An Introduction to 6502 Microprocessor Applications

Instructor's Solutions

Issue: MP616/A

Contents

LJ Technical Systems

Chapter 1 Using the MAC III Microcomputer

1.1a The Keypad/display is connected to the MAC III Microcomputer using:

[c] one 16-wire cable

1.1b Power is connected to the Applications Module using:

[b] one cable terminated in a 5-pin connector

1.2a Turn the potentiometer fully clockwise. Enter the hexadecimal value shown on the display.

Answer: In the range $F0_H$ to FF_H

1.3a Turn the potentiometer fully counter-clockwise. Enter the hexadecimal value shown on the display.

Answer: In the range 00_H to 10_H

1.6a With the constant speed control program running, turn the "LOAD" control fully clockwise. Wait for 5 seconds and then enter the motor speed value shown on the display.

Answer: In the range 97 to 103

1.7a **With the speed control program running, turn the potentiometer fully clockwise. Wait for 5 seconds and then enter the motor speed value shown on the display.**

Answer: In the range 90 to 150

1.9a **With the optical feedback program running, place a piece of thin card or paper between the optical sender and the receiver. Enter the light intensity value shown on the display.**

Answer: In the range 00_H to 14_H

1.9b **With the optical feedback program running, remove any thin card or paper between the optical sender and the receiver. Enter the light intensity value shown on the display.**

Answer: In the range 14_H to 60_H

Student Assessment 1

1. The MAC III Microcomputer is connected to the Applications Module using:

   d **two 9-wire cables**

2. The Applications Module power cable is connected in the:

   c **top right hand corner of the Applications Module**

3 When power is applied to the MAC III Microcomputer, the display shows:

   b **'rEAdy'**

4. The Applications Module demonstration program is executed by pressing:

   b **G** **F** **6** **0** **0** **G**

5. When the Applications Module demonstration program is first run, the display sequence is:

   a **"APPLICAtIONS", "SELECt", then "AnLOG"**

6. The keys which are used to select different sections of the Applications Module demonstration software are:

   a **+** **and** **-**

7. When the Variable Motor Speed Control section of the Applications Module demonstration software is selected, the display will show:

   d **"rPS"**

Chapter 2 Introduction to 6502 Programming

2.1a **Enter the hexadecimal contents of the MAC III memory location $FFFD_H$.**

Answer: $F0_H$

2.2a **An easily-remembered abbreviation used when writing a microprocessor instruction is called a:**

[d] Mnemonic Code

2.2b **Programming using Mnemonic Codes is called:**

[a] Assembly Language Programming

2.2c **The function of the section of MAC III memory which includes location 0800_H is:**

[d] User RAM

2.3a **Stop the program, change the data at location 1000_H to 72_H and run the program again. The pattern shown on the Applications Module Port Monitor LEDs (● = lit, ❍ = unlit) is:**

[c]

D7	D6	D5	D4	D3	D2	D1	D0
❍	●	●	●	❍	❍	●	❍

Student Assessment 2

1. The data word at MAC III memory address $E0DC_H$ is:

   b **$6C_H$**

2. The keystrokes required to change the contents of location 0407_H to $B2_H$ are:

   d **M** **0** **4** **0** **7** **M** **B** **2**

3. The form in which machine language is presented to the microprocessor is:

   a **Binary**

4. Giving instructions to the microcomputer in hexadecimal form is called:

   d **Machine Code Programming**

5. Programming using mnemonic codes is called:

   a **Assembly Language Programming**

6. The area of MAC III memory available for User Programs is:

   c **0400_H to $1FFF_H$**

7. The function of the MAC III memory area $A000_H$ to $BFFF_H$ is:

   d **User EPROM**

8. The key used to enter the memory examination mode is:

   d **M**

9. The keystrokes required to run the program which starts at location 1000_H are:

   c **G** **1** **0** **0** **0** **G**

Chapter 3 Writing Machine Code Programs

3.1a **Enter the number of bits within the 6502 Accumulator.**

Answer: 8

3.1b **A memory location initially contains the value 45_H. Enter the hexadecimal contents of this location after a 'Decrement' instruction has been executed.**

Answer: 44_H

3.5a **Enter the hexadecimal byte which must be placed in location 0404_H.**

Answer: 10_H

3.5b **In the instruction "`LDA #$65`", the operand is:**

b `#$65`

3.5c **The program in Worked Example 3.5 is to be modified so that the value 88_H is placed in location 1000_H. The memory location which must be changed is:**

b 0401_H

3.6a **The re-coded program in Worked Example 3.6 is to be modified so that the result is saved in location 1040_H. Enter the byte that must be placed in location 0607_H.**

Answer: 40_H

3.7a **Write a program, starting at memory location 0900_H, that will add the hexadecimal values 56_H and 78_H. The result should then be saved in memory location 1060_H. Run your program and then examine the contents of location 1060_H. Enter the byte that you find at this location.**

Answer: CE_H

Student Assessment 3

1. The primary 6502 Register is:

   a **the Accumulator**

2. The 6502 instruction which copies the Accumulator to a specified memory location is:

   c **Store**

3. The 6502 instruction which subtracts one from a specified register or memory location is:

   d **Decrement**

4. The function of the "Load" instruction is to:

   b **copy a specified memory location to the Accumulator**

5. The 6502 instruction which allows program execution to continue from some point other than the next location in sequence is:

   d **Jump**

6. The part of an instruction which provides any additional information necessary to complete that instruction is called the:

   c **Operand**

7. The part of an instruction that defines the function to be carried out is called the:

   d **Operator**

8. The 6502 Assembly Language mnemonics for "copy the contents of memory location 1100_H into the accumulator" are:

   b `LDA $1100`

9. If the carry flag has previously been cleared, the 6502 Assembly Language instruction `"ADC $1200"` will add:

   b **the contents of location 1200_H to the Accumulator**

Continued ...

Student Assessment 3 Continued ...

10. The machine code for the instruction "`DEC $1020`" is:

    b **CE 20 10**

11. The 6502 Assembly Language sequence which will place the hexadecimal value CC_H in location $10B0_H$ is:

    a
    ```
    LDA  #$CC
    STA  $10B0
    ```

Chapter 4 Program Debugging

4.2a **Debugging is often necessary because user programs may:**

[b] not be entirely correct when first executed

4.2b **The keypad sequence "R R 0 4 1 7" will:**

[d] insert a break point at location 0417_H

4.2c **The display**

indicates that:

[d] a break point has been reached at location $043A_H$

4.3a **The keypad sequence required to start a program single stepping is:**

[a] [G] [+]

Student Assessment 4

1. The process of finding and then correcting faults within a program is called:

   c **debugging**

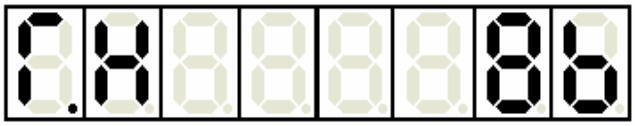
2. The key which is used at a break point to examine the contents of various registers is:

   b **R**

3. The key sequence required to set a break point at location 0428_H is:

   c **R R 0 4 2 8**

4. The display

   indicates that the contents of:

   b **the X Register are $8B_H$**

Chapter 5 Using the Merlin Text Editor

Some familiarity with the fundamental operations of a PC has been assumed. Students new to the PC may require extra work to cover basic aspects such as text editing, navigating menus and submenus etc. As with assembly language programming, students will learn more effectively if the number of commands available is restricted in the early stages. The text takes this approach, leading the student through graded exercises. No attempt is made at formal "keyboard familiarization" since experience has shown that actual use of the text editor will lead naturally to such familiarization.

The importance of regularly saving work should be stressed to students. It is important to acquire this habit now, before any problems arise.

Short text passages can be prepared and edited as desired if learning is proving slow. It is however important to emphasize that this chapter is designed to give the basic skills in text editing required to produce assembly language source code programs.

5.3a **Which of the following sequences is correct for copying text from one place to another?**

[c] Select – Copy – Paste.

5.3b **In which menu are the Find and Replace commands located?**

[b] Edit.

5.3c **The file extension used for Assembly language source code is:**

[c] .ASM

5.4a **Which of the following fonts is *not* available from the Fonts window?**

[c] Garamond.

5.6a **The link displayed at the bottom of the 'Using the source code editor' page is:**

[a] Assembling a program.

Solutions to Student Assessment 5

1. The three options that are contained in the Tools menu are:

   b **Editor, Assembler and Terminal.**

2. This  button will:

   c **Paste the text that is currently on the clipboard.**

3. The Merlin toolbar button that creates a new blank text file is:

   a 

4. The Merlin command used to place a duplicate of the selected text onto the clipboard is:

   b **Copy.**

5. The *Options* command can be found in which menu?

   a **File.**

6. The Merlin command which will locate each occurrence of a given word is:

   c **Find.**

7. The links on the help pages are colored:

   d **blue.**

8. The Merlin On Screen Help pages are split into how many sections?

   c **3**

Chapter 6 Introduction to Development Systems

6.1a **The character used to indicate binary data to the Cross Assembler is:**

[a] %

6.1b **In a source program, the start address is specified using:**

[b] an ORG directive.

6.2a **Assembly is the conversion of a source code program into:**

[c] an object code program.

6.2b **The error displayed when placing the cursor on the line is:**

[a] ERROR 2: Missing space after label '1'.

6.3a **After entering "M 0480;8" the display screen shows:**

```
0480: 3D 06 E3 78 EF D2 10 05 >.......
```

This indicates that the contents of location 0486_H are:

[b] 10_H

6.3b **The Terminal Mode key sequence [G] [0] [5] [4] [0] [Enter] will cause:**

[c] program execution from location 0540_H

6.4a **If the label "VAL1" is assigned the value $2D_H$, the 6502 Cross Assembler will interpret the label "VAL1+2" as:**

[d] $2F_H$

6.4b **The maximum number of characters for a label recognized by the 6502 Cross Assembler, is:**

[b] 8

6.5a **The correct Terminal Mode key sequence to examine the contents of location 0480_H is :**

[c] [M] [0] [4] [8] [0] [Enter]

6.5b **The Terminal Mode key sequence [C] [0] [6] [A] [0] [Enter] will allow:**

[c] the contents of location $06A0_H$ to be examined and modified if required.

Student Assessment 6

1. The ORG assembler directive is used to:

   c **define the start address for an object code program.**

2. Which of the following lines is a comment and will be ignored by the assembler?

   c **; Program 1**

3. The instruction "LDA #$01" executes which operation?

   c **Loads 01_H into the accumulator**.

