CSE2006
Microprocessor & Interfacing

Module – 6
Co-Processor

Dr. E. Konguvel

Assistant Professor (Sr. Gr. 1),

Dept. of Embedded Technology,

School of Electronics Engineering (SENSE),

konguvel.e@vit.ac.in

9597812810

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

Module 6: Co-Processor

- Introduction
- 8087 Numeric Data Processor
- Block Diagram
- Pin Description
- Interfacing 8087 with 8086
- Addressing Modes & Data Formats
- **Instruction Sets**
- Assembly Language Programs

Instruction Sets

- 8087 co-processor has 68 additional instructions to the instruction set of 8086.
- These instructions are fetched by 8086 but are executed by 8087.
- When the 8086 comes across an 8087 instruction, it executes the ESCAPE instruction code to bypass the instruction opcode and control of the local bus to 8087 co-processor.
- The execution of 8087 instructions is transparent to the programmer.

Instruction Sets

- 8087 co-processor instructions are divided into six different groups:

1. Data Transfer Instructions

- Real Transfers (Example : FLD)
- Integer Transfers (Example : FILD)
- Packed Decimal Transfers (Example : FBLD, FBSTP)

2. Arithmetic Instructions

- Addition (Example : FADD, FADDP)
- Subtraction (Example : FSUB, FSUBP)
- Reversed Subtraction (Example : FSUBR, FSUBRP)
- Multiplication (Example : FMUL, FMULP)
- Division (Example : FDIV, FDIVP)
- Reversed Division (Example : FDIVR, FDIVRP)

Instruction Sets

3. Other Arithmetic Operations

(Example : FSQRT, FABS)

4. Compare Instructions

(Example : FCOM, FCOMP, FCOMPP)

5. Transcendental (Trigonometric & Exponential) Instructions

(Example : FPTAN, FPATAN)

6. Processor Control Instructions

(Example : FLD)

Instruction Sets

Floating Point Data Transfer Instructions

FLD (Load Real)

- Loads 32-bit, 64-bit or 80-bit floating-point data to Top of Stack (ST).
- Stack pointer is then decremented by 1.
- Data can be retrieved from memory, or another stack register.

Examples:

- FLD ST(2); Top of stack ← [ST(2)];

  Copies the contents of register ST(2) to top of the stack.
- FLD Memory_32; Top of stack ← [Memory_32];

  Copies the contents of memory_32 to top of the stack.

Instruction Sets

Floating Point Data Transfer Instructions

FST (Store Real)

- Stores the content of the top of the stack into memory or a specified co-processor register.
- During copy, the data rounding occurs using the rounding control bits in floating point control register.

Examples:

- FST ST(1) ; [ST(1)] ← Top of stack;

  Copies the contents of top of the stack to register ST(1).
- FST Memory_64; [Memory_64] ← Top of stack;

  Copies the contents of top of the stack to memory_64.

Instruction Sets

Floating Point Data Transfer Instructions

FSTP (Store Floating Point Number and Pop)

- Stores a copy of the top of the stack into memory or any specified co-processor register and pop the data from the top of stack.

Examples:

- FSTP ST(4) ; [ST(4)] ← Top of stack,

  Copies the contents of top of the stack to register ST(4).
- FSTP Memory_32; [Memory_32] ←Top of stack;

  Copies the contents of top of the stack to memory_32.

Instruction Sets

Floating Point Data Transfer Instructions

FXCH (Exchange)

- The FXCH instruction exchanges the contents of specified register with top of stack.

Example:

- FXCH ST(2) ; [ST(2)] ↔ Top of stack;

  Exchanges contents of top of the stack with register ST(2).

Instruction Sets

Integer Data Transfer Instructions

- Co-processor automatically converts extended floating-point number to integer data.
- FILD (Load integer)
- FIST (Store integer)
- FISTP (Store integer and pop)
- FILD ST(3) ; Top of stack ← [ST(3)];

  Copies the contents of register ST(3) to top of the stack.
- FIST ST(5) ; ST(5) ← Top of stack;

  Copies the contents of top of the stack to register ST(5).
- FST Memory_64; [Memory_64] ← Top of stack;

  Copies the contents of top of the stack to memory_64.

Instruction Sets

BCD Data Transfer Instructions

- FBLD (loads the top of stack with BCD memory data)
- FBSTP (stores top of the stack and does a pop).
- Both the instructions work in an exactly similar manner as FLD and FSTP except that the operands are BCD numbers.

Comparison Instructions

- FCOM, FCOMP, FCOMPP, FUCOM, FUCOMP, and FUCOMPP which are used to compare the two values on the top of stack and set the condition code flags appropriately.
- FCOM ST(1);

Compare ST(0) against ST(1) and set the condition code flags accordingly

Instruction Sets

Arithmetic Instructions

FADD & FADDP

- Two instructions perform real or integer addition of the specified operand with the stack top.
- After addition, the results are stored in the destination operand.
- The operand may be any of the stack registers or a memory location.
- FADD ST(0), ST(1) ; ST(0) ← ST(0) + ST(1)

  destination = destination + source.
- FADDP ST(3),ST(0) ; ST(3) ← ST(3) + ST(0)

  destination = destination + source.

