept. (9/7/2021)  Actually I am not so clear with the table on page 8 of the slide.  → That is to explain that larger values require more bits in binary. Therefore, BCD is an alternative option. (9/7/2021)  How do miss get -64+29? When doing the 2^7-->2^0?  128 -- 64 -- 32 -- 16 -- 8 -- 4 -- 2 -- 1  =27 26 25 24 23 22 21 20  This may help to ease translation. (9/7/2021) |
| DFT | leoing | miss where is the question for the participation?  → refer below, as highlighted in yellow |
| DCS2 | TanEL | The ans also need to in Hex form right?  → Show your working. Final answer in Hex. |
|  |  | I think the stream got delay  ??  yes about 5-10 seconds for me  Oic. Network congestion and other factors may also cause live streaming issues, which can delay your stream. Delays can happen even when you have a great network that can sustain your average streaming bitrate. |
| **1/7/2021, Thu, 2-4pm (Week 2)** | | |
| DCS2 | 2002621 | So BCD is only used for large numbers right?  → Good to use for computation in large value. As per sample, the result returned shall be the same as conventional binary. If the question requires student to use BCD in computation, the question must explicitly state (else use normal method)  It is not harm to use BCD for small number ?//okok *\*\*miss that one is last replacement class’s comment → Thks*  I think we are recommend to use when need a long division to convert a decimal into binary>save time  → Just one concern, many users forgot the additional step for the BCD value more than 9. |
| DCS2 | TanEL | BCD is the alternative for Decimal or Binary like we dont want to use binary then use BCD?  It's like a preference or in certain situations?  → Alternative for binary. You may opt to use conventional binary (BIN) like chapter 1 or BCD. If the question requires students to use BCD, the question must specify.By default, use conventional binary. |
| DCS2 | 2002683 | What do we use bcd for?  → Good to use for computation in large value. |
| DFT2 | 2003059 | Where to find the range (-128d to +127d)?  → Range or an 8 bit system:  Largest +ve number: 01111 1111B (+127D)  Smallest -ve number: 1000 0000B (-128D) |
|  | I think it’s down there | Submit in pdf? Just answer right not pdf also can gua, tcr didn't mention  → Mentioned in the earlier question. |
| DCS2 | diongzy | Miss, can explain more about overflow and carry?   |  | OF | CF | | --- | --- | --- | | Occur when | Not fit into destination | exceeded destination | | Used in | Signed number | Unsigned number | | Detected by | Sign of result is opposite of operands | Extra 1 bit generated | |
| DCS2 | 2003998 | BCD can only convert base-10 to base-2 ? Can we use to convert base-8 to base-2 ? Like changing the largest value to 8, then largest bcd is 0111. If exceed 0111 then minus 1000. Can have something like that?  → BCD is **B**inary **C**oded **D**ecimal. TO concert Decimal value to binary (BCD) only. Not applicable in other numbering system. |
| DCS2 | 2003133 | Can we directly use the calculator to convert base-10 to base-2 during the exam?  → In such a case, no mark could be awarded to working. Usually the final answer carries 1 mark only. |
| **8/7/2021, Thu, 2-4pm (Week 3)** | | |
| DFT | 2010518 | If the highlighted 3 become 8 do we need to round off to let the answer become 1 37 21378?  → Round up is required. |
| DFT | 2003059 | why 7-9 = 1, 3-1=8 and so on….  → It is 109190 - 021375 due to the sign used. |
|  | Please provide your programme & ID so we could record your participation during lecture class too. Thks. | Miss, later can upload this pdf onto google classroom? Including the drawing/handwriting if can :) TQ  → Will do. Stay tuned. Tq miss  → Please refer to the excerpt of the PDF as belod for revision ya. (9/7/2021) |
|  | Please provide your programme & ID so we could record your participation during lecture class too. Thks. | What participation question ? :D  → No quiz today. Attendance based on G. Attendance. |
|  | Please provide your programme & ID so we could record your participation during lecture class too. Thks. | So today no participation quiz?  → No quiz today. Attendance based on G. Attendance. |
| DFT | 2006341 | For the scientific notation, the front of floating point must be zero? Eg. (a) Convert 55220311 to scientific notation.  Answer : 0.55220311 x 10^8  → Standard format, starting with “0.” |
|  |  | Has anyone received an assignment question?  → Check with tutor |
| **15/7/2021, Thu, 2-4pm (Week 4)** | | |
| DCS2 | 2003859 | Miss, can you please put the notes in a different google docs, and set it to non editable? Sometimes other students would edit it accidentally and just leave it there :) no offence and tqvm  → Thank you for your suggestions |
| DCS2 | 2002621 | Miss, about the calculation of the FETCH-EXECUTE INSTRUCTION CYCLE, if the question asks us to find a location value which is located in the center of that location range given, should we find and calculate from the start, or we can just calculate the value from the middle?  I mean just like the quiz today, we are asked to find location 21, but the question starts from 20, is it okay to calculate straight starting from 21?  → You may write the whole working. However, marks will be awarded to the required part as per question only.  \*I think that some photos are missing at part 7…calculation there?  → Will consider to move all the pics to an un-editable gdoc for the complete mind map soon. |
| **22/7/2021, Thu, 2-4pm (Week 5)** | | |
| DFT2 | 2001369 | No quiz??  → No quiz for week 5. |
| DFT2 | 2001232 | Assumption in the assignment is like an additional function that you think by yourself?  → Assumption is like business rules / program limit. E.g.: User input must be 3 digits. |
| DFT2 | 2003333 | What is the purpose of our assignment?  → The purpose of the assignment is to allow students to better understand the internal operation of the computer, the process of machine cycle and computer architecture. But using registers & memory, it mimics the machine operation.  By understanding the computer operation, then a technical person could better manage the computer resources for efficient operation. |
| DCS2 | 2002903 | Miss, about the participation questions, may we have the answer or a simple discussion after every class for checking purposes?  → Thanks for the suggestion. Will do in the coming weeks. |

| **29/7/2021, Thu, 2-4pm (Week 6)** | | |
| --- | --- | --- |
| DFT2 | 2001369 | Miss, I found this error after I added the code. How to fix it?  START:  CMP COUNT, 3  JE EXIT |
|  |  | Check line 43. |
|  |  |  |
|  |  |  |

| **5/8/2021, Thu, 2-4pm (Week 7)** | | |
| --- | --- | --- |
| DFT2 | 2001369 |  |
| DCS2 | 2010524 | What's the difference between 13, 10 and 0DH, 0AH  Decimal and hexa  → 13,10 are in Decimal  → 0DH, 0AH are in Hex |
| DCS2 | 2002892 | What is the difference between FAR and NEAR? When do we use it?  → FAR and NEAR may be used to pass control, such as using JMP, CALL, etc from one function / procedure to another. |
| DFT2 | 2001232 | What is the text macro in the pwp?  → Text macro is a text procedure / function where it could be called dynamically for reference. |

| **12/8/2021, Thu, 2-4pm (Week 8)** | | |
| --- | --- | --- |
| DCS2 | 2003859 | In **3. Conditional Loop**, ***Sums the value in an array***,  How does the program know that JNZ is for CX without comparing? Is it because DEC CX is in the previous line of JNZ A20?  → Yes. tq miss |
|  |  |  |
| DFT2 | 2001232 | Miss, does the JMP instruction can let the dosbox to run the code in another asm file?  ---> Best to use the “call” function. |
|  |  |  |

| **19/8/2021, Thu, 2-4pm (Week 9)** | | |
| --- | --- | --- |
| DCS2 | 2011275 | Miss, for our assignment, how can we combine with our teammates functions?Is it we need to do all things in one file?  → Possible to use the “Call” function. |
| DFT2 | 2001232 | Miss, is it possible to use 09H to display the string entered in the 07H?  → Possible. Make sure you insert “$” for the last input byte so the 09H (output string) knows where to stop. |
| DFT2 | 2004866 | Miss, how to input more than one character and digit in one command?  → Use 0AH for the input string. |
| DFT2 | 2003059 | Miss, is it possible to have validation? Eg: name cannot have number  → Possible.  CMP input,”0”  JAE check2  JB error  Check2:  CMP input,”9”  JBE validNum  JA error |
| DFT2 | 2004938 | Today don’t have any participation questions?  ---> No for week 9 as today we have walked through a few coding sessions together. |

| **26/8/2021, Thu, 2-4pm (Week 10)** | | |
| --- | --- | --- |
| DCS | 2003741 | Miss, can you provide the sample answer for week10 attendance quiz? Thank you  ---> Will do. Stay tuned. |
|  |  |  |
|  |  |  |
|  |  |  |

| **2/9/2021, Thu, 2-4pm (Week 11)** | | |
| --- | --- | --- |
| DCS | 2010524 | Hit ratio is recorded but what exactly is the use of it?  → Hit ratio could be used to calculate the latency involved. Higher hit ratio results a min delay (latency). And, it helps to optimize the resource utilization too. |
| DCS | 2000715 | Is virtual memory the same as virtual Ram?  → Similar concept. Larger program is not limited to smaller memory capacity for execution. However, some books refer them differently. So, we follow to this reference here (virtual memory a.k.a virtual storage) for standardization. |
|  |  |  |
|  |  |  |

| **9/9/2021, Thu, 2-4pm (Week 12)** | | |
| --- | --- | --- |
| DCS | 2001900 | For calculating Mantissa, converting from dec to bin, oct and hex. How many decimal point should we put, or can we put as many decimal as we want?  → 3 decimal places |
| DCS | 2000715 | Ms can u update the google docs to the version u present just now? Some item is missing  → Thanks for alerting.  → It is all in now. |
| DCS | 2010524 | Same with above, miss, can you upload the summary.docx separately so no student can edit the notes?  → Will lock for edit soon  → Edit mode only during class. |
|  |  |  |