4. The Terminal Mode key sequence M 0 5 0 0 Enter will allow**:**

   a **the contents of location 0500_H to be examined.**

5. Assembling a file called "PROG6.ASM" will also create a file named:

   a **PROG6.OBJ**

6. The Terminal Mode key sequence G 0 2 0 0 Enter will allow:

   c **the execution of the object program which starts at location 0200_H.**

7. The contents of memory location 0380_H can be examined and modified using the Terminal Mode key sequence (followed by Enter):

   b C 0 3 8 0

8. The execution of a program starting at address 0600_H can be traced using the key sequence (followed by Enter):

   d T 0 6 0 0

Chapter 7 Addressing Modes

7.3a **The Accumulator initially contains the value $2C_H$ and location 0580_H initially contains $4D_H$. Enter the value which would be found in the Accumulator after the instruction "`LDA $0580`" has been executed.**

Answer: $4D_H$

7.4a **The 6502 Assembly Language program section:**

```
LDA  #$42
STA  $70
```

[c] will place the value 42_H in memory location 0070_H

7.5a **In the program for Worked Example 7.5, the addressing mode used by the instruction "`ADC $0500`" is:**

[a] absolute

7.5b **Place the value $3A_H$ in location 0500_H. Run the program for Worked Example 7.5 and then examine the contents of memory location $00F0_H$. Enter the hexadecimal value which you find.**

Answer: $4C_H$

7.6a **Enter the decimal value represented by the BCD number 01110010_2.**

Answer: 72_{10}

7.6b **The BCD number which represents 42_{10} is:**

[b] 01000010_2

7.6c **The flag which is set to perform decimal arithmetic is the:**

[a] D-flag

7.7a In the program for Worked Example 7.7, the addressing mode used by the instruction "`LDA  #$12`" is:

b immediate

7.7b The program for Worked Example 7.7, is to be changed so that the result will be saved in location 0500_H. The instruction "`STA  $E0`" must be replaced by:

d `STA  $0500`

7.8 Practical Assignment

Write a program, starting at location 0400_H, which will perform **binary** addition of the contents of memory locations 0050_H, 0051_H, and 0052_H. The result should be saved in memory location 1000_H.

Note: This requires binary arithmetic.

Typical Solution:

```
                ORG  $0400      ;Defines the start address
0400    D8      CLD             ;Selects binary arithmetic mode
0401    A5      LDA  $50        ;Loads accumulator with the contents
0402    50                      ;of location 0050H
0403    18      CLC             ;Clears the carry flag
0404    65      ADC  $51        ;Adds the contents of location 0051H
0405    51                      ;to the accumulator
0406    65      ADC  $52        ;Adds the contents of location 0052H
0407    52                      ;to the accumulator
0408    8D      STA  $1000      ;Saves the contents of the accumulator
0409    00                      ;in location 1000H
040A    10
040B    60      RTS             ;Returns to MAC III monitor
```

7.8a **Place the value $2B_H$ in memory locations 0050_H, 0051_H, and 0052_H. Run your program for Practical Assignment 7.8 and enter the hexadecimal value you find in location 1000_H.**

Answer: 81_H

7.8b **Modify your program for Practical Assignment 7.8 so that it will calculate the decimal sum of the contents of locations 0050_H, 0051_H, and 0052_H. Place the BCD number representing the decimal value 19_{10}, into memory locations 0050_H, 0051_H, and 0052_H. Run your modified program, then enter the decimal value represented by the BCD number which you find in location 1000_H.**

Answer: 57_{10}

7.10a **Run the program for Worked Example 7.10. Examine the contents of location 1100_H. Enter the hexadecimal value you find at this location.**

Answer: $0D_H$

7.10b **Modify the program for Worked Example 7.10 so that it will subtract $4D_H$ from 71_H. Run your program and then examine the contents of location 1100_H. Enter the hexadecimal value you find at this location.**

Answer: 24_H

7.11 Practical Assignment

Write a program which will add the BCD number representing the value 21_{10} to the BCD number at location 0070_H and then subtract the BCD number at location 0510_H from the result. The final result must be stored as a BCD number in location 0520_H.

Note: This problem requires <u>decimal</u> arithmetic.

Typical Solution:

```
                ORG     $0400       ;Defines the start address
0400    F8      SED                 ;Selects decimal arithmetic mode
0401    A5      LDA     $70         ;Loads accumulator with the BCD
                                    ;number at memory location
0402    70                          ;0070H
0403    18      CLC
0404    69      ADC     #$21        ;Adds the BCD number representing the
0405    21                          ;decimal value 21, to the accumulator
0406    38      SEC
0407    ED      SBC     $0510       ;Subtracts the BCD number at
0408    10                          ;location 0510H from the accumulator
0409    05
040A    8D      STA     $0520       ;Saves the final result in location
040B    20                          ;0520H as a BCD number
040C    05
040D    60      RTS                 ;Returns to MAC III monitor
```

7.11a Place the BCD number representing 32_{10} in memory location 0070_H and the BCD number representing 34_{10} in location 0510_H. Run your program for Practical Assignment 7.11 and enter the decimal value represented by the BCD number at location 0520_H.

<u>Answer</u>: 19_{10}

7.11b Modify your program for Practical Assignment 7.11 so that it will perform <u>binary</u> arithmetic. Place the value $3E_H$ in memory location 0070_H and the value 42_H in location 0510_H. Run your modified program and enter the hexadecimal value you find in location 0520_H.

<u>Answer</u>: $1D_H$

Student Assessment 7

1. The 6502 addressing mode in which no operand bytes are required is called:

   a **Implied Addressing**

2. In Zero Page addressing, the number of operand bytes required is:

   b **1**

3. In Absolute addressing, the total number of bytes for an instruction is:

   d **3**

4. The 6502 Assembly Language instruction "`LDA $60`" will load the accumulator:

   c **from location 0060_H**

5. The 6502 Assembly Language instruction which causes the microprocessor to perform decimal arithmetic is:

   d **`SED`**

6. When a 6502 Subtract instruction is executed, the Carry Flag:

   b **shows any Borrow**

7. The 6502 Assembly Language instruction "`SBC $1200`" will subtract :

   c **the contents of location 1200_H from the accumulator**

8. The machine code for the 6502 Assembly Language instruction "`SBC #$3C`" is:

   c **E9 3C**

9. The 6502 Assembly Language instructions required to subtract the contents of location 0080_H from the Accumulator are:

   d
   ```
   SEC
   SBC  $80
   ```

10. The program section:
    ```
    SED
    LDA #$48
    CLC
    ADC #$22
    ```

    a **Performs decimal addition of 48_{10} and 22_{10}**

Chapter 8 Negative Binary Numbers

8.2a **The 1's complement of 01001011_2 is:**

[c] 10110100_2

8.2b **The 2's complement of 01001011_2 is:**

[d] 10110101_2

8.2c **110001_2 - 11111_2 is:**

[a] 10010_2

8.3a **The 1's complement of $3E_H$ is:**

[c] $C1_H$

8.3b **The 2's complement of 60_H is:**

[b] $A0_H$

8.3c **$3E_H$ - $0D_H$ is:**

[a] 31_H

8.4a **The 8-bit 2's complement form of -21_H is:**

[d] DF_H

8.4b **Enter the 8-bit 2's complement form of -55_H (in hexadecimal).**

Answer: AB_H

8.5a **Enter the 8-bit 2's complement form of $-B_H$ (in hexadecimal).**

Answer: $F5_H$

8.5b **$39_H - 62_H$ is:**

[b] -29_H

Student Assessment 8

1. The 1's complement of $0010\ 1110_2$ is:
   b **$1101\ 0001_2$**

2. The 2's complement of $0110\ 0111_2$ is:
   d **$1001\ 1001_2$**

3. The 2's complement of a binary number is found by:
   b **adding 1 to the 1's complement**

4. The value $-0011\ 0111_2$ can be represented using 8-bit 2's complements as:
   d **$+1100\ 1001_2$**

5. The value -37_H can be represented using 8-bit 2's complements as:
   d **$+C9_H$**

6. The result of the subtraction $0100\ 1111_2 - 0010\ 1101_2$ is:
   a **$0010\ 0010_2$**

7. The result of the subtraction $69_H - 4C_H$ is:
   b **$1D_H$**

Chapter 9 Programs with Loops

9.2a **The types of 6502 instructions which allow program execution to continue from a point other than the next location in sequence are called:**

[d] Jump or Branch instructions

9.2b **In relative addressing, the destination is specified by:**

[a] a 2's complement displacement

9.3a **After the 6502 has subtracted $4A_H$ from 67_H, the Zero (Z) and Carry (C) Flags will be:**

[c] C=1, Z=0

9.3b **After the 6502 has added 52_H to 67_H, the Zero (Z) and Carry (C) Flags will be:**

[a] C=0, Z=0

9.3c **After the 6502 has added 75_H to $8E_H$, the Zero (Z) and Carry (C) Flags will be:**

[c] C=1, Z=0

9.3d **After the 6502 has subtracted 72_H from 72_H, the Zero (Z) and Carry (C) Flags will be:**

[b] C=0, Z=1

9.4a **The 6502 assembly language instruction** "BNE WAIT" **will branch to the location identified by the label** 'WAIT' **if:**

[d] the Zero Flag is clear (Z=0)

9.5a **Load the above program into the MAC III and then place the following values into MAC III memory:**

Location	**Contents**
0500_H	12_H
0501_H	34_H

Run the program and examine the contents of location 0502_H. Enter the hexadecimal value which you find.

Answer: 01_H

9.5b **With the above program still loaded into MAC III memory, modify the following locations as indicated below:**

Location	**Contents**
0500_H	AB_H
0501_H	CD_H

Run the program again and examine the contents of location 0502_H. Enter the hexadecimal value which you now find.

Answer: 80_H

9.5c **Load this modified program into the MAC III and place the following values in MAC III memory:**

Location	**Contents**
0500_H	56_H
0501_H	78_H

Run the program and examine the contents of location 0502_H. Enter the hexadecimal value which you find.

Answer: 01_H

9.6a **Use the program for Worked Example 9.6 to calculate $67_H + 89_H$. Enter the result you find.**

Answer: $00F0_H$

9.6b Use the program for Worked Example 9.6 to calculate $CD_H + EF_H$. Enter the result you find.

