|  |  |  |
| --- | --- | --- |
| **Rubrics form for Microprocessor Systems and Interfacing** | | |
| **Lab #:** | **04** | |
| **Lab Title:** | Introduction to Addressing Modes and Program Status Word Register | |
| **Submitted by:** | | |
| **Name** | | **Registration #** |
| **SAAD RAFIQUE**  **WAJEH UL HASSAN** | | **FA19-BCE-020**  **FA19-BCE-024** |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***Rubrics to follow*** | | | | | | | | | | | | | |
| **Rubrics #** | **R1** | **R2** | **R3** | **R4** | **R5** | **R6** | **R7** | **R8** | **R9** | **R10** | **R11** | **R12** | **R13** |
| **In Lab** |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Post- Lab** |  |  |  |  |  |  |  |  |  |  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Rubrics name & number** | | **Marks** | |
| **In-Lab** | **Post-Lab** |
| **Engineering Knowledge** | ***R2: Use of Engineering Knowledge and follow Experiment Procedures:***  *Ability to follow experimental procedures, control variables, and record procedural steps on lab report.* |  |  |
| ***R3: Interpretation of Subject Knowledge:***  *Ability to interpret and explain mathematical and/or visual forms, including equations, diagrams, graphics, figures and tables.* |  |  |
| **Problem Analysis** | ***R4: Limitations and***  ***Implications:***  *Ability to point out*  *limitations and implications of Hardware and software Components.* |  |  |
| ***R5: Data/Evidence Measurements:***  *Ability to record raw data / evidence.* |  |  |
| ***R6: Experimental Data Analysis:***  *Ability to interpret findings, compare them to values in the literature, identify weaknesses and limitations.* |  |  |
| **Design** | **R7: Implementing Design Strategy:** Ability to execute a solution taking into consideration design requirements and pertinent contextual elements.  [Block Diagram/Flow chart/Circuit Diagram] |  |  |
| ***R8: Best Coding Standards:***  *Ability to follow the coding standards and programming practices.* |  |  |
| **Modern Tools Usage** | ***R11: Tools Evaluation:***  *Ability to simulate the experiment and then using hardware tools to verify the results.* |  |  |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Rubrics #** | R2 | R3 | R4 | R5 | R6 | R7 | R8 | R11 |
| **In –Lab** |  |  |  |  |  |  |  |  |
| **Post- Lab** |  |  |  |  |  |  |  |  |

**LAB#4**

**Introduction to Addressing Modes and Program Status Word Register**

**Objectives:**

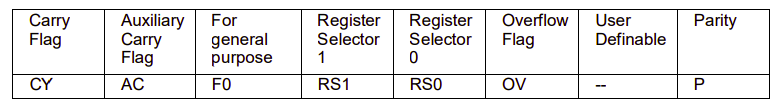
The objective of this lab is to get awareness and simulate in uVision different addressing modes for different instructions, PSW Register and the condition on which they change.

**Introduction:**

**PSW (Program Status Word) Register:**

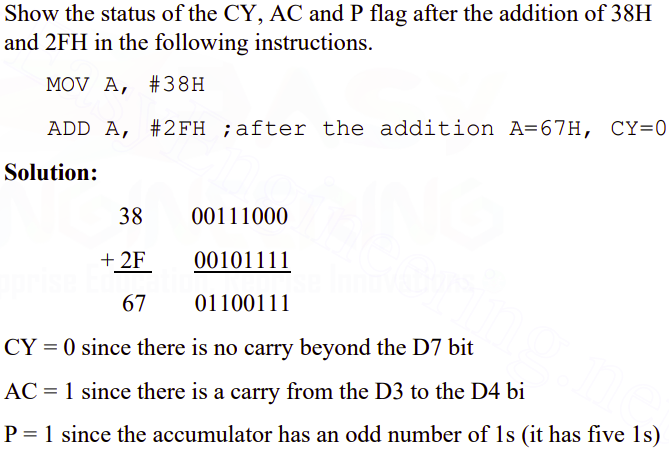
The PSW register is 8 bits wide. Four of the flags are called conditional flags, meaning that they indicate some conditions that result after an instruction is executed. Two bits are unused and two are to select register bank.