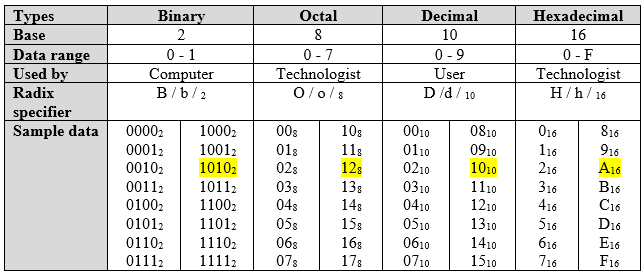
| **16/9/2021, Thu, 2-4pm (Week 13)** | | |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

| **23/9/2021, Thu, 2-4pm (Week 14)** | | |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

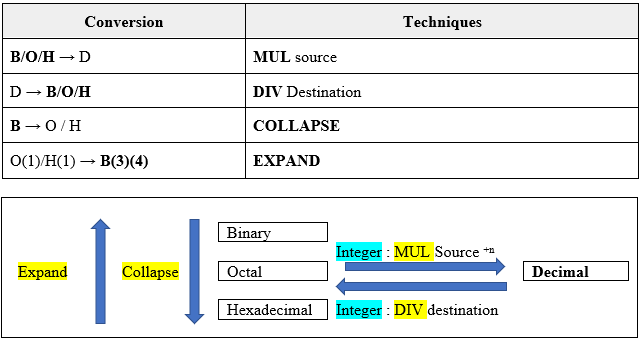
**Note: Refer to below to the Question for participation.**

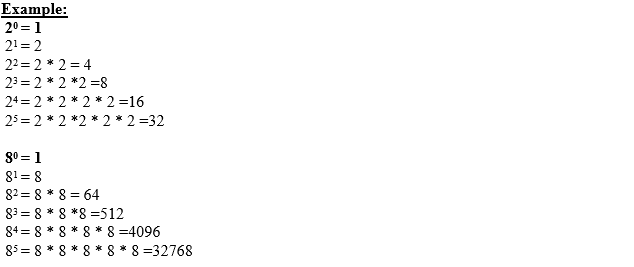
| **Chapter 1: Numbering System**   1. **Numbering System** 2. **Conversion** 3. **Arithmetic Computations** |
| --- |

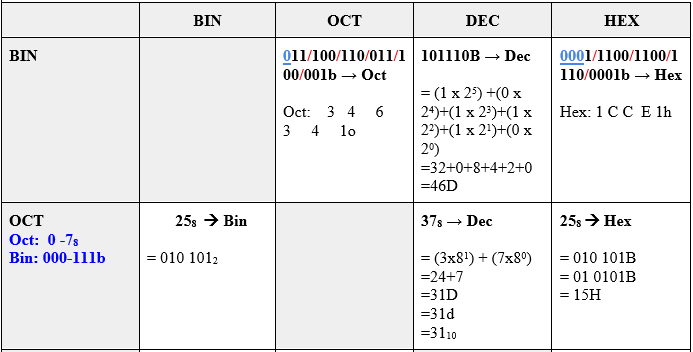
1. **Numbering System**

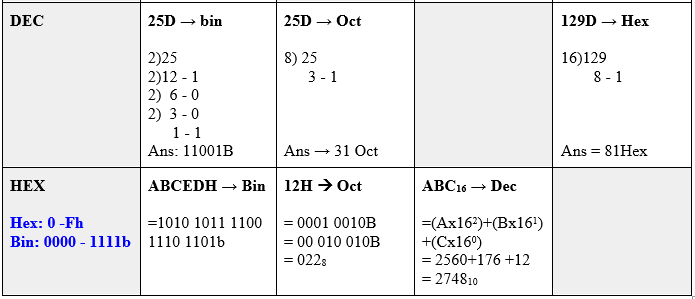


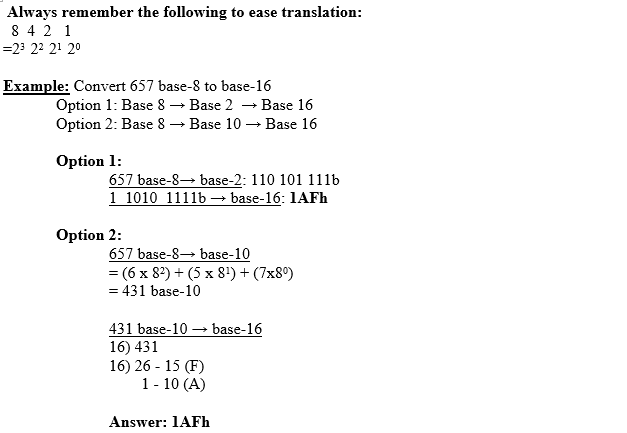
1. **Conversion**



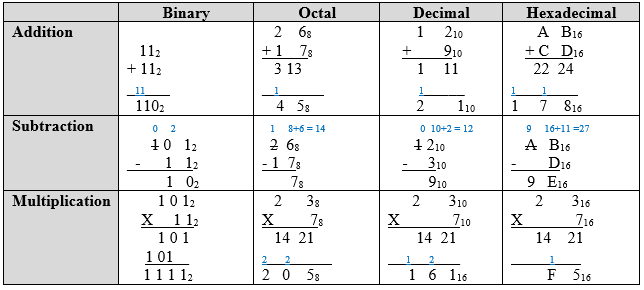






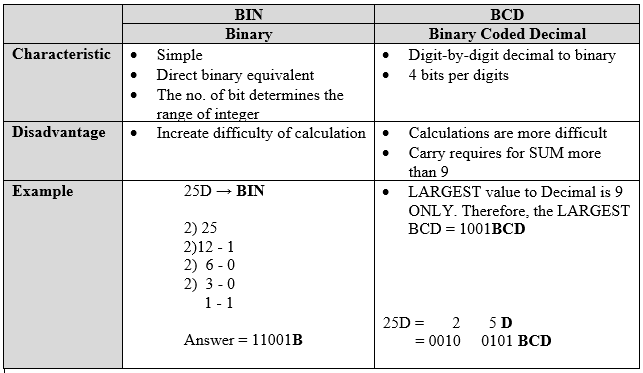


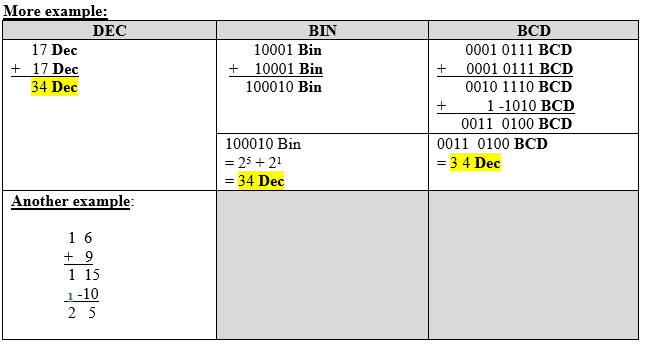
1. **Calculation**



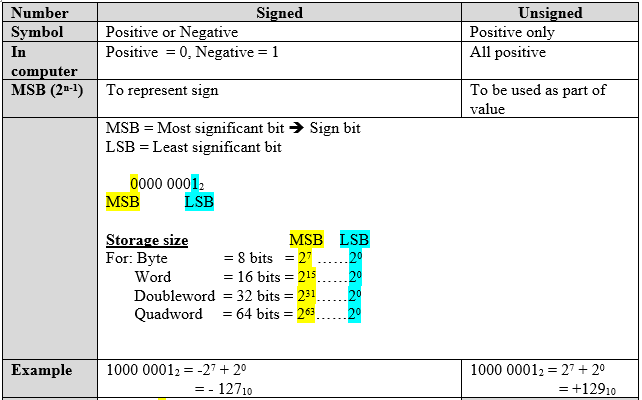
| **Chapter 2: Representing Numerical Datad**   1. **Unsigned & signed** 2. **Signed integer representation** 3. **Floating point number** |
| --- |