Answer: $01BC_H$

9.7 Practical Assignment

Write a program which will examine the contents of location 0500_H. If the contents are 00_H, location $00FF_H$ should be loaded with 80_H. If the contents are non-zero, location $00FF_H$ should be loaded with $7F_H$.

Typical Solution:

```
                        ORG     $0400       ;Defines the start address
0400    AD              LDA     $0500       ;Loads the accumulator from
0401    00                                  ;location 0500H
0402    05
0403    F0              BEQ     ZSET        ;Is the Zero Flag Set ?
0404    05
0405    A9              LDA     #$7F        ;Z=0 so load marker value 7FH
0406    7F                                  ;into the accumulator
0407    85              STA     $FF         ;Save marker value in location
0408    FF                                  ;00FFH
0419    60              RTS                 ;Returns to MAC III system
040A    A9      ZSET:   LDA     #$80        ;Z=1 so load marker value 80H
040B    80                                  ;into the accumulator
040C    85              STA     $FF         ;Save marker value in
040D    FF                                  ;location 00FFH
040E    60              RTS                 ;Returns to MAC III system
```

9.7a **Load your program for Practical Assignment 9.7 into the MAC III. Place the value 56_H in memory location 0500_H. Run your program and then examine the contents of location $00FF_H$. Enter the hexadecimal value which you find.**

Answer: $7F_H$

9.7b **With your program for Practical Assignment 9.7 still loaded in MAC III memory, now place the value 00_H in memory location 0500_H. Run your program again and examine the contents of location $00FF_H$. Enter the hexadecimal value which you find.**

Answer: 80_H

9.9a **Load the above program into the MAC III. Place the value 28_H in memory location 0500_H. Run your program and then examine the contents of location 0500_H. Enter the hexadecimal value which you find.**

Answer: $2F_H$

9.10 Practical Assignment

Location 0500_H contains a value between 00_H and 12_H which is to be multiplied by the value $0E_H$. Write a program which will perform this multiplication, saving the result in location $00F0_H$.

HINT: A simple means of achieving multiplication is to add a value to itself a given number of times.

Typical Solution:

```
                        ORG   $0400   ;Defines the start address
0400   A9               LDA   #$0D    ;Loads the accumulator with the
0401   0D                             ;count value (0DH)
0402   85               STA   $FF     ;Saves the count value in location
0403   FF                             ;00FFH
0404   D8               CLD           ;Selects binary arithmetic mode
0405   18               CLC
0406   AD               LDA   $0500   ;Loads the contents of location
0407   00                             ;0500H into the accumulator
0408   05
0409   6D      ADDVAL:  ADC   $0500   ;Adds the contents of location
040A   00                             ;0500H to itself
040B   05
040C   C6               DEC   $FF     ;Reduces the loop count by 01H
040D   FF
040E   D0               BNE   ADDVAL  ;Branch back to repeat the addition
040F   F9                             ;until the loop count is zero
0410   85               STA   $F0     ;Saves the result in location 00F0H
0411   F0
0412   60               RTS           ;Returns to MAC III system
```

A common student error is to make the initial count value $0E_H$ rather than $0D_H$. The program structure shown performs the addition **before** the loop counter is decremented. Thus the initial value must be 01_H **less** than the required count.

Class discussion could focus on alternative programming strategies for this type of problem.

9.10a Use your program for Practical Assignment 9.10 to calculate $0A_H$ x $0E_H$. Enter the result you find.

Answer: $8C_H$

9.10b Modify your program for Practical Assignment 9.10 to calculate 09_H x 08_H. Enter the result you find.

Answer: 48_H

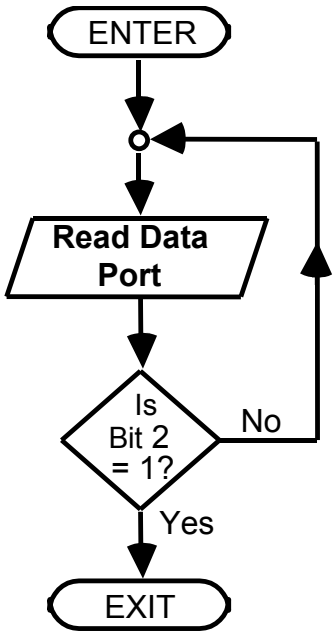
Student Assessment 9

1. The type of structure used to repeat a section of program several times is called:

   c **a Loop**

2. The program section described by the flowchart shown below will:

   b **repeat until a condition becomes true**



3. The type of JUMP or BRANCH which is always taken is called a

   d **Unconditional Jump or Branch**

4. The types of JUMPs or BRANCHes which allow the microprocessor to make decisions are called:

   a **Conditional Jumps or Branches**

5. The type of addressing where the destination is expressed in terms of the number of bytes forward or backward from the present location is called:

   d **Relative**

6. The largest positive 8-bit offset for relative addressing is:

   c **127_{10}**

Student Assessment 9 Continued ...

7. The assembly language instruction at location 0418_H is "`BCC INCPRT`". If the location identified by the label "INCPRT" is $041E_H$, the 2's complement displacement for the branch instruction will be:

   b **04_H**

8. The Carry Flag is set when the result of the last arithmetic operation is:

   d **greater than 8 bits**

9. The Flag which is set when the result of the last arithmetic operation is zero is the:

   d **Zero Flag**

10. The program section which will repeatedly (and indefinitely) add 02_H to the Accumulator is:

    a
    ```
    HERE:   ADC  #$02
            JMP  HERE
    ```

11. The program section below

    ```
    NEXT:   ADC  $2000
            BCS  DONE
            JMP  NEXT
    ```

    will add the contents of location 2000_H to the Accumulator:

    b **until the result is greater than 8 bits**

Chapter 10 Further Programs with Loops

10.1a If the Accumulator contains the value 49_H and then the instruction "`CMP #$49`" is executed, the status of the Carry (C) and Zero (Z) Flags will be:

[d] C=1, Z=1

10.1b The Accumulator initially contains the value $3A_H$. The instruction "`CMP #$25`" is then executed. Enter the new contents of the Accumulator (in hexadecimal).

Answer: $3A_H$

10.1c The Accumulator initially contains the value 77_H. A COMPARE instruction is executed. This sets the Carry (C) Flag and clears the Zero (Z) Flag. The value which was compared with the Accumulator was:

[a] less than 77_H

10.2a Load the program for Worked Example 10.2 into MAC III memory. Place the value 46_H in location 0500_H and the value 71_H in location 0501_H. Run the program and examine the contents of location 0502_H. Enter the hexadecimal value which you find.

Answer: 71_H

10.3a Load the program for Worked Example 10.3 into MAC III memory. Place the value 52_H in location 0500_H. Run the program and then examine the contents of location 0500_H. Enter the hexadecimal value which you find.

Answer: AA_H

10.4 Practical Assignment

Write a program which will examine the contents of location 0050_H. If this location contains 99_H, then location 0500_H should be loaded with 81_H. Otherwise location 0500_H should be loaded with $7E_H$.

Typical Solution

```
                        ORG     $0400    ;Defines the start address
0400   A5               LDA     $50      ;Loads the accumulator
0401   50                                ;from location 0050H
0402   C9               CMP     #$99     ;Compares the accumulator
0403   99                                ;with 99H
0404   F0               BEQ     TRUE     ;If the result is zero,
0405   06                                ;branch to location 040CH
0406   A9   FALSE:     LDA     #$7E
0407   7E
0408   8D               STA     $0500    ;Z=0 so save marker 7EH
0409   00                                ;in location 0500H
040A   05
040B   60               RTS              ;Returns to MAC III system
040C   A9   TRUE:      LDA     #$81
040D   81
040E   8D               STA     $0500    ;Z=1 so save marker 81H
040F   00                                ;in location 0500H
0410   05
0411   60               RTS              ;Returns to MAC III system
```

10.4a Load your program for Practical Assignment 10.4 into MAC III memory. Place the value $3B_H$ in location 0050_H. Run the program and then examine the contents of location 0500_H. Enter the hexadecimal value which you find.

Answer: $7E_H$

10.4b The number of times that your program for Practical Assignment 10.4 uses a "CMP" instruction is:

[a] once

10.5 Practical Assignment

Write a program which will inspect the contents of location 0580_H. Location $00FF_H$ should then be loaded with a marker value thus:

If the contents of location 0580_H are:

less than 37_H: load location $00FF_H$ with 80_H
equal to 37_H: load location $00FF_H$ with AA_H
greater than 37_H: load location $00FF_H$ with 01_H

Typical Solution

```
                    ORG     $0400    ;Defines the start address
0400   AD           LDA     $0580    ;Read contents of
0401   80                            ;location 0580H into
0402   05                            ;the accumulator
0403   C9           CMP     #$37     ;Compare accumulator
0404   37                            ;with the value 37H
0405   F0           BEQ     SAME     ;If zero flag is set,
0406   07                            ;branch to location 040EH
0407   90           BCC     LESS     ;If carry flag is clear,
0408   0A                            ;branch to location 0413H
0409   A9  GREATER: LDA     #$01     ;Contents of location
040A   01                            ;0580H are GREATER THAN
                                     ;37H so load marker 01H
040B   85           STA     $FF      ;Saves marker 01H in
040C   FF                            ;location 00FFH
040D   60           RTS              ;Returns to MAC III system
040E   A9  SAME:     LDA     #$AA     ;Contents of location
040F   AA                            ;0580H EQUAL 37H so load
                                     ;marker AAH
0410   85           STA     $FF      ;Saves marker AAH in
0411   FF                            ;location 00FFH
0412   60           RTS              ;Returns to MAC III system
0413   A9  LESS:    LDA     #$80     ;Contents of location
0414   80                            ;0580H are LESS THAN 37H
                                     ;so load marker 80H
0415   85           STA     $FF      ;Saves marker 80H in
0416   FF                            ;location 00FFH
0417   60           RTS              ;Returns to MAC III system
```

10.5a Load your program for Practical Assignment 10.5 into MAC III memory. Place the value 93_H in location 0580_H. Run the program and then examine the contents of location $00FF_H$. Enter the hexadecimal value which you find.

Answer: 01_H

10.5b Enter the number of times that your program for Practical Assignment 10.5 uses a "`CMP`" instruction.

Answer: range 1 to 2

10.6 Practical Assignment

Write a program which will inspect the contents of locations 0050_H, 0051_H and 0052_H. The largest of these should then be saved in location 0500_H.