* **Carry Flag:** Set to 1 when there is carry from D7 to D8
* **Auxiliary Carry Flag:** Set to 1 when there is carry form D3 to D4
* **Overflow:** Set to 1 when the result of signed number operation is too large to accommodate in 8 bit.
* **Parity:** Even parity, if the A register contains an odd number of is, then P = 1

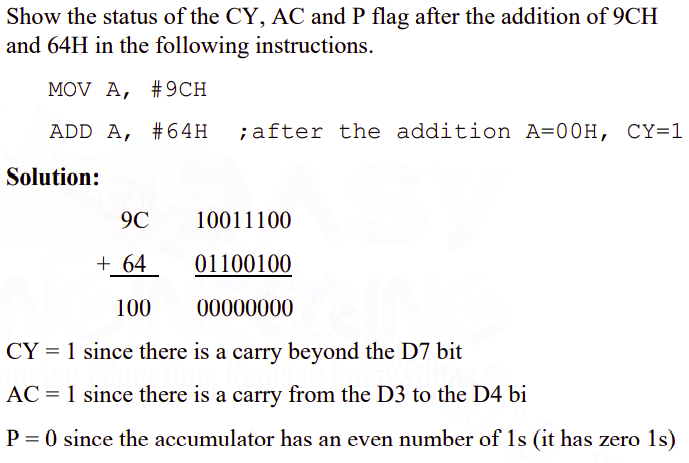


The flag bits affected by ADD instruction are **CY**, **P**, **AC** and **OV**.

**Example 1:**



**Example 2:**



**In-Lab Tasks**

**Task#1**

Write code and debugging results in tabular form for the following number’s addition and subtraction using immediate and register addressing.

1. 3567A4H + A679BEH
2. 3567A4H - A679BEH

Examine the PSW register after each instruction and explain it with the help of dry run.

1. **3567A4H + A679BEH**

**Task description:**

In this task we are required to add two hexadecimal numbers that are bigger than one byte, So that we cannot add them directly. In order to add these two numbers we have to add them conventionally.

|  |
| --- |
| **Code:** |
| ORG 0H;  MOV A, #0A4H ;Moving A4H in accumulator  ADD A, #0BEH ;Adding BEH with the value in accumulator and store result in accumulator  MOV R0, A ;Moving value of accumulator in R0 register  MOV A, #67H ;Moving 67H in accumulator  ADDC A, #79H ;Adding carry and 79H with the value in accumulator and store result in accumulator  MOV R1, A ;Moving value of accumulator in R1 register  MOV A, #35H ;Moving 35H in accumulator  ADDC A, #0A6H ;Adding carry and A6H with the value in accumulator and store result in accumulator  MOV R2, A ;Moving value of accumulator in R2 register  END; |

|  |  |
| --- | --- |
| **Code and Dry Run** | **Register Status** |
| |  |  | | --- | --- | | **Instruction** | **Dry Run** | | MOV A, #0A4H | A=10100100  PSW=00000001  P=1 | |  |
| |  |  | | --- | --- | | **Instruction** | **Dry Run** | | ADD A, #0BEH | A=A+BEH  A=10100100+10111110  A=01100010  CY=1  P=1  AC=1  OV=1  PSW=11010101 | | MOV R0, A | R0=01100010 | |  |
| |  |  | | --- | --- | | **Instruction** | **Dry Run** | | MOV A, #67H | A=01100111  CY=1  P=1  AC=1  OV=1  PSW=11010101 | |  |
| |  |  | | --- | --- | | **Instruction** | **Dry Run** | | ADD A, #0BEH | A=A+79H  A=01100111+01111001  A=11100001  CY=0  P=0  AC=1  OV=1  PSW=01000100 | | MOV R1, A | R1=11100001 | |  |
| |  |  | | --- | --- | | **Instruction** | **Dry Run** | | MOV A, #35H | A=00110101  CY=0  P=0  AC=1  OV=1  PSW=01000100 | |  |
| |  |  | | --- | --- | | **Instruction** | **Dry Run** | | ADD A, #0BEH | A=A+A6H  A=00110101+10100110  A=11011011  CY=0  P=0  AC=0  OV=0  PSW=00000000 | | MOV R2, A | R2=11011011 | |  |

**Results and Discussion:**

The most significant nibble of the result is stored in R2 and the least significant nibble is stored in R0 and the whole result is stored in R0, R1 and R2, this because we cannot store 24 bit result in any of the register of AT89C51 MC.