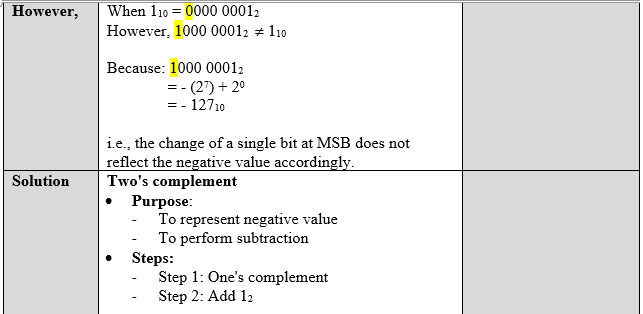
1. **Unsigned Binary & Binary Coded Decimal (BCD)**

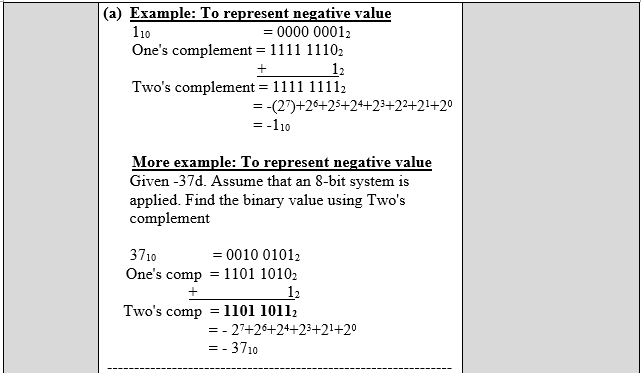


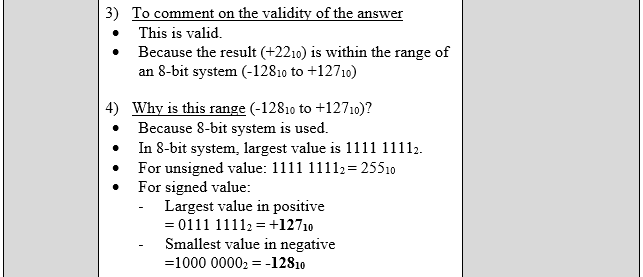


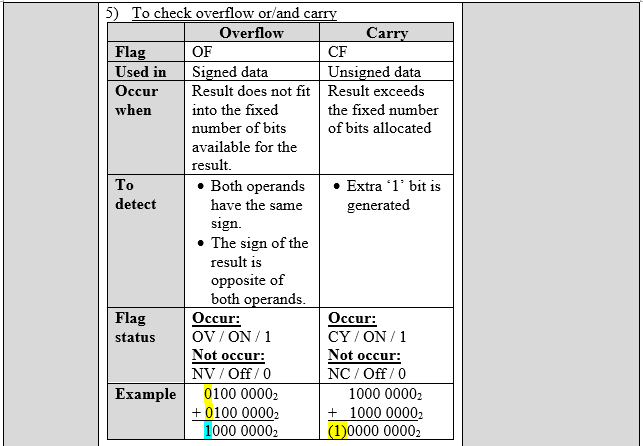
1. **Signed integer representation**

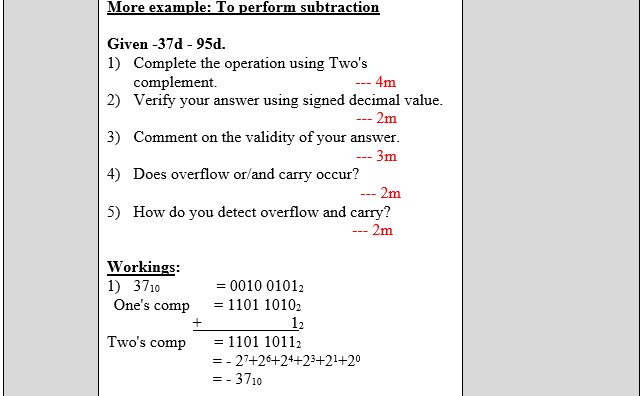


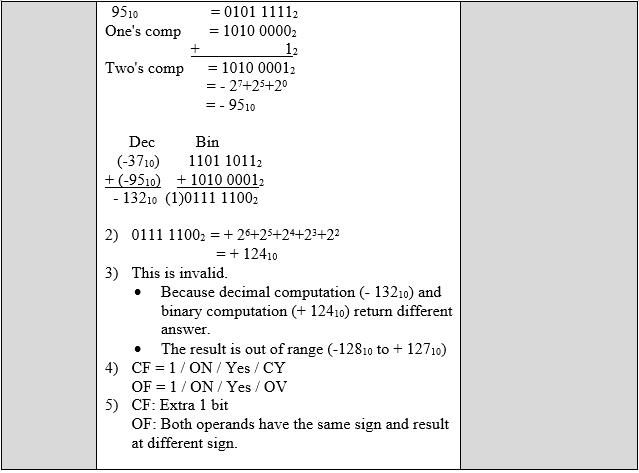






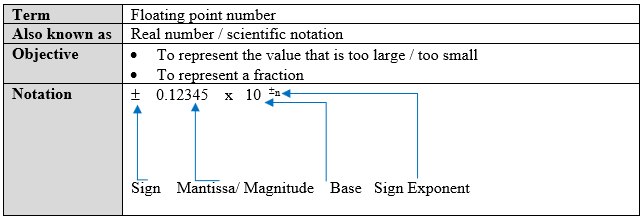




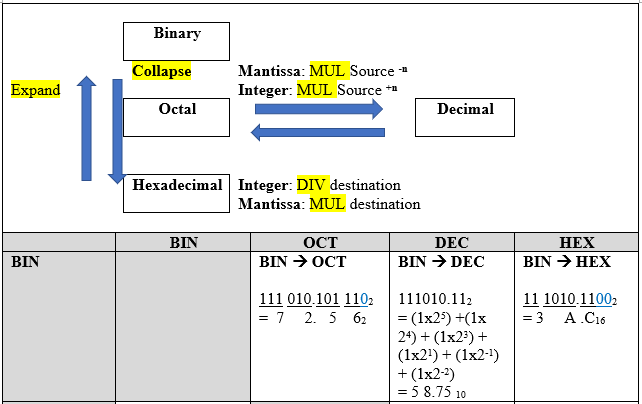


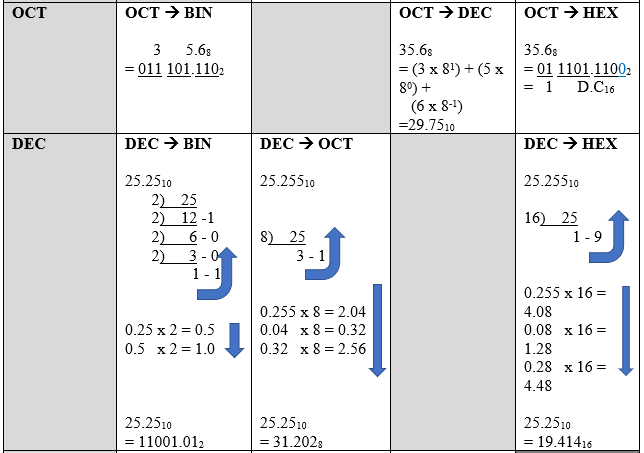
1. **Floating Point Number Representation**