Typical Solution

```
                        ORG     $0400    ;Defines the start address
0400   A5               LDA     $50      ;Read contents of
0401   50                                ;location 0050H into
                                          ;the accumulator
0402   C5               CMP     $51      ;Compares accumulator
0403   51                                ;with the contents of
                                          ;location 0051H
0404   90               BCC     MEM1     ;If carry flag is clear,
0405   08                                ;branch to the label MEM1
0406   C5  CHECK:       CMP     $52      ;Compare with contents of
0407   52                                ;location 0052H
0408   90               BCC     MEM2     ;If carry flag is clear,
0409   09                                ;branch to label MEM2
040A   8D  MEM0:        STA     $0500    ;Contents of location
040B   00                                ;with greatest value are
040C   05                                ;saved in location 0500H
040D   60               RTS              ;Returns to MAC III system
040E   A5  MEM1:        LDA $51          ;Contents of location
040F   51                                ;0051H are greater than
                                          ;contents of 0050H, so
                                          ;load accumulator from 0051H
0410   4C               JMP     CHECK    ;Jump back to compare
0411   06                                ;contents of location
0412   04                                ;0051H with those of
                                          ;location 0052H
0413   A5  MEM2:        LDA     $52      ;Contents of location
0414   52                                ;0052H are the greatest
                                          ;so load accumulator from
                                          ;location 0052H
0415   4C               JMP     MEM0     ;Jump back to label MEM0
0416   0A                                ;save contents of location
0417   04                                ;0052H in location 0500H
```

10.6a Load your program for Practical Assignment 10.6 into the MAC III. Place the values shown below in the memory locations indicated:

Location	Contents
0050_H	$2D_H$
0051_H	71_H
0052_H	$5E_H$

Run your program and then examine the contents of location 0500_H. Enter the hexadecimal value which you find.

Answer: 71_H

10.6b With your program for Practical Assignment 10.6 still loaded in the MAC III, change the values stored in the memory locations below thus:

Location	Contents
0050_H	52_H
0051_H	$4A_H$
0052_H	67_H

Run your program and then examine the contents of location 0500_H. Enter the hexadecimal value which you find.

Answer: 67_H

Student Assessment 10

1. The 6502 Assembly Language instruction which will subtract the contents of a memory location from the Accumulator and set or clear flags accordingly, without changing the contents of the memory location or the Accumulator is:

   a **Compare**

2. The 6502 Assembly Language instruction which can be used to check if the contents of the Accumulator are equal to 56_H is:

   d **`CMP #$56`**

3. Following a COMPARE instruction, both the Zero and Carry Flags are **clear** (i.e. = 0). This indicates that:

   c **the accumulator is smaller than the operand**

4. If the accumulator is **greater** than the operand for a COMPARE instruction, the Zero and Carry Flags will be:

   b **Z = 0 C = 1**

5. Consider the program section:

```
        CMP  $1800
        BCC  DEST1
        LDA  #$11
        STA  $60
        RTS
DEST1:  LDA  #$88
        STA  $60
        RTS
```

   The action of this program section will be to place the value:

   c **88_H in location 0060_H if the Carry Flag is clear**

6. For the program in Question 5 above; if the value in location 1800_H was equal to the contents of the Accumulator, the value placed in location 0060_H would be:

   b **11_H**

Chapter 11 Indexed Addressing

11.1a The 6502 instruction which will copy the contents of memory location 0527_H into the X Register is:

[b] `LDX  $0527`

11.1b The 6502 instruction "`CPY $7A`" will:

[d] compare the contents of location $007A_H$ with the Y Register

11.4a The 6502 program section:

```
LDX  #$2E
LDA  #$45
STA  $90,X
```

will place the value:

[b] 45_H in location $00BE_H$

11.4b The 6502 instruction which will copy the contents of the memory location in a data table starting at location 0200_H and pointed to by the Y Register into the accumulator is:

[a] `LDA  $0200,Y`

11.5a In the program above, the instruction which tests to see whether the next location is to be filled with 88_H is:

[c] `BNE  LOOP`

11.5b Load the program above into the MAC III and then execute from location 0400_H. Examine the contents of location 0500_H. Enter the hexadecimal value which you find at this location.

Answer: 88_H

11.6 Practical Assignment

Write a program which will fill locations 0500_H to 0580_H with the value AA_H.

Typical Solution

```
                    ORG     $0400     ;Defines the start address
0400   A2           LDX     #$81      ;Set initial count to 81H
0401   81
0402   A9           LDA     #$AA      ;Loads accumulator with AAH
0403   AA
0404   CA    LOOP:  DEX               ;Decrements the count
0405   9D           STA     $0500,X   ;Save accumulator in the
0406   00                             ;'Xth' location
0407   05
0408   D0           BNE     LOOP      ;If X-Register is not yet
0409   FA                             ;zero, branch to
                                      ;location 0404H
040A   60           RTS               ;Returns to MAC III system
```

11.6a Place the value 00_H in location 0580_H. Load your program for Practical Assignment 11.6 into the MAC III and execute. Examine the contents of location 0580_H. Enter the hexadecimal value which you find at this location.

Answer: AA_H

11.6b Place the value 00_H in location 0581_H. Check that your program for Practical Assignment 11.6 is still loaded in the MAC III. Run the program and then examine the contents of location 0581_H. Enter the hexadecimal value which you find at this location.

Answer: 00_H

11.7 Practical Assignment

Write a program which will copy the block of data 0500_H - 0520_H to locations 0580_H - $05A0_H$.

Typical Solution

```
                        ORG     $0400     ;Defines the start address
0400    A2              LDX     #$20      ;Sets the count to 20H
0401    20
0402    BD    NEXT:     LDA     $0500,X   ;Loads the accumulator
0403    00                                ;from the 'Xth' location
0404    05
0405    9D              STA     $0580,X   ;Saves the accumulator in
0406    80                                ;the 'Xth' location
0407    05
0408    CA              DEX               ;Decrements the X-Register
0409    10              BPL     NEXT      ;If the X-register is positive,
040A    F7                                ;branch back to location 0402H
040B    60              RTS               ;Returns to MAC III system
```

Note: This solution tests the Negative flag for the first time, using the BPL instruction. Note that the Negative flag will be set if the result of the decrement operation is a **negative 2's complement value** (indicated by bit 7 of the X-Register being set), and cleared if the result is positive or zero (bit 7 cleared). Encourage students to experiment with the BPL and BMI instructions.

11.7a Place the value 68_H in location 0520_H. Load your program for Practical Assignment 11.7 into the MAC III and execute. Examine the contents of location $05A0_H$. Enter the hexadecimal value which you find at this location.

Answer: 68_H

11.7b Place the value 22_H in location $05A1_H$. Check that your program for Practical Assignment 11.7 is still loaded in the MAC III. Run the program and then examine the contents of location $05A1_H$. Enter the hexadecimal value which you find at this location.

Answer: 22_H

11.8 Practical Assignment

Write a program which will examine the contents of each location from 0040_H to 0060_H and save the largest value found in location $00FF_H$.

Typical Solution

```
                        ORG     $0400     ;Defines the start address
0400   A2               LDX     #$20      ;Sets initial count
0401   20
0402   A9               LDA     #$00
0403   00
0404   85               STA     $FF       ;Clears temporary store
0405   FF                                 ;(to hold highest current value)
0406   B5   NEXT:       LDA     $40,X     ;Reads Xth location
0407   40
0408   C5               CMP     $FF       ;Compares with temporary store
0409   FF                                 ;value
040A   90               BCC     TEMP      ;If temp store >
040B   02                                 ;accumulator, read next value
040C   85               STA     $FF       ;Accumulator > temporary
040D   FF                                 ;store so save accumulator in
                                           ;temp store
040E   CA   TEMP:       DEX               ;Decrement X-register
040F   10               BPL     NEXT      ;Branch back to 0406H if
0410   F5                                 ;not all done
0411   60               RTS               ;Returns to MAC III system
```

11.8a Place the value 00_H in every location from 0040_H to 0060_H. Now place the following values in the locations shown:

Location	Contents
0040_H	45_H
0050_H	67_H
0060_H	32_H

Load your program for Practical Assignment 11.8 into the MAC III and execute. Examine the contents of location $00FF_H$. Enter the hexadecimal value which you find at this location.

Answer: 67_H

Student Assessment 11

1. The 6502 instruction which will save the contents of the X Register in location 0500_H is:

   b **`STX  $0500`**

2. The Y Register initially holds the value $4F_H$. After the instruction " `DEY` " has been executed, the contents of the Y Register will be:

   c **$4E_H$**

3. The instruction which copies the contents of the Accumulator into the X Register is:

   a **`TAX`**

4. The sequence of 6502 Assembly Language instructions required to transfer the contents of the X Register to the Y Register is:

   a
   ```
   TXA
   TAY
   ```

5. For the program section:
   ```
   LDX  #$16
   LDA  $0515,X
   ```
   The second instruction will load the accumulator from location:

   c **$052B_H$**

6. The mode of addressing used by the 6502 instruction " `STA  $0680,Y` " is:

   b **Absolute Indexed Y**

7. The 6502 instruction " `LDA  $80,X` " will load:

   d **the Accumulator from location (0080_H + X)**

8. The 6502 program section:
   ```
   LDX  #$42
   STA  $0800,X
   ```
   will:

   d **Save the accumulator in location 0842_H**

Continued...

Student Assessment 11 Continued ...

9. After the 6502 instruction sequence below has been executed,

```
LDY  #$4D
STA  $0780,Y
DEY
STA  $0780,Y
DEY
STA  $0780,Y
```

the contents of the Y Register will be:

a **$4B_H$**

Chapter 12 Logical and Test Instructions

12.1a The Accumulator initially contains the value $B7_H$. Enter the value found in the Accumulator after it has been ANDed with $C6_H$.

Answer: 86_H

12.2a The 6502 instruction which can be used to test for several bits of a memory location set at the same time is:

[a] `AND`

12.2b Load the program for Worked Example 12.2 into the MAC III. Place the value 16_H in location 0500_H. Run the program and then examine the contents of location $00F0_H$. Enter the hexadecimal value which you find.

Answer: 03_H

12.2c The program for Worked Example 12.2 is to be modified to test for any of bits 2, 3 or 4 set in memory location 0500_H. Enter the required hexadecimal mask value.

Answer: $1C_H$

12.4a The Accumulator initially contains the value $A6_H$. Enter the value found in the Accumulator after the instruction "`BIT $0500`" has been executed.