1. **3567A4H - A679BEH**

**Task description:**

In this task we are required to subtract two hexadecimal numbers that are bigger than one byte, So that we cannot subtract them directly. In order to subtract these two numbers we have to subtract them conventionally.

|  |
| --- |
| **Code:** |
| ORG 0H;  MOV A, #0BEH ;Moving BEH in accumulator  SUBB A, #0A4H ;Subtracting A4H with the value in accumulator and store result in accumulator  MOV R0, A ;Moving value of accumulator in R0 register  MOV A, #79H ;Moving 79H in accumulator  SUBB A, #67H ;Subtracting carry and 67H with the value in accumulator and store result in accumulator  MOV R1, A ;Moving value of accumulator in R1 register  MOV A, #0A6H ;Moving A6H in accumulator  SUBB A, #35H ;Subtracting carry and 35H with the value in accumulator and store result in accumulator  MOV R2, A ;Moving value of accumulator in R2 register  END |

|  |  |
| --- | --- |
| **Code and Dry Run** | **Register Status** |
| |  |  | | --- | --- | | **Instruction** | **Dry Run** | | MOV A, #0BEH | A=10111110  PSW=00000000 | |  |
| |  |  | | --- | --- | | **Instruction** | **Dry Run** | | SUBB A, #0A4H | A=A-A4H  A=10111110-10100100  A=00011010  P=1  PSW=00000001 | | MOV R0, A | R0=00011010 | |  |
| |  |  | | --- | --- | | **Instruction** | **Dry Run** | | MOV A, #79H | A=01111001  P=1  PSW=00000001 | |  |
| |  |  | | --- | --- | | **Instruction** | **Dry Run** | | SUBB A, #67H | A=A-67H  A=01111001-01100111  A=00010010  PSW=00000000 | | MOV R1, A | R1=00010010 | |  |
| |  |  | | --- | --- | | **Instruction** | **Dry Run** | | MOV A, #0A6H | A=10100110  PSW=00000000 | |  |
| |  |  | | --- | --- | | **Instruction** | **Dry Run** | | SUBB A, #35H | A=A-35H  A=10100110  -00110101  A=01110001  OV=1  PSW=00000100 | | MOV R2, A | R2=01110001 | |  |

**Results and Discussion:**

The most significant nibble of the result is stored in R2 and the least significant nibble is stored in R0 and the whole result is stored in R0, R1 and R2, this because we cannot store 24 bit result in any of the register of AT89C51 MC.

**Task#2**

Write code and debugging results in tabular form for the following number’s addition and subtraction.

ABEDA123H + CDEAB272H – 37456789H

Examine the PSW register after each instruction and explain it with the help of dry run.

**Task description:**

In this task we are required to add two hexadecimal numbers and then subtract the result with an other hexadecimal number that are bigger than one byte, So that we cannot add or subtract them directly. In order to calculate the result we have to subtract them conventionally.

|  |
| --- |
| **Code:** |
| ORG 0H;  MOV A, #23H ;Moving BEH in accumulator  ADD A, #72H ;Subtracting A4H with the value in accumulator and store result in accumulaltor  MOV R0, A ;Moving value of accumulator in R0 register  MOV A, #0A1H ;Moving 79H in accumulator  ADDC A, #0B2H ;Subtracting carry and 67H with the value in accumulator and store result in accumulaltor  MOV R1, A ;Moving value of accumulator in R1 register  MOV A, #0EDH ;Moving A6H in accumulator  ADDC A, #0EAH ;Subtracting carry and 35H with the value in accumulator and store result in accumulaltor  MOV R2, A ;Moving value of accumulator in R2 register  MOV A, #0ABH ;Moving A6H in accumulator  ADDC A, #0CDH ;Subtracting carry and 35H with the value in accumulator and store result in accumulaltor  MOV R3, A ;Moving value of accumulator in R2 register  MOV A, R0  SUBB A, #89H  MOV R0, A ;Moving value of accumulator in R0 register  MOV A, R1  SUBB A, #67H  MOV R1, A ;Moving value of accumulator in R0 register  MOV A, R2  SUBB A, #45H  MOV R2, A ;Moving value of accumulator in R0 register  MOV A, R3  SUBB A, #37H  MOV R3, A ;Moving value of accumulator in R0 register  END |

|  |  |
| --- | --- |
| **Code and Dry Run** | **Register Status** |
| A B E D A 1 2 3 H  + C D E A B 2 7 2 H  1 7 9 D 8 5 3 9 5 H  7 9 D 8 5 3 9 5 H  - 3 7 4 5 6 7 8 9 H  4 2 9 2 E C 0 B H |  |

**Results and Discussion:**

The most significant nibble of the result is stored in R3 and the least significant nibble is stored in R0 and the whole result is stored in R0, R1, R2 and R3, this because we cannot store 32 bit result in any of the register of AT89C51 MC.

