CPE/EE 421/521
Fall 2004
Chapter 6 – Exception Handling and the 68000

Dr. Rhonda Kay Gaede

UAH
Exception Handling and the 68000 – Overview

• Interrupts and exceptions are events that alter the normal execution of a program
  – Exception –
  – Interrupt –

• Exception examples
  – _____________________
  – _____________________
  – _____________________
  – _____________________

• Interrupt examples
  – _____________________
  – _____________________

• Each type of exception has its own ____________________
6.1 Interrupts – Interrupt Processing Mechanism

- Interrupt is an __________________ event
- When an interrupt occur, the computer can:
  - ________________
  - ________________
6.1 Interrupts – Interrupt Service Actions

1. The computer completes its current machine-level instruction
2. The contents of PC is saved (on stack)
3. The state of the processor (status word) is saved on the stack
4. Jump to the location of the interrupt handling routine
6.1 Interrupts –
Interrupt Processing Facts

• The interrupt is ______________ to the interrupted program

• Interrupt request:
  – Can be ________ or __________
  – When it is deferred, it is said to be masked
  – Special one: nonmaskable interrupt request (NMI)
  – The 68000 NMI: IRQ7 (MSP430: RST*/NMI pin)

• Prioritized interrupts

• Vectored interrupts
  – Requesting peripheral identifies itself, CPU doesn’t have to poll the status of each device to discover the interrupter
6.1 Interrupts –
68K Interrupt Interface: Schematic

- A peripheral asserts output IRQ* that is connected to one of IRQ1* to IRQ7*, which are, in turn, input to a ______________
- The priority level of the interrupt is compared to the level of interrupts currently being accepted ______________
- When the 68000 decides to service an interrupt, ___ is output on FC2-FC0 and the ______________ is output on A3-A1
- FCs and As generate an ______ using a decoder
- When the peripheral receives an IACK, the device identifies itself by sending an ______________ to the CPU
6.1 Interrupts - 68K Interrupt Interface: Interrupt Encoding Details

- A level 7 interrupt CAN interrupt level 7 interrupt
- Reset, bus error, address error, and trace exceptions take precedence over an interrupt

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Level</td>
<td>IRQ1*</td>
</tr>
<tr>
<td>7</td>
<td>X</td>
</tr>
<tr>
<td>6</td>
<td>X</td>
</tr>
<tr>
<td>5</td>
<td>X</td>
</tr>
<tr>
<td>4</td>
<td>X</td>
</tr>
<tr>
<td>3</td>
<td>X</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Note: 0 = low level, 1 = high level, X = don’t care
6.1 Interrupts – Processing the Interrupts: IACK* Sequence

1. Place interrupt level on A21, A20, and A19. All other address lines high
2. Set R/W* to read
3. Set FC to interrupt acknowledge (1,1,1)
4. Assert AS*
5. Assert lower data strobe LDS* (UDS* is also asserted)

Provide interrupt vector number
1. Place vector number on D00–D07
2. Assert DTACK*

Acquire vector number
1. Latch vector number
2. Negate LDS* (and UDS*)
3. Negate AS*

Release bus
Negate DTACK*

Interrupt processing
6.1 Interrupts - Processing the Interrupts: Interrupt Timing Diagram

- CLK
- A_{04}-A_{23}
- A_{01}-A_{03}
- AS*
- UDS*
- LDS*
- R/W*
- DTACK*
- D_{08}-D_{15}
- D_{00}-D_{07}
- FC0-2
- IPL0*-IPL2*

Last bus cycle of instruction (read or write)  Stack PLC (SSP)  IACK cycle (vector number acquisition)  Stack and vector fetch
6.1 Interrupts – Vectored Interrupts
6.1 Interrupts – Exception Vectors

• A vector is associated with each type of exception
  – Vector is the 32-bit absolute address
    of the appropriate ________________________

• 256 exception vectors, 32 bits (4 bytes) each,
  extending from address $00 0000 to $__ ______

• Vectors 0-63 : EXCEPTIONS

• Vectors 64-255 : INTERRUPT HANDLING ROUTINES

• Difference between the reset vector and all other exceptions:
  – It requires 2 longwords
  – Located in SP space (FC = 110);
    others are in SD space (FC = 101)
6.1 Interrupts – Reset Facts

• When the RESET* pin is asserted for the appropriate duration:
  – SR = $2700
  – SSP is loaded with the longword @ $00 0000
  – PC is loaded with the longword @ $00 0004
6.2 Privileged States and the 68000 – The Status Register

S
When set, the supervisor-state bit indicates that the 68000 is in its supervisor state. When clear, S indicates that the 68000 is in its user state.

