**Assignment No.10**

|  |
| --- |
| **Title of Assignment:**  Write 80387 ALP to find the mean, variance, and standard deviation. |
| **Relevant Theory:**  **Explanation:**  **a) Features of 80387:**  • High performance 80-Bit Internal Architecture  • Implements ANSI/IEEE standard 754-1985 for Binary floating-point arithmetic  • Expands Intel386DX CPU data types to include 32-, 64-, 80-bit floating point, 32-, 64-bit  integers and 18-bit BCD operands  • Extends Intel386DX CPU instruction set to include Trigonometric, Logarithmic,  Exponential and Arithmetic instructions for all data types  • Upward object code compatible  • Full-range transcendental operations for SINE, COSINE, TANGENT, ARCTANGENT  and LOGARITHM  • Built-in Exception handling  • Operates independently in all modes of 80386  • Eight 80-bit Numeric registers  • Available in 68-pin PGA package  • One version supports 16MHz-33MHz  **80387 Functional block diagram:**    Figure . Intel387TM DX Math CoProcessor Block Diagram  **Register Set**   * Data registers: Eight 80-bit registers, * Tag Word: the tag word marks the content of each numeric data register, two bits for each data register * Status word: the 16-bit status word reflects the overall state of the MCP * Instruction and Data pointers: two pointer registers allows identification of the   failing numeric instruction which supply the address of failing numeric instruction  and the address of its numeric memory operand.   * Control Word: several processing options are selected by loading a control word   from memory into the control register  c) Instruction of co-processor used in the assignment:  FINIT: Initialise Co-processor  FLDZ: Load zero on stack top  FILD: Load Integer on stack  FIDIV: Divide stack top by an integer value  FIMUL: Multiply stack top by an integer value  FST: Store stack top  FADD: Add in stack top  FBSTP: Store integer part of stack top in 10 byte packed BCD format  FMUL: Multiply stack top  FSQRT: Square Root of Stack Top  FSTSW: Stores the coprocessor status word  FTS: compares ST0 and 0  **d) Concept of mean,variance and standard deviation:**  **mean = average of the numbers.**  **variance = (summation( ( Xi – average of numbers) \* ( Xi – average of numbers)) ) / Total no of elements. where i = 1 to N here N is the total no of elements.**  **Standard deviation = Squareroot of the variance.**  **Design Analysis/ Implementation Logic:**  **Algorithm**  i. Start  ii. Assign 5 elements in array in data section.  iii. Initialize the co-processor  iv Load 5 in CX register.  vi.Load zero on top of stack.  v.Assign the pointer to first element of array.  vi.Repeat step vii to viv until cx≠0  vii. Load element of array on stack top  viii. Add sto and st1. Now result is on top of stack.  ix. Increment pointer to access next element of array.  x. store the result on top of the stack  xi. Divide stack top with five  xii. store top of the stack in new variable let mean1.  xiii. Call Disp\_result procedure to display mean.  xiii. load first element on top of stack.  xiv. load 5 in cx register  xv. load zero on top of stack.  xvi. Initialize pointer to first element of array.  xvii. Repeat step xviii to xxi until cx≠0.  xviii. load element of array on top of stack.  xix. substract mean1 from top of stack.  xx. add st0 and st1.  xxi. increment the array pointer to access next array element.  xxii. Divide stack top by 5.  xxiii. Store top of stack in variable let variance.  xxiii. Call Disp\_result procedure to display variance  xiv . sqrt stack top.  xvii. Store top of stack in variable let deviation.  xviii. Call Disp\_result procedure to display variance  xiv. Stop |
| **Testing:**  **Test Conditions:**  **Input:**  **Input elementsin array**  **Output:**  Mean,variance,deviation. |
| **FAQs:**   1. Explain 80387 instruction 2. How 80387 is initialized.   . |
| **Conclusion:** Successfully implemented 80387 ALP to find the mean, variance, and standard deviation.  . |