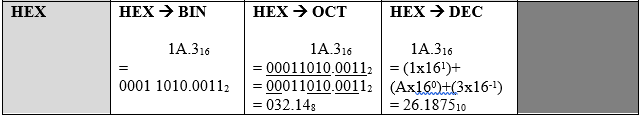
**3.1 Floating Point Number**



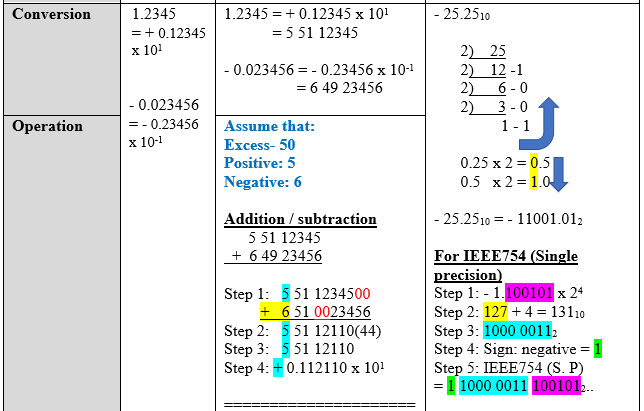
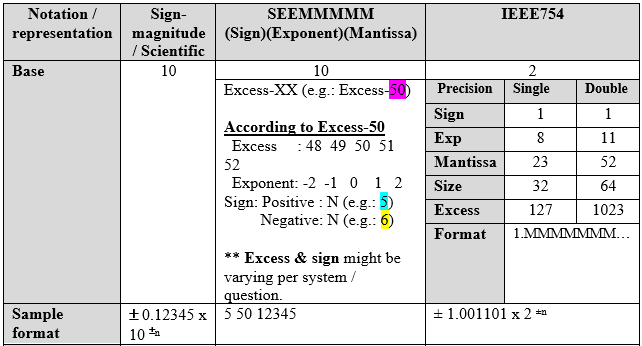
**3.2 Conversion**

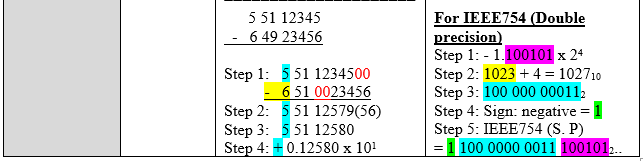
****

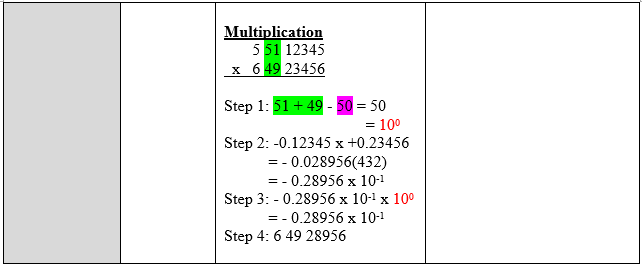
****

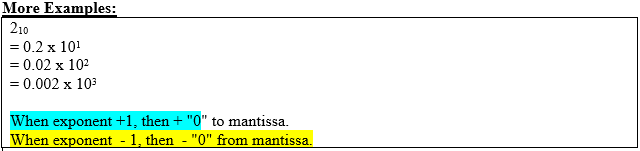
****

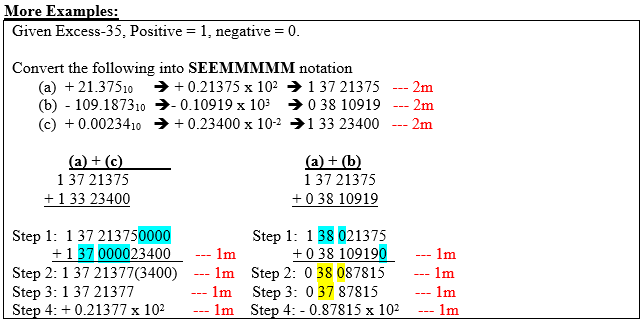
**3.3 Representation**

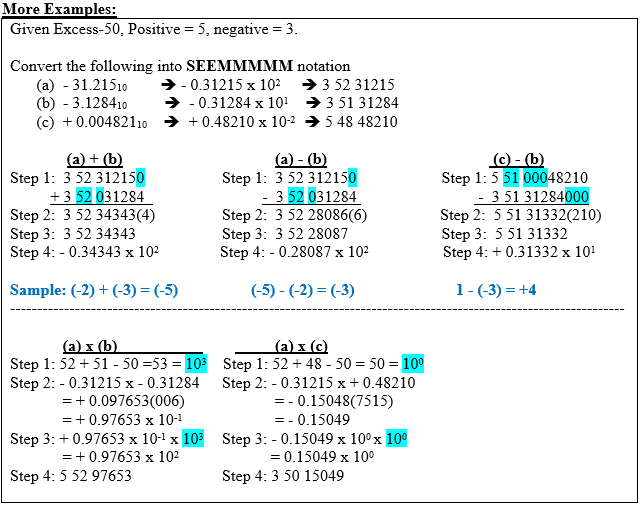


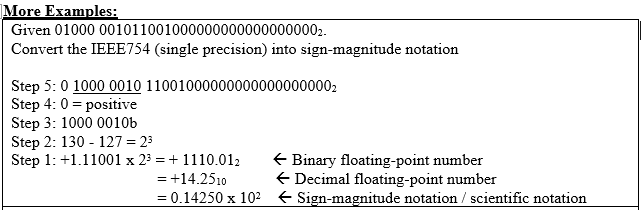






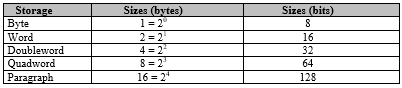






| **Chapter 3: Addressing Data in Memory Segment**   1. **Data STorage sizes** 2. **Data addressing** 3. **Segmented memory segment** 4. **Data registers** |
| --- |

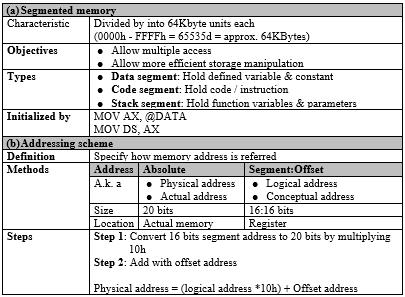
1. **Data storage**

****

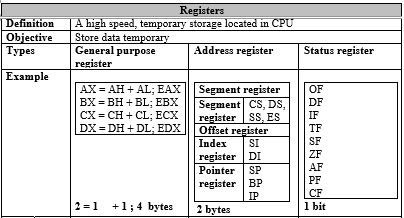
1. **Data addressing**

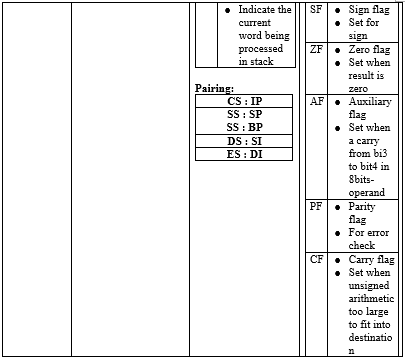
****

1. **Segmented memory segment**

****

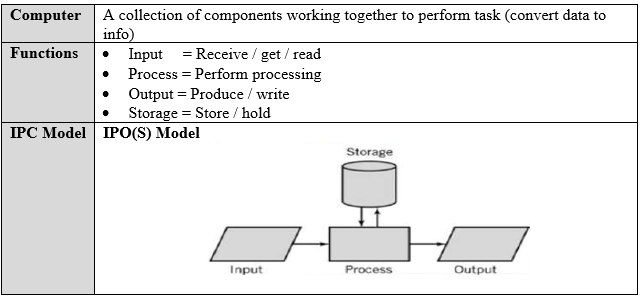
1. **Data registers**



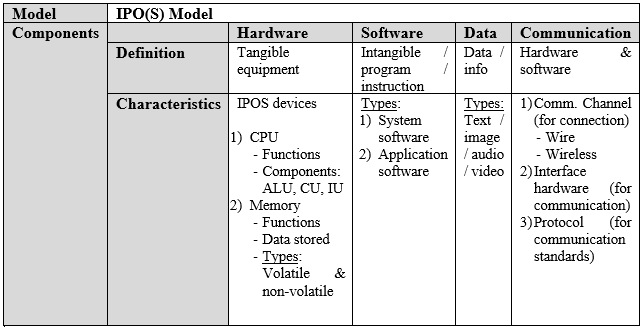


| **Chapter 4: Computer Architecture**   1. **The IPO(S) Model** 2. **Major components of Computer Systems** 3. **Buses** 4. **The components of the CPU** 5. **The Concepts of Registers** 6. **The Memory Unit** 7. **The Fetch-Execute Instruction Cycle** 8. **CISC and RISC Architecture** |
| --- |