Answer: $A6_H$

12.4b The program for Worked Example 12.2 is to be modified to test for <u>any</u> of bits 1, 2 or 3 set in memory location 0500_H. The instruction which must be changed is:

[a] `LDA  #$E0`

12.5a A register contains the byte $9C_H$. Enter the hexadecimal contents of this register after it has been shifted left 3 times.

<u>Answer</u>: $E0_H$

12.5b The 6502 instruction which will shift the contents of location 0524_H once to the right is:

[d] `LSR  $0524`

12.5c A register contains the byte 64_H. If the Carry Flag is clear, enter the hexadecimal contents of this register after it has been rotated right 4 times.

<u>Answer</u>: 86_H

12.5d The 6502 instruction which will rotate the contents of the Accumulator once to the left is:

[c] `ROL  A`

12.6a Load the program for Worked Example 12.6 into the MAC III. Use this program to calculate $6A_H$ x 92_H. Enter the hexadecimal result which you obtain.

<u>Answer</u>: $3C74_H$

Student Assessment 12

1. When the binary number 1001 1001_2 is logically ANDed with the mask 1111 0000_2, the result is:

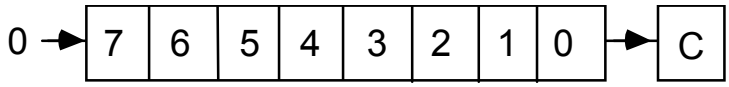
   c **1001 0000_2**

2. The mask required to test bits 6, 3 and 0 of the Accumulator is:

   b **49_H**

3. The Shift Right instruction (LSR) can be represented as:

   a

4. The 6502 Assembly Language instruction which allows the Accumulator to be ANDed with a memory location but which does not change the contents of either is:

   b **`BIT`**

5. Shifting a register one place to the left has the effect of:

   c **multiplication by 2**

6. The Accumulator initially contains 34_H. After the instruction " `AND  #$EB` " has been executed, the contents of the Accumulator will be:

   b **20_H**

7. Initially, memory location 0600_H contains the value 70_H and the Accumulator contains $2D_H$. After the instruction " `BIT  $0600` " has been executed, the contents of the Accumulator will be:

   d **$2D_H$**

Chapter 13 Input and Output Programming

13.1a The instruction that is used to output data from Port B of the MAC III 6522 VIA is:

c `STA PBDR`

13.1b All bits of Port A are to be programmed as inputs. Enter the hexadecimal value which must be written to Port A Data Direction Register.

Answer: 00_H

13.2a The program for Worked Example 13.2 is to be modified so that the byte which is output at Port A is 28_H. The instruction which must be changed is:

c `LDA #$7E`

13.3 Practical Assignment

Write a program which will add the contents of memory locations 0040_H and 0041_H. The result should be output from Port A.

Typical Solution

```
             PADR:   EQU     $9001
             PADDR:  EQU     $9003

                     ORG     $0400     ;Defines the start address
0400   A9            LDA     #$FF      ;Loads accumulator
0401   FF                              ;with 1111 1111 binary
0402   8D            STA     PADDR     ;Makes Port A all
0403   03                              ;output bits
0404   90
0405   A5            LDA     $40       ;Loads the
0406   40                              ;accumulator from location 0040H
0407   D8            CLD               ;Selects binary arithmetic mode
0408   18            CLC
0409   65            ADC     $41       ;Adds contents of location
040A   41                              ;0041H to the accumulator
040B   8D            STA     PADR      ;Outputs accumulator
040C   01                              ;contents at Port A
040D   90
040E   60            RTS               ;Returns to MAC III system
```

13.3a Set the contents of memory location 0040_H to $1B_H$ and the contents of location 0041_H to $2F_H$. Run your program for Practical Assignment 13.3 and enter the hexadecimal value output at Port A.

Answer: $4A_H$

13.4a In the program for Worked Example 13.4, if the instruction "`BNE  B4SET`" is changed to "`BEQ B4SET`", the program would:

[d] output 07_H when the input is a '0' and 70_H when the input is a '1'

13.6a The program for Worked Example 13.6 is to be modified to produce a delay of 800µs. Enter the hexadecimal value which the first instruction must load into the X Register.

Answer: $A0_H$

13.7a The program for Worked Example 13.7 is to be modified to produce a delay of 15.3ms. Enter the hexadecimal value which the first instruction must load into the X Register.

Answer: $0C_H$

13.8a Enter the delay in microseconds (µs) produced by a single "NOP" instruction.

Answer: 2

13.9 Practical Assignment

Write a program which will output a binary up-count, increasing by one about every 0.5 seconds at Port A. The Applications Module motor disc detector is to be used as an input. If the input is a "0", the binary count may continue. If the input is "1", the binary count should be suspended.

Typical Solution

```
                PADDR:    EQU     $9003
                PADR:     EQU     $9001
                PBDDR:    EQU     $9002
                PBDR:     EQU     $9000

                          ORG     $0400     ;Defines the start address
0400   A9                 LDA     #$FF
0401   FF
0402   8D                 STA     PADDR     ;Makes Port A all output bits
0403   03
0404   90
0405   A9                 LDA     #$00      ;Loads accumulator with 0000 0000
0406   00                                   ;binary
0407   8D                 STA     PBDDR     ;Makes Port B all input bits
0408   02
0409   90
040A   8D                 STA     PADR      ;Clears Port A initially
040B   01
040C   90
040D   A9     MASK:       LDA     #$10      ;Loads accumulator with
040E   10                                   ;mask for bit 4
040F   2C     B4TST:      BIT     PBDR      ;Tests bit 4 of Port B
0410   00
0411   90
0412   D0                 BNE     B4TST     ;Is bit 4 set ? If not,
0413   FB                                   ;test again
0414   EE                 INC     PADR      ;Increase output count
0415   01
0416   90
0417   A2                 LDX     #$FF
0418   FF
0419   A0                 LDY     #$FF      ;Initial values for delay
041A   FF
041B   CA     DELAY:      DEX
041C   EA                 NOP
041D   DO                 BNE     DELAY     ;Least significant delay
041E   FC                                   ;loop - 1.785ms
041F   88                 DEY
0420   D0                 BNE     DELAY     ;Most significant delay
0421   F9                                   ;loop - 0.455s
0422   4C                 JMP     MASK      ;Loop back to test bit 4
0423   0D                                   ;again
0424   04
```

13.9a Load your program for Practical Assignment 13.9 into the MAC III. Set the input to a logic "1" and run the program. Now set the input to logic "0" for 20 seconds and return it to logic "1". Enter the hexadecimal byte shown on the Port A monitor LED's.

Answer: range 24_H to 30_H

Student Assessment 13

1. Data enters and leaves the microcomputer by means of:

   [b] a Data Port

2. The 6502 Assembly Language instruction which will read the data input at Port B is:

   [b] LOAD

3. The 6502 Assembly Language instruction "`STA  $9001`" will:

   [c] output the contents of the Accumulator at Port A

4. The bits of a 6522-VIA Port which are to be inputs have a logic 0 written into the:

   [c] data direction register

5. The correct assembly language sequence required to output the value $D5_H$ from Port A on the MAC III is:

   [a]
   ```
   LDA  #$FF
   STA  PADDR
   LDA  #$D5
   STA  PADR
   ```

6. The 6502 Assembly Language instruction sequence:

   ```
   LDA  #$0F
   STA  $9002
   ```

   will configure Port B:

   [d] bits 0, 1, 2 and 3 as outputs and bits 4, 5, 6 and 7 as inputs

7. The time taken by the MAC III to execute a " `DEX` " instruction is:

   [b] 2 µs

8. The delay produced in the MAC III by the 6502 assembly Language program:

   ```
   0400 A2       LDX  #$20
   0401 20
   0402 CA LOOP: DEX
   0403 D0       BNE  LOOP
   0404 FD
   0405 60       RTS
   ```

   will be:

   [a] 160 µs

Chapter 14 Programming the Applications Module

14.2a In Worked Example 14.2, the effect of reducing the delay between each change of the output state to 100µs will change the frequency of the sound emitted to:

[a] 5kHz

14.3a The Ultrasonic Transmitter is switched on by applying a:

[b] logic "1" at PB6

14.4a In Worked Example 14.4, the effect of changing the second "`LDA #$40`" instruction to "`LDA #$00`" would be to:

[a] disable the Ultrasonic Transmitter

14.5 Practical Assignment

Write a program which uses the Ultrasonic Units within the Applications Module to act as a proximity detector. When an object is placed directly above the Ultrasonic Unit, the Piezo Sounder should be activated.

Typical Solution

```
                PBDDR:   EQU     $9002
                PBDR:    EQU     $9000
                         ORG     $0400    ;Defines the start address
0400   A9               LDA     #$60     ;Loads accumulator with
0401   60                                ;0110 0000 binary
0402   8D               STA     PBDDR    ;Sets Port B: PB7=I/P,
0403   02                                ;PB6=O/P, PB5= O/P
0404   90                                ;others don't care
0405   A9        UTXON: LDA     #$40     ;Loads accumulator with
0406   40                                ;01000000 binary
0407   8D               STA     PBDR     ;Outputs a "1" on PB6 to
0408   00                                ;switch on Ultrasonic
0409   90                                ;Transmitter
040A   A9               LDA     #$80     ;Loads accumulator with
040B   80                                ;10000000 binary
040C   2C        URXTST: BIT     PBDR     ;Test PB7
040D   00
040E   90
040F   F0               BEQ     ALARM    ;If PB7=0, branch to sound
0410   03                                ;alarm section
0411   4C               JMP     URXTST   ;Jump back to test for
0412   0C                                ;Ultrasound received
0413   04
0414   A0        ALARM: LDY     #$80     ;Count for alarm burst
0415   80
0416   A9        BURST: LDA     #$20     ;Loads accumulator with
0417   20                                ;0010 0000 binary
0418   8D               STA     PBDR     ;Outputs a "1" on PB5
0419   00                                ;(Piezo Sounder)
041A   90
041B   A2               LDX     #$64
041C   64
041D   CA        DELAY1: DEX
041E   DO               BNE     DELAY1   ;Wait 500µs
041F   FD
0420   A9               LDA     #$00
0421   00
```

Continued ...

