THE 68000 CPU HARDWARE MODEL  
Chapter 4

- 68000 interface
- Timing diagram
- Minimal configuration using the 68000

68000 Interface

- M68000: 64 pins, arranged in 9 groups:
  - Address Bus: $A_{23}$ – $A_{01}$
  - Data Bus: $D_{15}$ – $D_{00}$
  - Asynchronous bus control: $AS^*$, $R/W^*$, $UDS^*$, $LDS^*$, $DTACK^*$, $BERR^*$
  - Synchronous bus control: $E$, $VPA^*$, $VMA^*$
  - Bus arbitration control: $BR^*$, $BG^*$, $BGACK^*$
  - Function code: $FC0$, $FC1$, $FC2$
  - System control: $CLK$, $RESET^*$, $HALT^*$
  - Interrupt control: $IPL0^*$, $IPL1^*$, $IPL2^*$
  - Miscellaneous: $Vcc$(2), $Gnd$(2)

- Legend: Type
  - XX Input
  - XX Output
  - XX Input/Output

68000 Interface, cont’d

- Classification of pins based on function
  - SYSTEM SUPPORT PINS
    - Essential in every 68000 system (power supply, clock, …)
  - MEMORY AND PERIPHERAL INTERFACE PINS
    - Connect the processor to an external memory subsystem
  - SPECIAL-PURPOSE PINS
    - (not needed in a minimal application of the processor)
    - Provide functions beyond basic system functions

- Terminology
  - Asterisk following a name: indicates the signal is active low
  - "Signal is asserted" means
    - signal is placed in its active state
  - "Signal is negated" means
    - signal is placed in its inactive state

System Support Pins

- Power Supply
  - Single +5V power supply: 2 Vcc pins and 2 ground pins
- Clock
  - Single-phase, TTL-compatible signal
- Bus cycle: memory access, consists of a minimum 4 clock cycles
- Instruction: consists of one or more bus cycles
- $RESET^*$
  - Forces the 68000 into a known state on the initial application of power:
    - supervisor’s A7 is loaded from memory location $50 0000$
    - Program counter is loaded from address $50 0004$
    - During power-up sequence must be asserted together with the $HALT^*$ input for at least 100 ms.
    - Acts also as an output, when processor executes the instruction $RESET^*$ (used to reset peripherals w/out resetting the 68000)

Course Administration

- Instructor: Aleksandar Milenkovic
  - milenka@ece.uah.edu
  - www.ece.uah.edu/~milenka
- TA: Joel Wilder
- Labs: Lab #5 is on. Hw2 due is 11/02/05.
- Test II: 11/09/05 (MSP430 & related).
- Text: Microprocessor Systems Design: 68000 Hardware, Software, and Interfacing
- Review: M68K (Chapter 1; Chapter 2; Chapter 3), MSP430 (Introduction, Arch., Basic Clock System, WDT, Low Power Modes, Digital I/O), Timers, MSP 430 UART;
- Today: M68000 Hw (Chapter 4)
System Support Pins, cont’d

- **HALT**
  - In simple 68000 systems can be connected together with **RESET**.
  - Can be used:
    - by external devices to make the 68000 stop execution after current bus cycle (and to negate all control signals)
    - to single-step (bus cycle by bus cycle) through program
    - to rerun a failed bus cycle (if memory fails to respond correctly)
  - In conjunction with the bus error pin, **BERR**
  - It can be used as an output, to indicate that the 68000 found itself in situation from which it cannot recover (HALT is asserted)

Memory and Peripheral Interface Pins

- **Address Bus**
  - 23-bit address bus, permits $2^{23}$ 16-bit words to be addressed
  - Tri-state output pins (to permit devices other than the CPU to take control over it)
  - Auxiliary function:
    - supports vectored interrupts
    - Address lines $A_{20}$, $A_{21}$, $A_{22}$ indicate the level of the interrupt being serviced
    - All other address lines are set to a high level

- **Data Bus**
  - Bi-directional 16-bit wide data bus
    - During a CPU read cycle acts as an input
    - During a CPU write cycle acts as an output
    - Byte operations: only $D_{00}$-$D_{07}$ or $D_{08}$-$D_{15}$ are active
    - Interrupting device identifies itself to the CPU by placing an interrupt vector number on $D_{00}$-$D_{07}$ during an interrupt acknowledge cycle

Memory and Peripheral Interface Pins, cont’d

- **AS**
  - When asserted, indicates that the content of the address bus is valid.