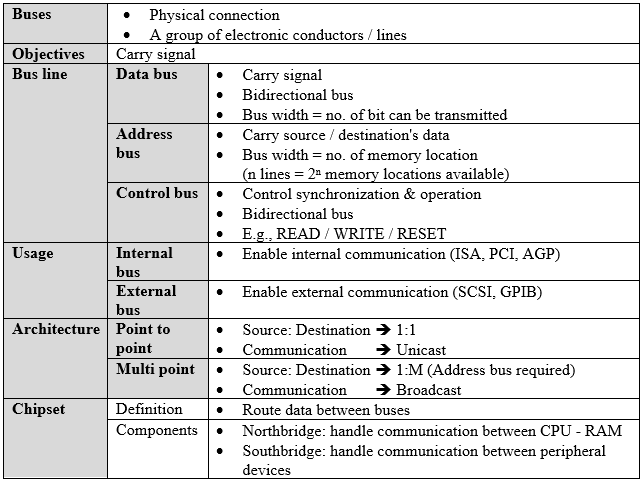
1. **The IPO(S) Model**



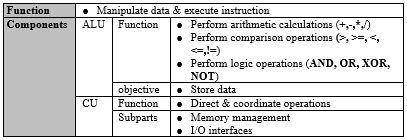
1. **Major components of Computer Systems**

****

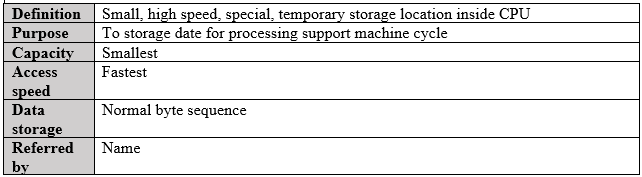
1. **Buses**

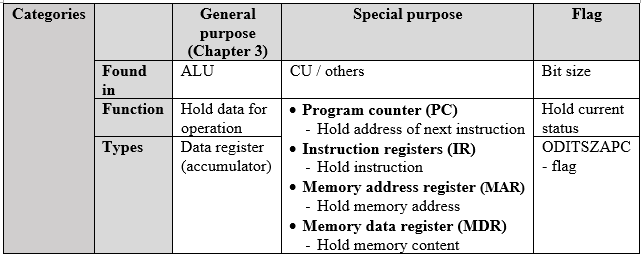
****

1. **The components of the CPU**

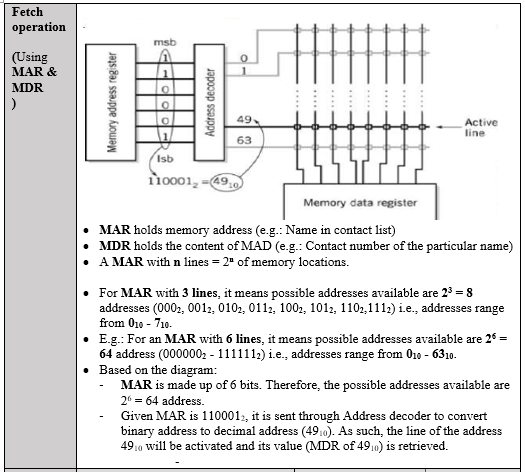
****

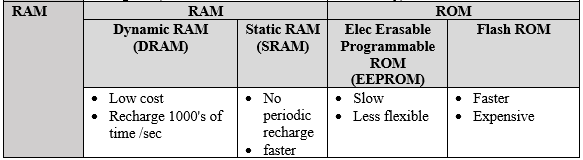
1. **The Concepts of Registers**

****

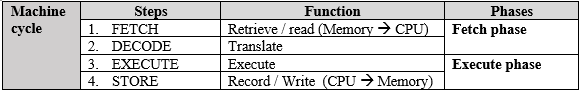
****

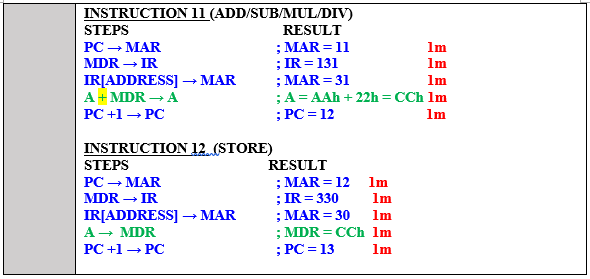
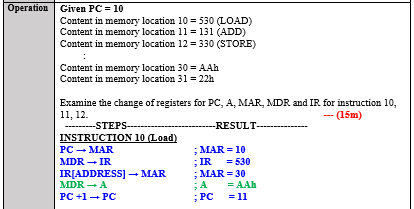
1. **The Memory Unit**

****

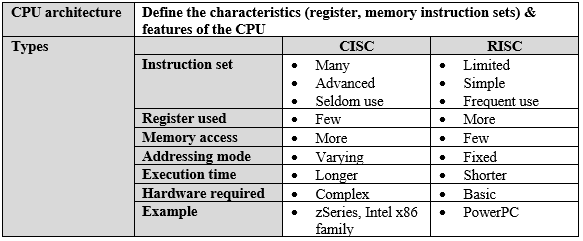
****

1. **The Fetch-Execute Instruction Cycle**

****

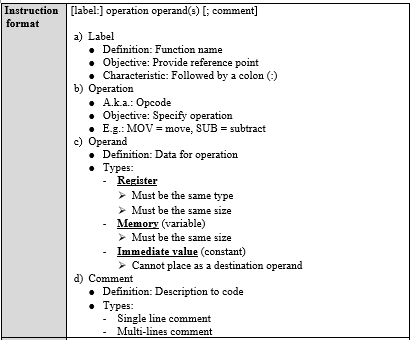
****

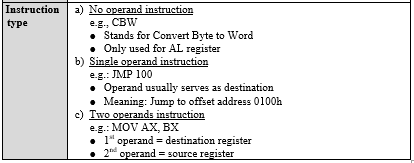
1. **CISC and RISC Architecture**



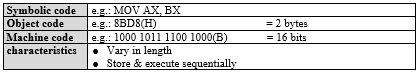
| **Chapter 5: Machine Execution Using Debug Program**   1. **Instruction format** 2. **Machine language instruction** 3. **Instruction execution** 4. **Debug Program** |
| --- |