Continued...

```
0422   8D            STA     PBDR    ;Outputs a "0" on PB5
0423   00                            ;(Piezo Sounder)
0424   90
0425   A2            LDX     #$64
0426   64
0427   CA   DELAY2:  DEX
0428   DO            BNE     DELAY2  ;Wait 500µs again
0429   FD
042A   88            DEY             ;Decrement burst count
042B   DO            BNE     BURST   ;If alarm burst count is not zero
042C   E9                            ;branch back to continue burst
042D   4C            JMP     UTXON   ;Repeat from switching
042E   05                            ;on the Ultrasonic
042F   04                            ;Transmitter
```

14.5a Run your program for Practical Assignment 14.5. The status of the "PZO" and "URX" LEDs when the alarm is sounding are:

[d] PZO LED on and URX LED on

14.5b In your program for Practical Assignment 14.5, the data bits which were written to bit positions 7, 6 and 5 respectively of Data Direction Register B were:

[c] 0, 1, 1

14.6a If an input code of 64_H is applied to the Applications Module Digital to Analog Converter (DAC), enter the output voltage (in volts).

Answer: 1 V

14.7a Run the above program again and note the hexadecimal count at the monitor LEDs when the motor just starts to rotate. Enter this hexadecimal byte.

Answer: range 10_H to 80_H

14.8a If an input voltage of 1.5V is applied to the Applications Module Analog to Digital Converter (ADC), enter the output hexadecimal byte.

Answer: 96_H

14.9a Part of the program in Worked Example 14.9 generates a short negative going pulse on PB1. The purpose of this section of the program is to:

a initiate Analog to Digital Conversion

14.11 Practical Assignment

Write a program which will sound the Piezo Sounder whenever the optical link between Optical Sender and Receiver is broken.

Note: *It can be assumed that if the optical link is unbroken, the ADC output will be greater than 15_H. When the link is broken, the ADC output will fall below 15_H.*

Typical Solution

```
                PADDR:   EQU     $9003
                PADR:    EQU     $9001
                PBDDR:   EQU     $9002
                PBDR:    EQU     $9000

                         ORG     $0400    ;Defines the start address
0400   A9       START:   LDA     #$2B
0401   2B
0402   8D               STA     PBDDR    ;Configures PB0, PB1, PB3, PB5 as
0403   02                                ;outputs, the rest as inputs
0404   90
0405   A9       LOOP:    LDA     #$0B
0406   0B
0407   8D               STA     PBDR     ;Disable DAC, take ADC RD and WR
0408   00                                ;high, PZO low
0409   90
040A   A9               LDA     #$FF
040B   FF
040C   8D               STA     PADDR    ;Configures Port A as all outputs
040D   03
040E   90
040F   A9               LDA     #$FF
0410   FF
0411   8D               STA     PADR     ;Output value for maximum light
0412   01                                ;intensity
0413   90
```

Continued ...

Continued...

```
0414   A9            LDA     #$0A
0415   0A
0416   8D            STA     PBDR     ;Enable DAC
0417   00
0418   90
0419   A9            LDA     #$0B
041A   0B
041B   8D            STA     PBDR     ;Latch output value inside DAC
041C   00
041D   90
041E   A9            LDA     #$00
041F   00
0420   8D            STA     PADDR    ;Configures Port A as all inputs
0421   03
0422   90
0423   A9            LDA     #$09
0424   09
0425   8D            STA     PBDR     ;Take ADC WR line low
0426   00
0427   90
0428   A9            LDA     #$0B
0429   0B
042A   8D            STA     PBDR     ;Return ADC WR line high, to
042B   00                              ;initiate A-D conversion
042C   90
042D   A9            LDA     #$04     ;Mask bit 2 (BSY)
042E   04
042F   2C   TSTB2:   BIT     PBDR     ;Test bit 2
0430   00
0431   90
0432   F0            BEQ     TSTB2    ;Waits for BSY=1 - conversion
0433   FB                              ;completed
0434   A9            LDA     #$03
0435   03
0436   8D            STA     PBDR     ;Sets RD=0 to enable ADC Output
0437   00
0438   90
0439   AD            LDA     PADR     ;Reads input value
043A   01
043B   90
043C   C9            CMP     #$15     ;Is it less than 15H?
043D   15
043E   90            BCC     ALARM    ;If less than 15H, sound alarm
043F   03
0440   4C            JMP     LOOP     ;Loop back to beginning
0441   05
0442   04
```

Continued...

Continued...

```
0443   A2   ALARM:   LDX     #$FF    ;Count value for delay 1
0444   FF
0445   CA   DELAY1:  DEX
0446   D0             BNE     DELAY1  ;Wait for 1.275ms with PZO line low
0447   FD
0448   A9             LDA     #$23
0449   23
044A   8D             STA     PBDR    ;Take PZO line high
044B   00
044C   90
044D   A2             LDX     #$FF    ;Count value for delay 2
044E   FF
044F   CA   DELAY2:  DEX
0450   D0             BNE     DELAY2  ;Wait for 1.275ms again
0451   FD
0452   4C             JMP     LOOP    ;Loop back
0453   05
0454   04
```

14.11a In your solution to Practical Assignment 14.11, which bit position of Data Direction Register B was written with a logic '0'?

d Bit 2

14.13 Practical Assignment

Write a program which will allow the speed of the DC Motor to be varied according to the setting of the Potentiometer.

Typical Solution

```
                PADDR:   EQU     $9003
                PADR:    EQU     $9001
                PBDDR:   EQU     $9002
                PBDR:    EQU     $9000

                         ORG     $0400 ;Defines the start address
0400   A9       START:   LDA     #$0B
0401   0B
0402   8D               STA     PBDDR ;Configures Port B:
0403   02                             ;PB3=O/P, PB2=I/P
0404   90                             ;PB1=O/P, PB0=O/P
0405   A9       LOOP:    LDA     #$0B
0406   0B
0407   8D               STA      PBDR ;Outputs a "1" on:
0408   00                             ;PB3, PB1 and PB0
0409   90
040A   A9               LDA     #$00
040B   00
040C   8D               STA     PADDR ;Configures Port A as all inputs
040D   03
040E   90
040F   A9               LDA     #$09
0410   09
0411   8D               STA     PBDR  ;Outputs a "0" on PB1 (WR)
0412   00
0413   90
0414   A9               LDA     #$0B
0415   0B
0416   8D               STA     PBDR  ;Outputs a "1" on PB1 (WR)
0417   00
0418   90
0419   A9       TSTB2:  LDA     #$04
041A   04
041B   2C               BIT     PBDR  ;Tests PB2 (BSY) for logic "1"
041C   00
041D   90
041E   F0               BEQ     TSTB2 ;Repeat test of PB2 until
041F   F9                             ;PB2=logic 1 (Conversion completed)
0420   A9               LDA     #$03
0421   03
```

Continued...

Continued...

```
0422   8D            STA     PBDR   ;Outputs a "0" on PB3 (RD) to enable
0423   00                           ;ADC output
0424   90
0425   AE            LDX     PADR   ;Reads Port A input into X Register
0426   01
0427   90
0428   A9            LDA     #$0B
0429   0B
042A   8D            STA     PBDR   ;Outputs a "1" on PB3 to disable
042B   00                           ;ADC output
042C   90
042D   A9            LDA     #$FF
042E   FF
042F   8D            STA     PADDR  ;Reconfigures Port A as all outputs
0430   03
0431   90
0432   8E            STX     PADR   ;Outputs potentiometer value at
0433   01                           ;Port A
0434   90
0435   A9            LDA     #$0A
0436   0A
0437   8D            STA     PBDR   ;Outputs a "0" on PB0 to enable DAC
0438   00
0439   90
043A   A0            LDY     #$08
043B   08
043C   A2            LDX     #$FF   ;Loads X and Y registers
043D   FF                           ;with delay values
043E   CA   DELAY:   DEX
043F   D0            BNE     DELAY
0440   FD
0441   88            DEY
0442   D0            BNE     DELAY  ;Wait for 10ms so output is
0443   FA                           ;displayed more often than input
0444   4C            JMP     LOOP   ;Loop back
0445   05
0446   04
```

14.13a Run your program for Practical Assignment 14.13. Set the potentiometer to a point midway between the maximum and minimum settings. Enter the hexadecimal byte output at Port A.

Answer: range 60_H to $A0_H$

Student Assessment 14

1. For the Piezo Sounder to produce an audio frequency, a TTL signal must be applied to:

   a **Port B, bit 5**

2. The Ultrasonic Transmitter is switched on/off by the state of:

   b **Port B, bit 6**

3. When the Ultrasonic Receiver detects a 40kHz ultrasound signal:

   d **PB7 has a 40kHz squarewave**

4. The section of the Applications Module which allows the microprocessor to produce an Analog output is the:

   b **DAC**

5. An increase of 01_H at the input of the Applications Module DAC produces a rise in output voltage of:

   b **10mV**

6. The section of the Applications Module which allows the microprocessor to read an Analog input is the:

   a **ADC**

7. The signal from the Applications Module ADC which indicates that conversion is complete is:

   c $\overline{\textbf{BSY}}$

8. The Applications Module units which could be used to form an ambient light measuring system are the:

   c **Optical Receiver and the ADC**

9. The number of pulses per revolution produced by the Applications Module Optical Disc Encoder is:

   c **2**

10. The effect of applying alternate logic '1' and logic '0' repeatedly at Port B, bit 5, with a delay of 0.1ms between each change, would be an output of:

    a **5 kHz at the Piezo Sounder**

Continued...

Student Assessment 14 Continued ...

11. The effect on the Applications Module of the program section:

```
LDA  #$40
STA  PBDDR
STA  PBDR
```

would be to:

c **switch the Ultrasonic Transmitter on.**

12. The program section required to enable the DAC is:

a
```
LDA  #$01
STA  PBDDR
LDA  #$00
STA  PBDR
```

13. For the Applications Module ADC, conversion is initiated by applying an output of:

c **a short negative-going pulse to Port B, bit 1**

Chapter 15 Stack and Subroutines

15.3a The Stack Pointer is initially set to $01E0_H$. Enter the contents of the Stack Pointer after 5 bytes have been saved on the Stack.

Answer: $01DB_H$

15.3b The Stack Pointer is set to 0152_H. Enter the hexadecimal contents of the Stack Pointer after the instruction "`PHA`" has been executed.

Answer: 0151_H

15.4a The function of a "`JSR`" instruction is to:

[d] transfer program execution to a subroutine

15.5a The Stack Pointer register initially contains 0147_H. Enter the contents of the Stack Pointer after the program for Worked Example 15.5 has been executed.

Answer: 0147_H

15.6 Practical Assignment

Write a subroutine which will use the Stack to exchange the contents of the X Register and the Status Register. The Stack should be used to preserve the contents of any other registers used by the subroutine.