Instruction Sets

Arithmetic Instructions

FSUB, FSUBP, FSUBR, FSUBRP

- The operand may be any of the stack register or memory.
- FSUB ST(0), ST(1) ; ST(0) ← ST(0)-ST(1),
  destination = destination–source.
- FSUBP ST(3),ST(0) ; ST(3) ← ST(3)-ST(0),
  destination = destination–source.
- FSUBR ST(0), ST(1) ; ST(0) ← ST(1)-ST(0),
  destination = source–destination.
- FSUBRP ST(3),ST(0) ; ST(3) ← ST(0)-ST(3),
  destination = source–destination.

Instruction Sets

Arithmetic Instructions

FMUL, FMULP

- The specified operand may be a register or a memory location.
- After multiplication, the result will be stored in the destination operand.
- FMUL ST(0), ST(1) ; ST(0) ← ST(0) × ST(1)

  destination = destination × source.
- FMULP ST(3),ST(0) ; ST(3) ← ST(3) × ST(0)

  destination = destination × source.

Instruction Sets

Arithmetic Instructions

FDIV, FIDVP, FDIVR, FDIVRP

- When the destination is not specified, the top of stack is the destination operand and source operand may be a memory operand of short real or long real type.
- If both destination and source operands are specified then compute the division and store the result in the destination.

```
FDIV ST(0), ST(1)      ; ST(0)←ST(0) / ST(1) , destination = destination / source.
FDIVP ST(3),ST(0)      ; ST(3)←ST(3) / ST(0) , destination = destination / source.

FDIVR ST(0), ST(1)     ; ST(0)←ST(1) / ST(0) , destination = source / destination.
FDIVRP ST(3),ST(0)     ; ST(3)←ST(0) / ST(3) , destination = source / destination.
```

Instruction Sets

Arithmetic Instructions

FSQRT

- The FSQRT instruction finds out the square root of the content of the top of stack and stores the result on the stack top.
- The value of the top of stack must be zero or positive otherwise FSQRT generates an invalid exception.

FABS

- The FABS instruction computes the absolute value of the content of the stack top and the result is stored in the top of stack.

Additionally qualify the arithmetic operations:

- P— Perform a register pop after the operation.
- R— Reverse mode for subtraction and division.
- I— Indicates that the memory operand is an integer. ‘I’ appears as the second letter in the instruction, such as FIADD, FISUB, FIMUL, FIDIV

Instruction Sets

Transcendental Instructions

- 8087 co-processor has eight transcendental operation instructions such as FTAN, FPTAN, F2XMI, FLY2X, FLY2XP1, FSIN, FCOS and FSINCOS.

✓ ***FPTAN*** The FPTAN instruction is used to compute the partial tangent of an angle θ, where θ must be in the range $0° \leq \theta \leq 90°$. The value of angle must be stored at the stack top. The result is computed in the form of a ratio of ST/ST(1).

✓ ***FPATAN*** The FPATAN instruction calculates the are tangent (inverse tangent) of a ratio ST(1)/ST(0). The stack is popped and the result is stored on the top of stack. Its function can be expressed as

$$ST(0) = \tan^{-1}\left(\frac{ST(1)}{ST(0)}\right)$$

✓ ***FYL2X*** This instruction is used to calculate $\log_2 X$ where X must be in the range of $0 \leq X \leq \infty$ and Y must be in the range $0 \leq Y \leq \infty$.

✓ ***FYL2XP1*** This instruction is used to compute the function $\log_2 (X + Y)$. This instruction is almost identical to FYL2X except that it gives more accurate results when computation log of a number very close to one.

Instruction Sets

Co-processor Control Instructions

- The co-processor control instructions are used to program the numeric processor or to handle the internal functions like flags manipulations, exception handling, processor environment maintenance and preparation, etc.

- The 8087 coprocessor control instructions are FINIT, FENI, FDISI, FLDCW, FSTCW, FSTSW, FCLEX, FINCSTP, FDECSTP, FFREE, FNOP, FWAIT, FSTENV, FLDENV, FRSTOR and FSAVE.

Instruction Sets

Co-processor Control Instructions

- FINIT: initializes the 8087 for further execution. This instruction must be executed and the hardware will be reset before executing FPU instructions.
- FENI: Enables the interrupt structure and response mechanism of 8087. The interrupt mask flag is cleared.
- FDISI: Sets the interrupt mask flag to disable the interrupt response mechanism of 8087.
- FWAIT: Causes the microprocessor to wait for the coprocessor to finish an operation.