1. **Instruction format**

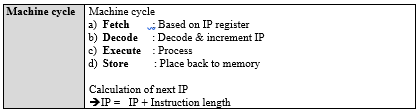
****

****

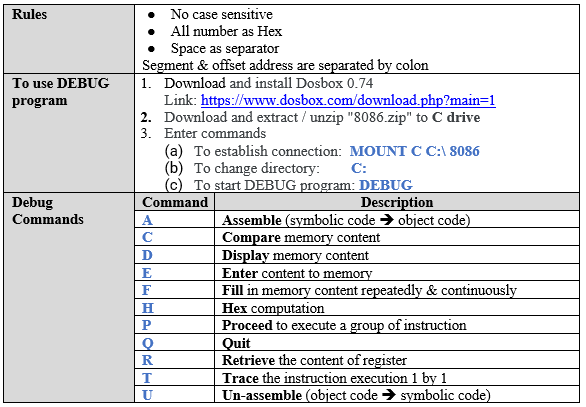
1. **Machine language instruction**

****

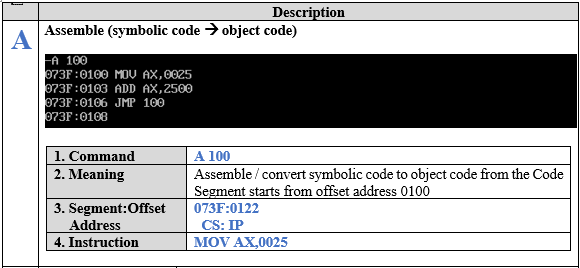
1. **Instruction execution**

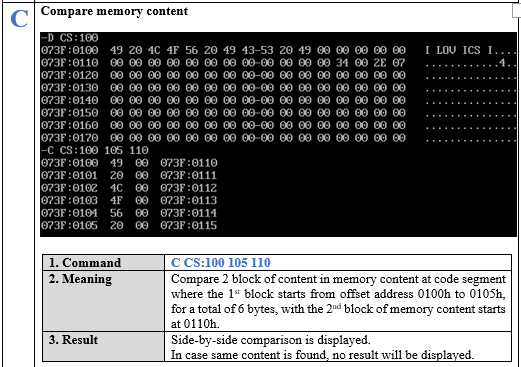
****

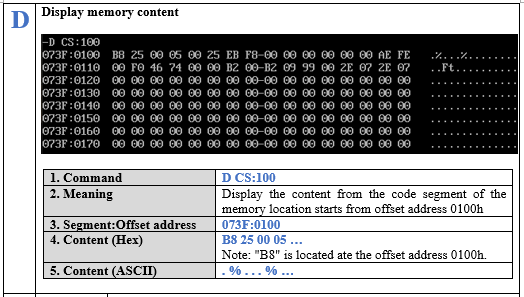
1. **Debug Program**

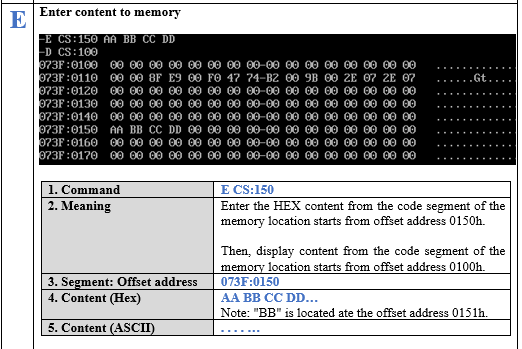


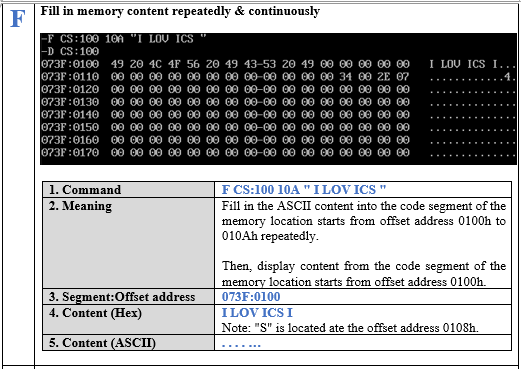
1. **Sample output**

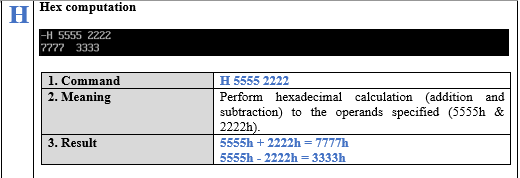
****

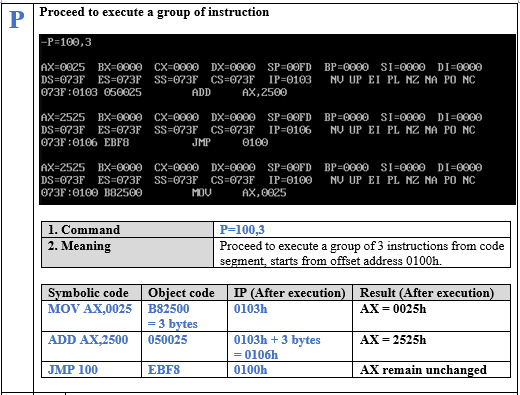
****

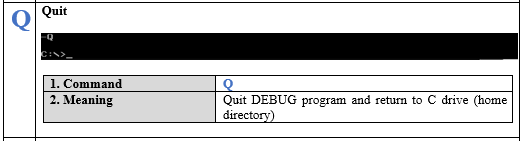
****

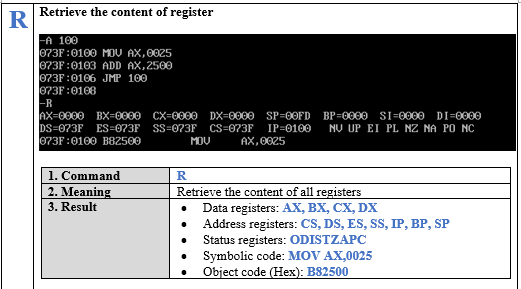
****

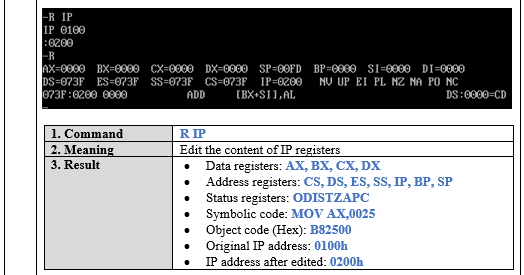
****

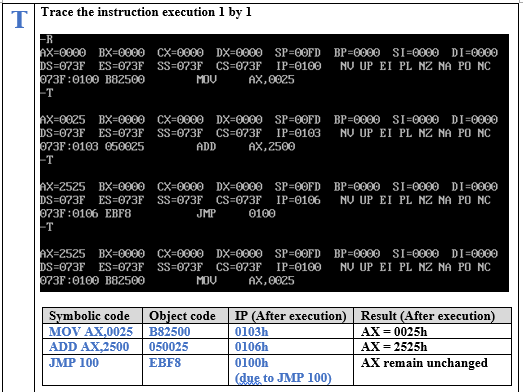
****

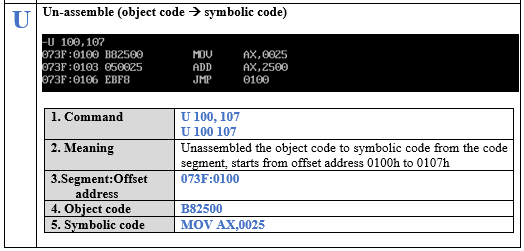
****





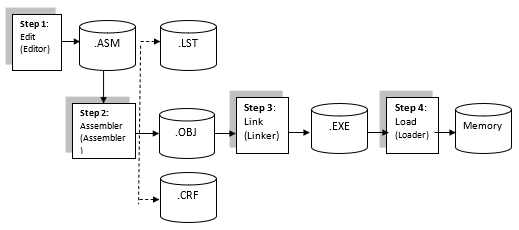


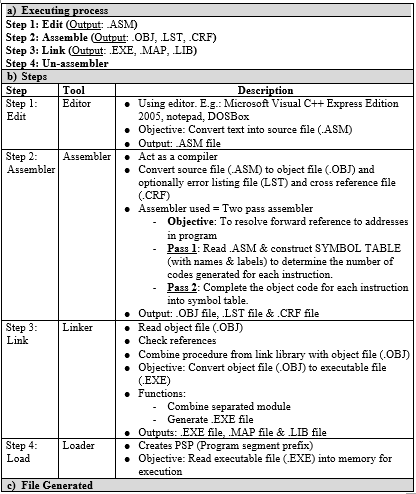


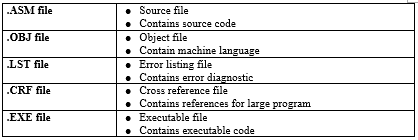


| **Chapter 6: Assembling, Linking and Executing Program**   1. **Preparing program for assembling & execution** 2. **Assembling a source program** 3. **Linking an object program** 4. **Executing a program** 5. **Error diagnostics** 6. **The assembler program counter** |
| --- |

1. Preparing program for assembling & execution
2. Assembling a source program
3. Linking an object program
4. Executing a program





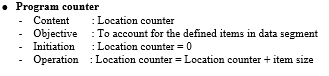


1. Error diagnostics

● Contains Line Number & Explanation

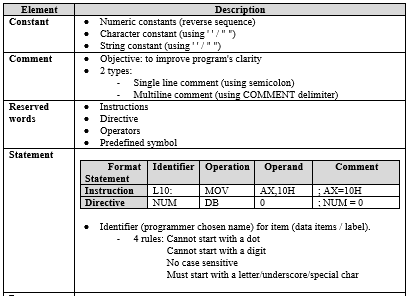
Use **–td** command in Debug program to trace & debug Assembly Language Program

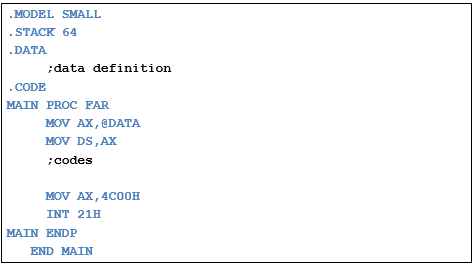
1. The assembler program counter



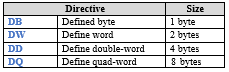
| **Chapter 7: Assembly Language Fundamental I**  **1.** **Basic Elements of Assembly Language**  **2.** **Defining Data**  **3.** **Symbolic Constant**  **4.** **Data Transfer**  **5.** **Arithmetic Instruction**  **6. Display Single Character** |
| --- |