Typical Solution

```
                    ORG  $0400        ;Defines the start address
0400   48           PHA               ;Saves accumulator on stack
0401   08           PHP               ;Saves status register on stack
0402   8A           TXA
0403   48           PHA               ;Saves X Register on stack
0404   28           PLP               ;Restores status register from
                                       ;stack(X Register)
0405   68           PLA
0406   AA           TAX               ;Restores X Register from stack
                                       ;(Status Register)
0407   68           PLA               ;Restores accumulator from stack
0408   60           RTS               ;Returns from subroutine
```

15.6a The first two instructions in your program for Practical Assignment 15.6 are:

a PHA and PHP

15.7a Enter the number of times that the delay subroutine is called during each pass through the program of Worked Example 15.7.

Answer: 2

15.9a The program for Worked Example 15.9 must be modified to display the character "*Z*". Enter the hexadecimal byte which the first instruction must write to the Accumulator.

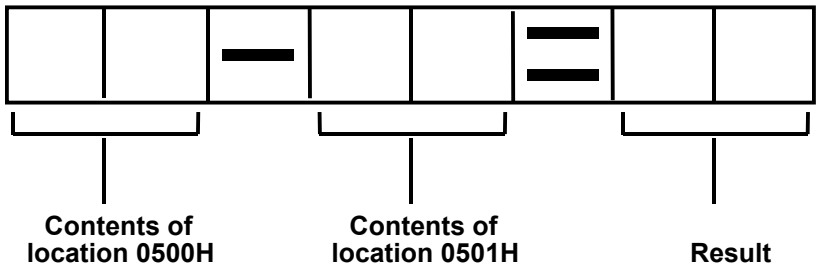
Answer: $5A_H$

15.10a In the program above, the instruction that tests the next character to see if the end of the buffer has been reached is:

a BEQ HERE

15.11 Practical Assignment

Write a program which uses MAC III monitor subroutines to subtract the hexadecimal contents of location 0501_H from the contents of location 0500_H and display the following on the seven-segment displays:



Typical Solution

```
                WRCHAR:     EQU     $C048
                WRBYTE:     EQU     $C04C

                            ORG     $0400     ;Defines the start address
0400    AD                  LDA     $0500
0401    00
0402    05
0403    20                  JSR     WRBYTE    ;Display contents of
0404    4C                                    ;location 0500H
0405    C0
0406    A9                  LDA     #$2D      ;ASCII code for "-" is
0407    2D                                    ;2DH
0408    20                  JSR     WRCHAR    ;Display "-"
0409    48
040A    C0
040B    AD                  LDA     $0501
040C    01
040D    05
040E    20                  JSR     WRBYTE    ;Display contents of
040F    4C                                    ;location 0501H
0410    C0
0411    A9                  LDA     #$3D      ;ASCII code for "=" is
0412    3D                                    ;3DH
0413    20                  JSR     WRCHAR    ;Display "="
0414    48
0415    C0
```

Continued...

Continued...

```
0416   D8            CLD              ;Selects binary arithmetic mode
0417   38            SEC              ;Prepare for subtraction
0418   AD            LDA    $0500
0419   00
041A   05
041B   ED            SBC    $0501     ;Subtract contents of
041C   01                              ;location 0501H from
041D   05                              ;contents of location 0500H
041E   20            JSR    WRBYTE    ;Display result
041F   4C
0420   C0
0421   4C  HERE:     JMP    HERE      ;Wait forever
0422   21
0423   04
```

15.11a Load location 0500_H with 87_H and location 0501_H with 39_H. Run your program for Practical Assignment 15.11 and enter the byte shown as the result.

Answer: $4E_H$

15.12 Practical Assignment

Write a program, using MAC III monitor subroutines, which will produce an increasing binary count at Port A. The count should be incremented once per second.

Typical Solution

```
                PADDR:    EQU      $9003
                PADR:     EQU      $9001
                WTNMS:    EQU      $C058

                          ORG      $0400     :Defines the start address
0400    A9                LDA      #$FF
0401    FF
0402    8D                STA      PADDR     ;Configures Port
0403    03                                   ;A as an output Port
0404    90
0405    A9                LDA      #$00
0406    00
0407    8D                STA      PADR      ;Outputs 0000
0408    01                                   ;000 binary
0409    90                                   ;initially
040A    A9    NEXT:       LDA      #$FA      ;Pass parameter to subroutine
040B    FA                                   ;WTNMS for 250 x 1ms=250ms
                                             ;(250 denary=FAH)
040C    A2                LDX      #$04
040D    04
040E    20    DELAY:      JSR      WTNMS     ;Delay of 250ms
040F    58
0410    C0
0411    CA                DEX
0412    D0                BNE      DELAY     ;Wait for 4 x 250ms
0413    FA                                   ;= 1 second
0414    EE                INC      PADR      ;Add one to Port A output
0415    01
0416    90
0417    4C                JMP      NEXT      ;Jump back for
0418    0A                                   ;next count
0419    04
```

15.12a In your solution to Practical Assignment 15.12, the MAC III System subroutine which could be used to give a 1 second delay is called:

d WTNMS

15.12b Run your program for Practical Assignment 15.12. Wait for 25 seconds and read the binary count value which is displayed at Port A. Enter this count value as a hexadecimal number.

Answer: in the range 17_H to $1B_H$

15.13 Practical Assignment

Write a program which will sound the piezo sounder whenever the "S" key is held down on the MAC III keypad.

Typical Solution

```
                PBDDR:      EQU       $9002
                PBDR:       EQU       $9000
                WT1MS:      EQU       $C054
                GETIN:      EQU       $C050

                            ORG       $0400       ;Defines the start address
0400    A9                  LDA       #$20
0401    20
0402    8D                  STA       PBDDR       ;Configures PB5 as an Output
0403    02
0404    90
0405    20      TSTKEY:     JSR       GETIN       ;Waits for a key closure
0406    50                                        ;and passes key value to the
0407    C0                                        ;accumulator
0408    B0                  BCS       TSTKEY
0409    FB
040A    C9                  CMP       #$53
040B    53
040C    D0                  BNE       TSTKEY      ;If "S" key not pressed, branch
040D    F7                                        ;back to wait for another key
040E    A9                  LDA       #$20
040F    20
0410    8D                  STA       PBDR        ;Outputs a "1" on PB5
0411    00
0412    90
0413    20                  JSR       WT1MS       ;Delay of 1ms
0414    54
0415    C0
0416    A9                  LDA       #$00
0417    00
0418    8D                  STA       PBDR        ;Outputs a "0" on PB5
0419    00
041A    90
041B    20                  JSR       WT1MS       ;Delay of 1ms
041C    54
041D    C0
041E    4C                  JMP       TSTKEY      ;Loop back to wait for
041F    05                                        ;another key closure
0420    04
```

Note: *If an attempt is made to run this program from the PC using the cross assembler Terminal mode, problems will be encountered due to the auto-repeat facility of the PC keyboard.*

15.13a In your solution to Practical Assignment 15.13, the instruction used to check if the S (and no other) key has been pressed is a:

a Compare

15.14 Practical Assignment

Write a program, using MAC III monitor subroutines, that will allow the speed of the DC Motor to be controlled by the "+" and "-" keys. The motor should slowly accelerate when the "+" key is pressed, hold speed constant if no keys are pressed and decelerate when the "-" key is pressed.

Typical Solution

```
                PADDR:    EQU     $9003
                PADR:     EQU     $9001
                PBDDR:    EQU     $9002
                PBDR:     EQU     $9000
                GETIN:    EQU     $C050
                WTNMS:    EQU     $C058

                          ORG     $0400     ;Defines the start address
0400   A9                 LDA     #$FF
0401   FF
0402   8D                 STA     PADDR     ;Configures Port A as an
0403   03                                   ;output port
0404   90
0405   A9                 LDA     #$01
0406   01
0407   8D                 STA     PBDDR     ;Configures PB0 as an
0408   02                                   ;output bit
0409   90
040A   A9                 LDA     #$00
040B   00
040C   8D                 STA     PBDR      ;Outputs a "0" on PB0 to
040D   00                                   ;enable DAC
040E   90
040F   A9                 LDA     #$00
0410   00
0411   8D                 STA     PADR      ;Initially Port A output
0412   01                                   ;= 0000 0000 binary
0413   90
0414   20   TSTKEY:   JSR     GETIN     ;Waits for a key closure
0415   50                                   ;and passes the key value
0416   C0                                   ;to the accumulator
```

Continued...

Continued...

```
0417   48            PHA              ;Saves key value on stack
0418   A9            LDA     #$20
0419   20
041A   20            JSR     WTNMS    ;Delay of 20 x 1ms = 20ms
041B   58                             ;to allow for switch bounce
041C   C0
041D   68            PLA              ;Restores key value from stack
041E   B0            BCS     TSTKEY   ;If C=1 then wait for a
041F   F4                             ;key closure again
0420   C9            CMP     #$2B     ;Tests ASCII key value for "+"
0421   2B
0422   F0            BEQ     FASTER   ;If "+" key, increase speed
0423   07
0424   C9            CMP     #$2D     ;Tests ASCII key value for "-"
0425   2D
0426   F0            BEQ     SLOWER   ;If "-" key, decrease speed
0427   09
0428   4C            JMP     TSTKEY   ;Loop back to wait for
0429   14                             ;another key closure
042A   04
042B   EE   FASTER:   INC     PADR     ;Increment output at
042C   01                             ;Port A
042D   90
042E   4C            JMP     TSTKEY   ;Loop back to wait for
042F   14                             ;another key closure
0430   04
0431   CE   SLOWER:  DEC     PADR     ;Decrement output at
0432   01                             ;Port A
0433   90
0434   4C            JMP     TSTKEY   ;Loop back to wait for a
0435   14                             ;another key closure
0436   04
```

15.14a Run your program for Practical Assignment 15.14. The effect of pressing the [S] key is that:

[d] motor speed is unchanged

15.14b In your program for Practical Assignment 15.14, Port B is configured by writing a hexadecimal byte to Data Direction Register B. Enter the bit number of this register which must be at logic "1".