- **R/W**
  - Determines the type of a memory access cycle
    - CPU is reading from memory: $R/W = 1$
    - CPU is writing to memory: $R/W = 0$
    - If CPU is performing internal operation, $R/W$ is always 1
    - When CPU relinquishes control of its busses, $R/W$ is undefined

- **UDS** and **LDS**
  - Used to determine the size of the data being accessed
  - If both UDS and LDS are asserted, word is accessed
  - $R/W$ UDS LDS
    - 010: write lower byte ($D_{00}$-$D_{07}$: data valid, replicated on $D_{8}$-$D_{15}$)
    - 011: write word ($D_{00}$-$D_{15}$: data valid)
    - 101: read upper byte ($D_{00}$-$D_{07}$: invalid, $D_{8}$-$D_{15}$: data valid)

Memory and Peripheral Interface Pins, cont’d

- **DTACK** (Data Transfer Acknowledge)
  - Handshake signal generated by the device being accessed
  - Indicates that the contents of the data bus is valid
  - If DTACK is not asserted, CPU generates wait-states until DTACK goes low or until an error state is declared.
  - When DTACK is asserted, CPU completes the current access and begins the next cycle

  DTACK has to be generated a certain time after the beginning of a valid memory access (timer supplied by the system designer).

Special-Function Pins of the 68000

- **BERR** (Bus Error Control)
  - Enables the 68000 to recover from errors within the memory system

- **BR**, **BG**, **BGACK** (Bus Arbitration Control)
  - Used to implement multiprocessor systems based on M68000

- **FC0-FC2** (Function Code Output)
  - Indicate the type of cycle currently being executed
  - Becomes valid approximately half a clock cycle earlier than the contents of the address bus

- **IPL0*-IPL2** (Interrupt Control Interface)
  - Used by an external device to indicate that it requires service
  - 3-bit code specifies one of eight levels of interrupt request

Figure 4.3
Special-Function Pins of the 68000, cont'd

<table>
<thead>
<tr>
<th>Function Code Output</th>
<th>Processor Cycle Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>FC2 FC1 FC0</td>
<td></td>
</tr>
<tr>
<td>0 0 0</td>
<td>Undefined, reserved</td>
</tr>
<tr>
<td>0 0 1</td>
<td>User data</td>
</tr>
<tr>
<td>0 1 0</td>
<td>User program</td>
</tr>
<tr>
<td>0 1 1</td>
<td>Undefined, reserved</td>
</tr>
<tr>
<td>1 0 0</td>
<td>Undefined, reserved</td>
</tr>
<tr>
<td>1 0 1</td>
<td>Supervisor data</td>
</tr>
<tr>
<td>1 1 0</td>
<td>Supervisor program</td>
</tr>
<tr>
<td>1 1 1</td>
<td>CPU space (interrupt acknowledge)</td>
</tr>
</tbody>
</table>

Special-Function Pins of the 68000, cont'd

Asynchronous Bus Control

The 68000 is not fully asynchronous because its actions are synchronized with a clock input. It can prolong a memory access until an ACK is received, but it has to be in increments of one clock cycle.

Interpreting the Timing Diagram
The Clock

- A microprocessor requires a clock that provides a stream of timing pulses to control its internal operations.
- A 68000 memory access takes a minimum of eight clock states numbered from clock state S0 to clock state S7.

A memory access begins in clock state S0 and ends in state S7.

The most important parameter of the clock is the duration of a cycle, t_{CYC}.

At the start of a memory access, the CPU sends the address of the location it wishes to read to the memory.

Address Timing

- We are interested in when the 68000 generates a new address for use in the current memory access.
- The next slide shows the relationship between the new address and the state of the 68000’s clock.

Initially, in state S0 the address bus contains the old address. In state S1 a new address becomes valid for the remainder of the memory access.
The time at which the contents of the address bus change can be related to the edges of the clock.

Address Timing
- Let’s look at the sequence of events that govern the timing of the address bus
- The “old” address is removed in state S0
- The address bus is floated for a short time, and the CPU puts out a new address in state S1

The old address is removed in clock state S0 and the address bus floated.

The designer is interested in the point at which the address first becomes valid. This point is $t_{CLAV}$ seconds after the falling edge of S0.

The memory needs to know when the address from the CPU is valid. An address strobe, $A^*$, is asserted to indicate that the address is valid.

Address and Address Strobe
- We are interested in the relationship between the time at which the address is valid and the time at which the address strobe, $A^*$, is asserted
- When $A^*$ is active-low it indicates that the address is valid
- We now look at the timing of the clock, the address, and the address strobe
The Data Strobes

- The 68000 has two data strobes LDS* and UDS*. These select the lower byte or the upper byte of a word during a memory access.
- To keep things simple, we will use a single data strobe, DS*.
- The timing of DS* in a read cycle is the same as the address strobe, AS*.

The Data Bus

- During a read cycle the memory provides the CPU with data.
- The next slide shows the data bus and the timing of the data signal.
- Note that valid data does not appear on the data bus until near the end of the read cycle.

Data from the memory appears near the end of the read cycle.
Analyzing the Timing Diagram

- We are going to redraw the timing diagram to remove clutter.
- We aren't interested in the signal paths themselves, only in the relationship between the signals.

We are interested in the relationship between the clock, AS*/DS*, and the data in a read cycle.