1. **Basic Elements of Assembly Language**

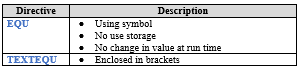
****

****

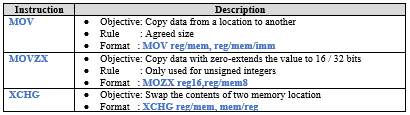
1. **Defining Data**

****

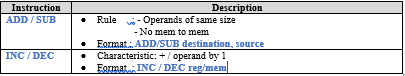
1. **Symbolic Constant**

****

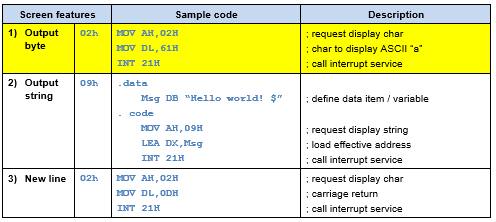
1. **Data Transfer**

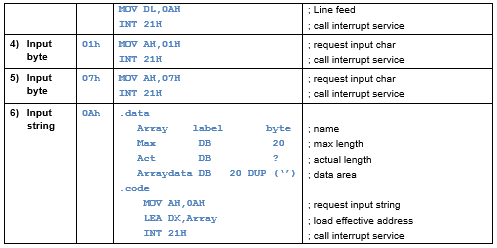
****

1. **Arithmetic Instruction**

****

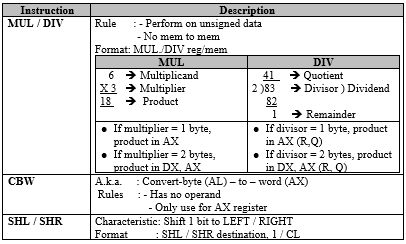
1. **Display Single Character**



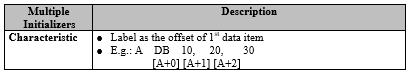


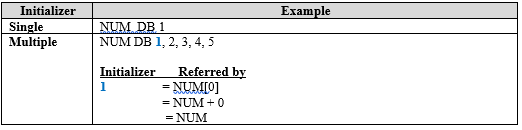
| **Chapter 8: Assembly Language Fundamental II**   1. **Arithmetic** 2. **Multiple initializer** 3. **Direct-offset operands** 4. **Data related operators and directives** 5. **Indirect operands** 6. **Unconditional Jump and Loop** |
| --- |

1. **Arithmetic**

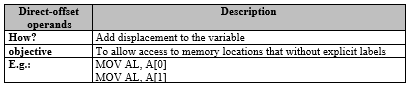
****

1. **Multiple initializer**

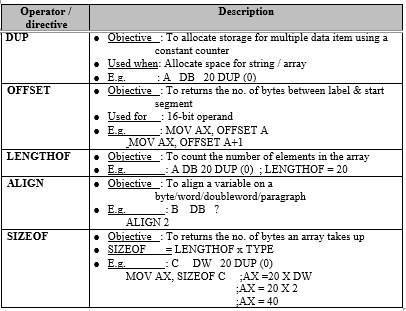
****

****

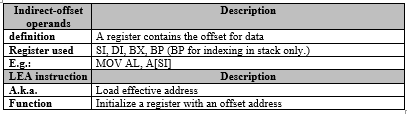
1. **Direct-offset operands**

****

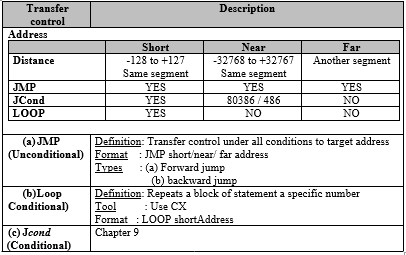
1. **Data related operators and directives**

****

1. **Indirect operands**

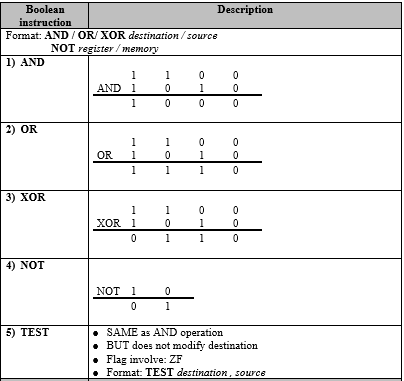
****

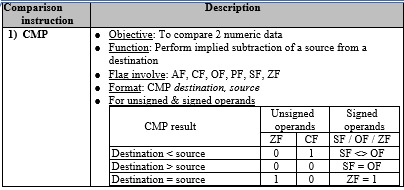
1. **Unconditional Jump and Loop**



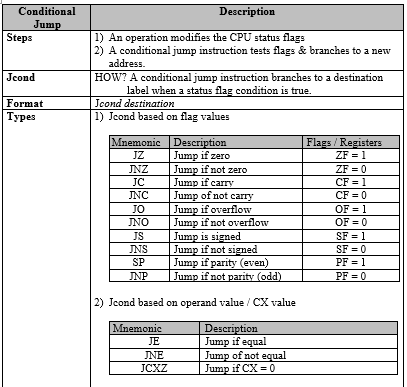
| **Chapter 9: Conditional Processing**   1. **Boolean and comparison instruction** 2. **Conditional jumps** 3. **Conditional loop** |
| --- |

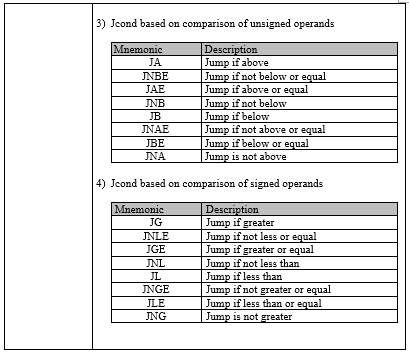
1. **Boolean and comparison instruction**

****

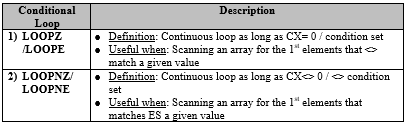
****

1. **Conditional jumps**

****

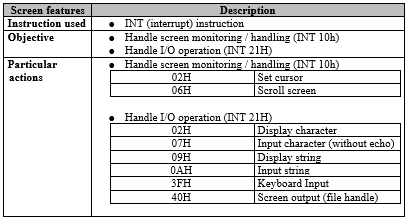
****

1. **Conditional loop**

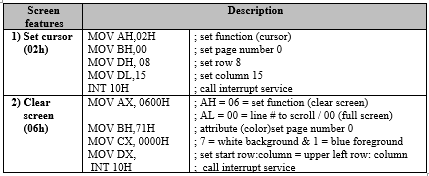


| **Chapter 10: Keyboard and Screen Processing**   1. **Screen features** 2. **BIOS interrupt (INT 10H)** 3. **MS-DOS function calls (INT 21H)** |
| --- |

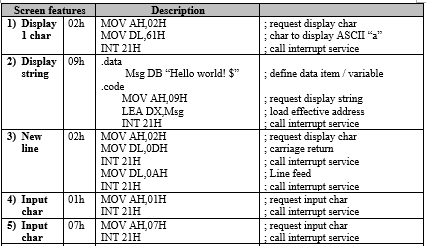
1. **Screen features**

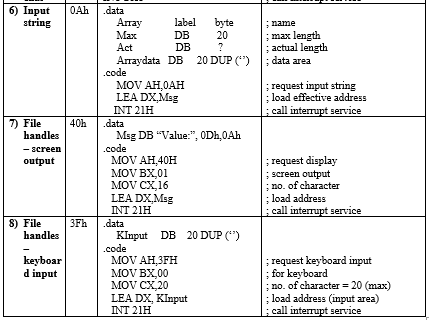
****

1. **BIOS interrupt (INT 10H)**

****

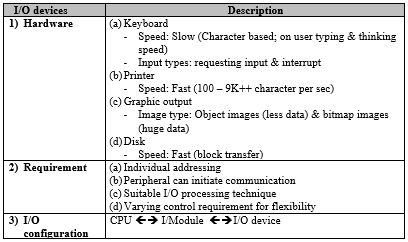
1. **MS-DOS function calls (INT 21H)**



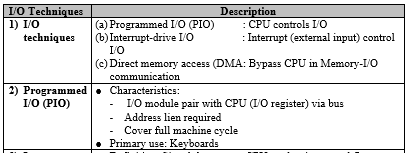


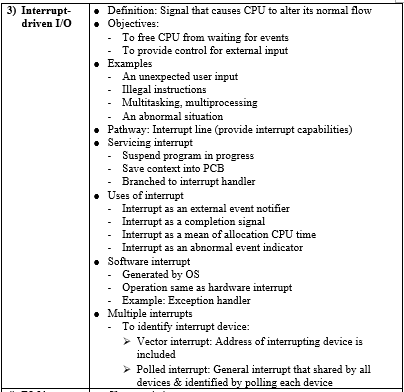
| **Chapter 11: I/O Facilities**   1. **Characteristics of typical I/O devices** 2. **I/O techniques** 3. **I/O architecture** 4. **I/O modules** |
| --- |

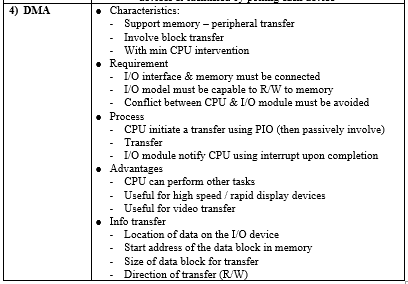
1. **Characteristics of typical I/O devices**