**Post-Lab Task**

**Task#1**

Perform the following instructions:

* Using Immediate Addressing write and add 55H to A
* Using Register Addressing write and add 10H to A and R0
* Using Direct Addressing write and add 55H to A from R4
* Using Register Indirect Addressing mode write 55H to address 40H on RAM and A, then 40H address Add to A

Explain the addressing mode in with each instruction.

**Task Description:**

In this task we performed different addressing modes like Immediate Addressing, Register Addressing, Direct Addressing and Register Indirect Addressing mode by applying different set of instructions on them

* Using Immediate Addressing write and add 55H to A

|  |
| --- |
| **Code:** |
| ORG 0000H  MOV A,#55H /\*As A=0 by default so adding 55H to A which means 0+55H which is 55\*/  END |

|  |  |
| --- | --- |
| **Code and Dry Run** | **Register Status** |
| MOV A,#55H |  |

* Using Register Addressing write and add 10H to A and R0

|  |
| --- |
| **Code:** |
| /\*\*\*\*\* Using Register Addressing write and add 10H to A and R0\*\*\*\*/  ORG 0000H  MOV A,#10H /\*Storing 10 in A\*/  ADD A,R0 /\*ADDING A+R0=10:::both source and destination are registers:::\*/  END |

|  |  |
| --- | --- |
| **Code and Dry Run** | **Register Status** |
| MOV A,#10H  ADD A,R0 |  |

* Using Direct Addressing write and add 55H to A from R4

|  |
| --- |
| **Code:** |
| /\*\*\*\* Using Direct Addressing write and add 55H to A from R4\*\*\*\*\*\*\*\*\*\*\*\*/  ORG 0000H  MOV 04H,#55H /\*Storing 55 AT 04 ADDRESS\*/  ADD A,04H /\*ADDING A+value at 04H address i-e 55H=55::source is register and destination is an address:::\*/  END |

|  |  |
| --- | --- |
| **Code and Dry Run** | **Register Status** |
| MOV 04H,#55H  ADD A,04H |  |

* Using Register Indirect Addressing mode write 55H to address 40H on RAM and A, then 40H address Add to A

|  |
| --- |
| **Code:** |
| /\*Using Register Indirect Addressing mode write 55H to address 40H on RAM and A, then 40H address Add to A\*/  ORG 0000H  MOV R1,#40H /\*Storing 40 IN R1 register\*/  MOV 40H,#55H /\*storing value at 40H address i-e 55H=55:::source is address and destination is value to be stored:::\*/  ADD A,@R1 /\*use value in R1 as an address and the value at that address is added withand value in A\*/  END |

|  |  |
| --- | --- |
| **Code and Dry Run** | **Register Status** |
| MOV R1,#40H  MOV 40H,#55H  ADD A,@R1 |  |

**Task#2**

Identify and explain the following addressing modes?

INC DPTR

SJMP 30H

CLR C

SUBB A,#43

MOV @R1,A

**INC DPTR**

It is used in immediate addressing mode.Increments the 16 bit data pointer by 1. DPTR is the only 16-bit register available and is often used to store 2-byte values.

**SJMP 30H**

It is used to jump to the new address.

**CLR C**

It is used to clear carry from PSW.

**SUBB A,#43**

In this subtraction is carried is direct addressing mode as in this addressing mode, the source or destination (or both source and destination) must be an address, but not value.

**MOV @R1,A**

‘@’ is used in indirect addressing mode. It means to use the value present in R1 as an address and the values at that address is to be stored in A.

# **Conclusion:**

* We get to know about addition and subtraction of the hexadecimal numbers. It is carried out in 8-8 bits.
* Different addressing modes for different instructions specifies how to calculate the effective memory address of an operand by using information held in registers or constants contained within a machine instruction.