The earliest time at which the memory can begin to access data is measured from the point at which the address is first valid.

Calculating the Access Time

- We need to calculate the memory’s access time.
- By knowing the access time, we can use the appropriate memory component.
- Equally, if we select a given memory component, we can calculate whether its access time is adequate for a particular system.

Data from the memory is latched into the 68000 by the falling edge of the clock in state S6.
Data must be valid \( t_{\text{DICL}} \) seconds before the falling edge of \( S_6 \) We know that the time between the address valid and data valid is \( t_{\text{free}} \)

The address becomes valid \( t_{\text{CLAV}} \) seconds after the falling edge of \( S_0 \)

From the falling edge of \( S_0 \) to the falling edge of \( S_6 \):
- the address becomes valid
- the data is accessed
- the data is captured

The falling edge of \( S_0 \) to the falling edge of \( S_6 \) is three clock cycles

\[ 3t_{\text{cyc}} = t_{\text{CLAV}} + t_{\text{acc}} + t_{\text{DICL}} \]
Timing Example

- 68000 clock 8 MHz  \( t_{\text{CYC}} = 125 \text{ ns} \)
- 68000 CPU  \( t_{\text{CLAV}} = 70 \text{ ns} \)
- 68000 CPU  \( t_{\text{DCL}} = 15 \text{ ns} \)
- What is the minimum \( t_{\text{ACC}} \)?
  \[ 3 \cdot t_{\text{CYC}} = t_{\text{CLAV}} + t_{\text{ACC}} + t_{\text{DCL}} \]
  \[ 375 = 70 + t_{\text{ACC}} + 15 \]
  \[ t_{\text{ACC}} = 290 \text{ ns} \]

Extended Read Cycle

- DTACK* did not go low at least 20 ns before the falling edge of state S4
- Designer has to provide logic to control DTACK*

Memory Timing Diagram

- The 6116 static memory component
  - 2K x 16 bit memory – byte-oriented!
  - Two 6116’s configured in parallel to allow word accesses
  - Eleven address inputs

Memory Timing Diagram, cont’d

- Assumptions:
  - R/W* is high for the duration of the read cycle
  - OE* is low
68000 Write Cycle

- 68000 transmits a byte or a word to memory or a peripheral
- Essential differences:
  - The CPU provides data at the beginning of a write cycle
  - One of the bus slaves (see later) reads the data
- In a read cycle DS* and AS* were asserted concurrently. **This will be not a case here!**
- Reason for that: 68000 asserts DS* only when the contents of data bus have stabilized
  - Therefore, memory can use UDS*/LDS* to latch data from the CPU.

Timing Example

- 68000 clock 8 MHz
- 68000 CPU
- tCLAV = 70 ns
- 68000 CPU
- tDICL = 15 ns
- What is the minimum tACC?
  - \(3 \times t_{CYC} > t_{CLAV} + t_{ACC} + t_{DICL}\)
  - \(375 > 70 + t_{ACC} + 15\)
  - \(t_{ACC} < 290\) ns (or tAA from the timing diagram, access time)

- For the 12.5 MHz version of 68000
  - tCLAV = 55 ns
  - tDICL = 10 ns
  - \(3 \times 80 > 55 + t_{ACC} + 10\)
  - \(t_{ACC} < 175\) ns

Remember, maximum tAA for the 6116 RAM was 200 ns

Simplified write cycle timing diagram

- In a write cycle UD5*/LD5* is asserted one cycle after AS*.

Write Cycle Timing Diagram of a 6116 RAM

- Address setup time (min 20 ns)
- Write pulse width (min 90 ns)
- Write recovery time (min 10 ns)
- Address valid to end of write (min 120 ns)
Write cycle ends with either CS* or WE* being negated (CS* and WE* internally combined)

- An address must be valid for at least $t_{AS}$ nanoseconds before WE* is asserted
- Must remain valid for at least $t_{WR}$ nanoseconds after WE* is negated
- Data from the CPU must be valid for at least $t_{DW}$ nanoseconds before WE* is negated
- Must remain valid for at least $t_{DH}$ nanoseconds after the end of the cycle

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Symbol</th>
<th>Minimum</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Write cycle time</td>
<td>$t_{WT}$</td>
<td>90</td>
<td>150</td>
</tr>
<tr>
<td>Chip select time to end of write</td>
<td>$t_{W}$</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>Write recovery time</td>
<td>$t_{WR}$</td>
<td>120</td>
<td></td>
</tr>
<tr>
<td>Address valid to end of write</td>
<td>$t_{AS}$</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>Address setup time</td>
<td>$t_{AR}$</td>
<td>50</td>
<td></td>
</tr>
<tr>
<td>Write pulse width</td>
<td>$t_{WP}$</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Data setup time</td>
<td>$t_{DS}$</td>
<td>10</td>
<td></td>
</tr>
</tbody>
</table>