T
When the trace bit is clear, the 68000 operates normally. When T = 1, the 68000 generates a trace exception after the execution of each instruction. The trace exception is used to debug programs.

I₂, I₁, I₀
The interrupt mask bits, I₂, I₁, and I₀ indicate the level of the current interrupt mask (i.e., 0 to 7).

Diagram:

Status byte

<table>
<thead>
<tr>
<th>15</th>
<th>14</th>
<th>13</th>
<th>12</th>
<th>11</th>
<th>10</th>
<th>9</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>S</td>
<td>I₂</td>
<td>I₁</td>
<td>I₀</td>
<td>X</td>
<td>N</td>
<td>Z</td>
</tr>
</tbody>
</table>

CCR

<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>N</td>
<td>Z</td>
<td>V</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Interrupt mask bits
Supervisor state bit
Trace bit
6.2 Privileged States and the 68000 – User/Supervisor Mode Transitions

An exception always forces the 68000 into the supervisor state.
6.3 Exception Processing

• Phase 1 – The 68000 makes a temporary internal copy of the pre-exception status register, turns on ________________ and disables ____________.

• Phase 2 – The vector number corresponding to the current exception is determined, then multiplied by 4 to calculate the value of a pointer to the ________________ routine.

• Phase 3 – The current CPU ______ is saved on the stack pointed to by the Supervisor Stack Pointer in a data structure called an ________________.
  – For _____, ______________, and ____________, this context consists of the program counter, the status register, the data access address and memory access type and function code
  – For all other exceptions, this context consists of only the _______ __________ and the ________________

• Phase 4 – Load the ______________ with the address of the first instruction of the exception handling routine.
6.4 Exceptions Implemented by the 68K

- All exceptions
  - Hardware (external)
    - RESET
    - Interrupts
    - Bus Error
  - Software (internal)
    - TRACE
    - Errors
      - Programmer initiated
      - Coprocessor
      - Address error
      - Privilege violation
      - Illegal instruction
      - DIVS, DIVU
      - CHK
      - TRAPV
      - TRAP #n
      - Emulator
6.5 Interrupts and Real-Time Processing

- Multitasking (multiprogramming)
  - concurrent execution
  - multiple tasks (processes)
  - resource sharing (multiple users using the same printer)

- Multiprocessing
  - parallel execution
  - multiple PROCESSORS!
6.5 Interrupts and Real-Time Processing - Multitasking

- ________________ (to schedule activities)
- ________________ (to switch between tasks)

<table>
<thead>
<tr>
<th>Resource</th>
<th>Activity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Slot 1</td>
</tr>
<tr>
<td>VDT1</td>
<td>Task A</td>
</tr>
<tr>
<td>VDT2</td>
<td></td>
</tr>
<tr>
<td>Disk</td>
<td></td>
</tr>
<tr>
<td>CPU</td>
<td>Task A</td>
</tr>
</tbody>
</table>
6.5 Interrupts and Real-Time Processing - Real-Time Operating System

- **Real time - meaningful time**
  - fast enough to influence the system at that moment
    - space shuttle / chemical plant

- **Real-time system**
  - Optimizes the response time to events
  - Tries to use resources efficiently

- **Multitasking system**
  - Optimizes resource utilization
  - Tries to provide a reasonable response time
6.5 Interrupts and Real-Time Processing - Real-Time Kernel

- Scheduler is the *kernel*, *nucleus*, of a real-time OS
- Functions
  - a first-level interrupt handler
  - scheduler - the sequence in which tasks are executed
  - interprocess communication
- Task States
  - Ready
  - Running
  - Blocked (dormant)
6.5 Interrupts and Real-Time Processing - Tasks

- Volatile portion (PC, status, registers)
- Task control block (TCB)
  - Task ID
  - Task block pointer
    - PC
    - SP
    - status register
    - other registers
  - Task status
    - run / ready / blckd
  - Task priority
  - Task time allocation
    - how many slots
6.5 Interrupts and Real-Time Processing - Exception Handling and Tasks

- Preemptive real-time OS:
  - RTC generates periodic interrupts
  - used by the kernel to locate and run the next task
- How to deal with other interrupts?
  - Service them independently, subject to priority
  - Integrate them into the real-time task structure

Diagram:

- Vectored interrupt
  - Turn on ready to run flag of appropriate task
- Exception
  - Turn on ready to run flag of appropriate task
- Real-time clock interrupt
  - Call RT task scheduler and switch tasks
- RTE
  - Turn on ready to run flag of appropriate task
  - Call RT task scheduler and switch tasks