****

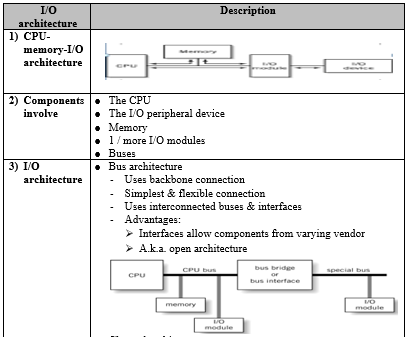
1. **I/O techniques**

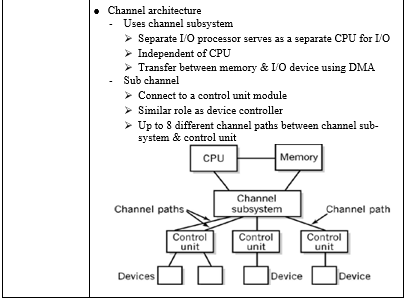
****

****

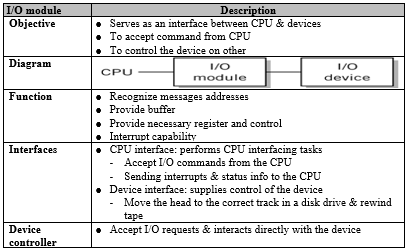
****

1. **I/O architecture**

****

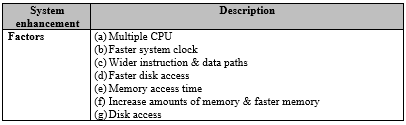
****

1. **I/O modules**

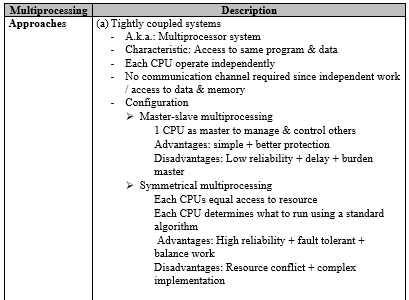


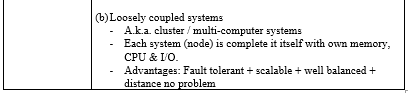
| **Chapter 12: Performance Enhancement**  **System performance enhancement & optimization**   1. **Multiprocessing** 2. **Memory enhancement** 3. **Virtual storage** 4. **Parallel processing** |
| --- |

1. **System performance enhancement & optimization**

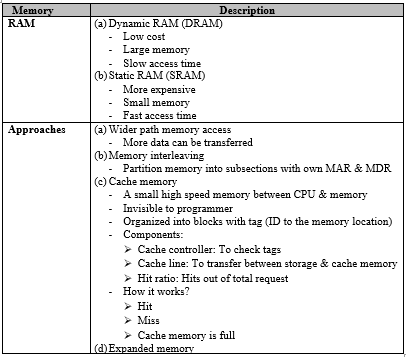
****

1. **Multiprocessing**

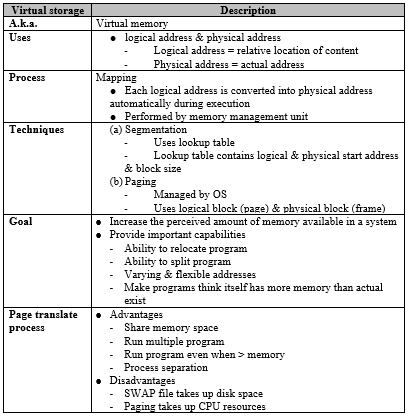
****

****

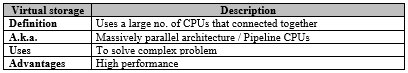
1. **Memory enhancement**

****

1. **Virtual storage**

****

1. **Parallel processing**



| Question for participation |
| --- |

| **Week** | **Question (Submission before 16:10PM)** |
| --- | --- |
| 1 | Different questions for each session (Session 1 at Mon, Session 2 at Tue) |
| 2 | Given 25D + 25D.  Complete the operation using:  1) Dec number  **25 D**   * **125 D**   **50 D**  2) Bin number  **0001 1001 B**   * **0001 1001 B**   **0011 0010 B ⇒ 25 + 24+21 = 50 D**  3) BCD number  **0010 0101 BCD**   * **0010 0101 BCD**   **0100 1010 BCD**  **-1010 BCD**   * **1**   **0101 0000 BCD ⇒ 50 D** |
| 3 | NONE |
| 4 | **Given PC = 20**  Content in memory location 20 = 530 (LOAD)  Content in memory location 21 = 231 (MUL)  Content in memory location 22 = 330 (STORE)  :  Content in memory location 30 = AAh  Content in memory location 31 = 22h    Examine the change of registers for PC, A, MAR, MDR and IR for instruction 21 -- **(5m)**  **PC → MAR ; MAR = 21**  **MDR → IR ; IR = 231**  **IR[ADDRESS] → MAR; MAR =31**  **A \* MDR → A ; A = AAH \* 22H = 1694H**  **PC + 1 → PC ; PC = 22** |
| 5 | NONE |
| 6 | Write an assembly program to output the following message.    **;---------------------------USING OUTPUT BYTE: 02H**  **.MODEL SMALL**  **.STACK 100**  **.DATA**  **.CODE**  **MAIN PROC**  **MOV AX, @ DATA**  **MOV DS,AX**    **;---OUTPUT BYTE:02H (“I”)**  **MOV AH,02H**  **MOV DL,”I”**  **INT 21H**  **;---OUTPUT BYTE:02H (03H)**  **MOV AH,02H**  **MOV DL,03H**  **INT 21H**  **;---OUTPUT BYTE:02H (“C”)**  **MOV AH,02H**  **MOV DL,”C”**  **INT 21H**  **;---OUTPUT BYTE:02H (“S”)**  **MOV AH,02H**  **MOV DL,”S”**  **INT 21H**  **;---OUTPUT BYTE:02H (“A”)**  **MOV AH,02H**  **MOV DL,”A”**  **INT 21H**  **MOV AH,4CH**  **INT 21H**  **MAIN ENDP**  **END MAIN** |
| 7 | Write a program to get user input byte (password) for 5 digits using LOOP instruction and store into a variable as defined below.  PSW DB 5 DUP (0)  **Submission latest by 5/8/2021, 8pm.**  **;---------------------------USING OUTPUT BYTE: 02H**  **.MODEL SMALL**  **.STACK 100**  **.DATA**  **PSW DB 5 DUP (0)**  **.CODE**  **MAIN PROC**  **MOV AX, @ DATA**  **MOV DS,AX**  **;---GET 5 BYTE OF USER PASSWORD**  **MOV CX,5**  **MOV SI,0**  **GETPSW:**  **MOV AH,07H**  **INT 21H**  **MOV PSW[SI],AL**  **INC SI**  **LOOP GETPSW**  **MOV AH,4CH**  **INT 21H**  **MAIN ENDP**  **END MAIN** |
| 8 | Given:  .Data  MyData DB 2,4,7,"H","A",9  Write an Assembly Language program to move the first Character into DL and output. |
| 9 | None |
| 10 | Lol miss kena battery interrupt  → Good example on the Usage of interrupt lol.  Link to videos:   1. PIO: <https://www.youtube.com/watch?v=kGnkGbAbt6M> 2. INT:<https://www.youtube.com/watch?v=qMUh9Pnb0L8> 3. DMA: Video 1: <https://www.youtube.com/watch?v=w3K1JklY6D4>   Video 2: <https://www.youtube.com/watch?v=wi-po2SesoE>  Question:  Give a real life example on how the CPU handles multiple interrupts  **Example**  **Student (CPU) attending lecturer class (current process). During the class, the student receive a phone call (interrupt A) while the delivery man is calling at door step (interrupt B) to collect parcel.**  **When multiple interrupts occur, the student (CPU) suspends the class (current process) and remember where the lesson stopped (store current states into PCB). Then consider the priority of Interrupt A & B. The process with the higher interrupt will be executed first.**  **Once the student finished handled the interrupt, the student recall / restore where the lesson suspended just now and resume the class.** |
| 11 | Describe the operation of cache memory. Providing a real life example on the operation of cache memory could be applied for a sales manager. - 8m |
| 12 | Write an assembly language program in order to calculate the total number of odd numbers and even numbers.  Given:  NUM DB 1,5,2,4,11,10,6   1. Program structure -2m   2)l Appropriate data definition -2m  3) Calculation using LOOP instruction -4m  4) Appropriate output -2m  Sample output:  There are 3 odd numbers and 4 even numbers in the NUM. |
| 13 |  |
| 14 |  |