Answer: 0

Student Assessment 15

1. In a LIFO stack, the last data word stored will be restored:
   b **first**
2. The last stack location used is defined by the contents of the:
   b **Stack Pointer Register**
3. The Stack Pointer contains $015D_H$. After the instruction "`PLA`" has been executed, the Stack Pointer will contain:
   c **$015E_H$**
4. The 6502 instruction that saves data on the stack is called:
   b **PUSH**
5. A sequence of object code that appears once but which may be used several times is called a:
   d **Subroutine**
6. When a subroutine is called, the return address is saved:
   a **on the Stack**
7. The 6502 instruction that transfers program execution to a subroutine is:
   d **JSR**
8. The 6502 instruction that usually occurs at the end of a subroutine is:
   b **RTS**
9. The 6502 instruction sequence that will save the X Register on the Stack is:
   d
   ```
   TXA
   PHA
   ```
10. The MAC III monitor subroutine that allows ASCII characters to be written to the display is:
    b **WRCHAR**
11. If a key is pressed, the MAC III monitor subroutine "GETIN" will place the corresponding value in the:
    a **Accumulator**

Chapter 16 Interrupts

16.1a Usually, when an interrupt service routine has been completed:

[c] the interrupted program is resumed

16.3a If location 0789_H contains 50_H and location $078A_H$ contains 00_H, the instruction "`JMP ($0789)`" will cause program execution to continue from location:

[a] 0050_H

16.5a In the 6502, maskable interrupts are prevented from interrupting the processor by:

[d] setting the I-flag

16.6a The program for Worked Example 16.6 is to be modified so that the NMI routine starts at location 0580_H. Enter the address for the memory location that must be changed.

Answer: 0200_H

16.7 Practical Assignment

Write a program which will activate the piezo sounder if a non-maskable interrupt occurs.

Typical Solution

```
            PBDDR:    EQU    $9002
            PBDR:     EQU    $9000
;Main Program:
                      ORG    $0400    ;Main Program start address
0400   4C   HERE:     JMP    HERE     ;Loop forever - dummy
0401   00                             ;main program
0402   04
;NMI Vectors:
                      ORG    $0200
0200   00             WORD   $0500    ;NMI vectors point to
0201   05                             ;location 0500H
;NMI routine: Sound Piezo Sounder
                      ORG    $0500    ;NMI routine start address
0500   A9             LDA    #$20
0501   20
0502   8D             STA    PBDDR    ;Configures Port A bit 5
0503   02                             ;as an output bit
0504   90
0505   A9   LOOP:     LDA    #$20
0506   20
0507   8D             STA    PBDR     ;Outputs a "1" on PB5
0508   00
0509   90
050A   20             JSR    DELAY    ;Calls delay of 500µs
050B   80
050C   04
050D   A9             LDA    #$00
050E   00
050F   8D             STA    PBDR     ;Outputs a "0" on PB5
0510   00
0511   90
0512   20             JSR    DELAY    ;Calls delay of 500µs
0513   80
0514   04
0515   4C             JMP    LOOP     ;Loops back to output a
0516   05                             ;"1" on PB5 again
0517   05
;Delay Subroutine:
                      ORG    $0480    ;Delay subroutine start address
0480   A2   DELAY:    LDX    #$64
0481   64
0482   CA   CNT1:     DEX
0483   D0             BNE    CNT1     ;Wait 500µs
0484   FD
0485   60             RTS
```

16.7a In your program for Practical Assignment 16.7, the program section which produces an output on the piezo sounder is within the:

b NMI Routine

16.8a The effect of removing the instruction at location 0400_H in the program for Worked Example 16.8 would be to:

c prevent the main program from being interrupted

16.9a The effect of removing the instruction at location 0400_H in the program for Worked Example 16.9 would be to:

d only allow a NMI to interrupt the main program

16.10 Practical Assignment

Write a program which will continually output 99_H at Port A. If a non-maskable interrupt occurs, the piezo sounder should also be activated.

Typical Solution

```
                PADDR:    EQU      $9003
                PADR:     EQU      $9001
                PBDDR:    EQU      $9002
                PBDR:     EQU      $9000
;Main Program:
                          ORG      $0400          ;Main Program start address
0400    A9                LDA      #$FF
0401    FF
0402    8D                STA      PADDR          ;Configures Port A as
0403    03                                        ;an output port
0404    90
0405    A9                LDA      #$20
0406    20
0407    8D                STA      PBDDR          ;Configures Port B bit 5
0408    02                                        ;(PB5) as an   output bit
0409    90
040A    A9    OUT99:      LDA      #$99
040B    99
040C    8D                STA      PADR           ;Outputs 99H at Port A
040D    01
040E    90
040F    4C                JMP      OUT99          ;Repeat forever
0410    0A
0411    04
;NMI Vectors:
                          ORG      $0200
0200    00                WORD     $0500          ;NMI vectors point to
0201    05                                        ;location 0500H
;NMI Routine - Sound Piezo Sounder
                          ORG      $0500          ;NMI routine start address
0500    A0                LDY      #$10           ;Loads burst count
0501    10
0502    A9    PIEZO:      LDA      #$20
0503    20
0504    8D                STA      PBDR           ;Outputs a "1" on PB5
0505    00
0506    90
```

Continued ...

Continued…

```
0507   20             JSR     DELAY   ;Calls delay of 500µs
0508   80
0509   04
050A   A9             LDA     #$00
050B   00
050C   8D             STA     PBDR    ;O/P a "0" on PB5
050D   00
050E   90
050F   20             JSR     DELAY   ;Calls delay of 500µs
0510   80
0511   04
0512   88             DEY             ;Decrements burst count.
0513   D0             BNE     PIEZO   ;If burst not finished,
0514   ED                             ;branch back to repeat
0515   40             RTI             ;Returns to main program
;Delay Subroutine:
                      ORG     $0480   ;Delay subroutine start address
0480   A2   DELAY:    LDX     #$64
0481   64
0482   CA   COUNT:    DEX
0483   D0             BNE     COUNT   ;Wait 500µs
0484   FD
0485   60             RTS             ;Returns to NMI routine
```

16.10a In your program for Practical Assignment 16.10, the section of the program which configures the Ports is the:

[a] Main Program

16.12a The interrupt vectors for a 6502 Software Interrupt (BRK) are at locations:

[d] $FFFE_H$ and $FFFF_H$

16.12b The 6502 interrupt which does not save the current program counter contents on the stack is:

[d] Reset

16.13a The MAC III location that is used to control the Auto-Run facility is:

[b] 0206_H

16.14 Practical Assignment

Write a program which displays "HELLO" on the MAC III display. If a non-maskable interrupt occurs the display should change to "NON MASK". If a maskable interrupt occurs the display should change to "MASKABLE".

Typical Solution

```
;ASCII codes for messages:
                    ORG     $0040
0040    48          BYTE    "HELLO"       ;Codes for "HELLO"
0041    45
0042    4C
0043    4C
0044    4F
0045    00          BYTE    0             ;End code
                    ORG     $0050
0050    4E          BYTE    "NON MASK"    ;Codes for "NON MASK"
0051    4F
0052    4E
0053    20
0054    4D
0055    41
0056    53
0057    4B
0058    00          BYTE    0             ;End code
                    ORG     $0060
0060    4D          BYTE    "MASKABLE"    ;Codes for "MASKABLE"
0061    41
0062    53
0063    4B
0064    41
0065    42
0066    4C
0067    45
0068    00          BYTE    0             ;End code
;Interrupt Vectors:
                    ORG     $0200
0200    00          WORD    $0500         ;NMI vector points to
0201    05                                ;location 0500H
0202    80          WORD    $0580         ;IRQ vector points to
0203    05                                ;location 0580H
;Background program - display "HELLO":
             WRCHAR: EQU     $C048
                    ORG     $0400         ;Background Program start address
0400    58          CLI                   ;Enable maskable interrupts
0401    A2          LDX     #$00          ;Defines start of display
0402    00                                ;buffer
0403    B5   NEXT:  LDA     $40,X         ;Read next character
0404    40
0405    F0          BEQ     FINSH         ;If value = 0, finish
0406    07                                ;display
```

Continued…

Continued ...

```
0407    20              JSR     WRCHAR   ;Call display subroutine
0408    48
0409    C0
040A    E8              INX
040B    4C              JMP     NEXT     ;Loop back for next character
040C    03
040D    04
040E    4C    FINSH:    JMP     FINSH    ;Wait forever to allow
040F    0E                               ;steady display
0410    04
;NMI service routine: display "NON MASK":
                        ORG     $0500    ;NMI routine start address
0500    58              CLI              ;Enable maskable interrupts
0501    A2              LDX     #$10     ;Re-defines start of
0502    10                               ;display buffer
0503    4C              JMP     NEXT     ;Display message
0504    03
0505    04
;IRQ service routine: display "MASKABLE":
                        ORG     $0580    ;IRQ routine start address
0580    58              CLI              ;Enable maskable interrupts
0581    A2              LDX     #$20     ;Re-defines start of
0582    20                               ;display buffer
0583    4C              JMP     NEXT     ;Display message
0584    03
0585    04
```

Note the use of the BYTE Cross Assembler directive in the above Typical Solution. This allocates one byte of memory for each character which appears in double quotes after the BYTE directive, and initializes that byte to the corresponding ASCII code.

Where a value appears without quotes after a BYTE directive (for example, 0 in this program), then a byte of memory is directly initialized with that value.

16.14a In your program for Practical Assignment 16.14, the number of different interrupt service routines is:

[b] 2

16.14b Your program for Practical Assignment 16.14 is to be modified such that it will not respond to maskable interrupts. The part of the program which must be altered is the:

[a] main program

Student Assessment 16

1. An input to a microprocessor that causes it to suspend the current program is called:

   b **an Interrupt**

2. Usually, when an interrupt service routine has been completed:

   c **the interrupted program is resumed**

3. The process of a microprocessor periodically checking a peripheral to see if it is ready for data transfer is called:

   d **Polled Input/Output**

4. The main advantage of Interrupt Input/Output, as compared with Polled Input/Output is that it:

   a **does not waste microprocessor time**

5. An interrupt service routine which has been interrupted by a second interrupt is an example of:

   c **Nested Interrupts**

6. Interrupt inputs which the microprocessor may ignore are said to be:

   b **Maskable**

7. The 6502 instruction that allows maskable interrupts to be acknowledged is:

   a `CLI`

8. The 6502 instruction mnemonics for an indirect Jump to the location pointed to by locations 0400_H and 0401_H are:

   d `JMP ($0400)`

9. The vector for the 6502 NMI interrupt is at locations:

   b **$FFFA_H$ and $FFFB_H$**

10. The 6502 instruction that is usually found at the end of an interrupt service routine is:

    b `RTI`

11. The highest priority 6502 interrupt is:

    d **Reset**

12. The 6502 addressing mode used to redirect interrupt vectors is called:

    b **absolute indirect**