THE 68000 CPU HARDWARE MODEL
Chapter 4

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

Review: 68000 Interface
- M68000: 64 pins, arranged in 9 groups:
  - Address Bus: \( A_{31} \sim A_{18} \)
  - Data Bus: \( D_{15} \sim D_0 \)
  - 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:
  - \( XX \) Input
  - \( XX \) Output
  - \( XX \) Input/Output

Review: Special-Function Pins of the 68000

<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>
Review: Special-Function Pins of the 68000, cont'd

Using the 68000’s function code outputs

Figure 4.8

Review: 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.

Figure 4.11

Review: Bus Read Cycle

3 $t_{CE} = t_{CLAV} + t_{wex} + t_{DCIL}$

Figure 4.14

Extended Read Cycle

- The 6116 static memory component
  - 2K x 8bit memory - byte-oriented!
  - Two 6116’s configured in parallel to allow word accesses
  - Eleven address inputs

Figure 4.15

Memory Timing Diagram
Assumptions:
- R/W* is high for the duration of the read cycle
- OE* is low

Timing Example
- 68000 clock 8 MHz
  - t_{DCL} = 125 ns
  - t_{CLAV} = 70 ns
  - t_{DICL} = 15 ns

For the 12.5MHz version of 68000
- t_{DCL} = 80 ns
- t_{CLAV} = 55 ns
- t_{DICL} = 10 ns
- t_{DCL} < 175 ns

Reason for this: 68000 asserts DS* only when the contents of data bus have stabilized
- Therefore, memory can use UDS*/LDS* to latch data from the CPU

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
Follow this sequence of events in a write cycle:

- Address stable
- AS* asserted
- R/W* brought low
- Data valid
- DS* asserted

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

Designing a Memory Subsystem, an example

- Design a M68000 memory subsystem using
  - Two 32K x 8 RAM chips residing at address $500000$
  - Two 8K x 8 RAM chips residing in the consecutive window
  - LS 138 (3 to 8 decoder) and basic logic gates

- Solution
  - 32K is 4 x 8
    - Let’s split the address space into 8K modules
  - In total, we have five (4+1) 8K windows
  - To address each line in 8K window
    - $2^4 = 16$ bits (2 bits = 1 bit)
  - To address five modules we need 3 bits
  - Don’t forget that there is no $A_0$, we will use LDS/UDS

Designing a Memory Subsystem, an example

- Two 32K x 8 RAM chips residing at address $500000$
- Two 8K x 8 RAM chips residing in the consecutive window
- LS 138 (3 to 8 decoder) and basic logic gates

- Solution
  - 32K is 4 x 8
    - Let’s split the address space into 8K modules
  - In total, we have five (4+1) 8K windows
  - To address each line in 8K window
    - $2^4 = 16$ bits (2 bits = 1 bit)
  - To address five modules we need 3 bits
  - Don’t forget that there is no $A_0$, we will use LDS/UDS
Interrupt Control Interface (details later)

<table>
<thead>
<tr>
<th>Address bus</th>
<th>Data bus</th>
<th>Control bus</th>
<th>Arbitration bus</th>
</tr>
</thead>
<tbody>
<tr>
<td>Slave module</td>
<td>Master module</td>
<td>Master module</td>
<td>Memory</td>
</tr>
<tr>
<td>Memory</td>
<td>CPU</td>
<td>Local memory</td>
<td>HD</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>I/O</td>
</tr>
</tbody>
</table>

Bus Arbitration Control
- When 68000 controls the address and data buses, we call it the bus master
- The 68000 may allow another 68000 or DMA controller to take control over buses
- In the system with only one bus master, 68000 would have permanent control of the address and data buses

68000 must respond to BR* request (it cannot be masked)
- Assertion of BG* indicates that the bus will be given up at the end of present bus cycle
- Requesting device waits until AS*, DTACK*, and BGACK* have been negated, and only then asserts its own BGACK* output
- Old master negates its BG*, and BR* can be asserted by another potential master

Data Bus Contention in Microcomputers
- Situation where more than one device attempts to drive the bus simultaneously
- Example: Two memory modules, M1 selected during read cycle 1, M2 selected during read cycle 2
- Assumption:
  - M1 has data bus drivers with relatively long turn-off times
  - M2 has data bus drivers with relatively short turn-on times

Data Bus Contention in Microcomputers, cont’d

Bus Contention and Data Bus Transceivers
- Data bus transceiver – consists of a transmitter (driver) and a receiver
- Driver – tristate output, can be driven high, low, or internally disconnected form the rest of the circuit
- Two control inputs: **Enable** (active low) and **DIR** (direction)
- Dynamic data bus contention

Bus Arbitration Control, cont’d
Minimal Configuration Using The 68000

DESIGN CONSTRAINTS
- Used in stand-alone mode
- Classroom teaching aid
- 16 KB EPROM-based monitor
- Speed is not important
- At least 4 KB RAM
- 1 serial and 1 parallel port
- Memory expandable
- No interrupts and multiple processors

MAJOR COMPONENTS
- ROM – Two 8K × 8 components
- RAM – Two 2K × 8 components
- Parallel – 6821 Peripheral Interface Adapter (PIA)
- Serial – 6850 Asynchronous Comm. Interface Adapter (ACIA)

DESIGN CHOICES
- Chose the location of ROM (16KB) and RAM (8 KB) within the address space (16 MB)
  - Unimportant, as long as the reset vectors are located at $00 0000$
- Chose the location of memory-mapped peripherals
- Control of DTACK* (is delay applied or not?)

The 68000's Reset Sequence

RESET SEQUENCE
- Set SR S bit to 1
- Set SR T bit to 0
- Set SR mode to 11
- Fetch SSP from address 0
- Transfer longword to SSP
- Fetch initial PC from address 4
- Transfer longword to PC
- Begin processing in the supervisor state

REMEMBER
- 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 @ $00 0004$
Memory and Peripheral Components

- We assigned address lines to address pins, and data lines to data pins.
- Before designing logic that will generate chip select signals, we have to decide about RAM/ROM location.
- To assure that the reset vector location is at $00\ 0000$, let’s situate $16\ KB$ of ROM at $00\ 0000$.

Control Section

- We will divide the memory space $00\ 0000$ - $01\ FFFF$ into eight blocks of $16\ KB$ (IC1a, b, IC2a, IC3)
- $16\ KBytes$ of ROM are at $00\ 0000$ to $00\ 3FFF$
- Where is the RAM situated? Peripherals?
- Note: there is no delay applied to DTACK*.
- What will happen if we access non-decoded memory?

Different approaches to memory arrangement

- Largest memory window (16 KB) [MEMORY GAPS]
Different approaches to memory arrangement, cont’d

- Smallest memory window (4 KB) [NO MEMORY GAPS]

![Memory Diagram]

ROM (16 KB)

4 Windows (Blocks)

RAM

- ROM is EPROM-based, and thus slower
- With EPROMs from the same generation, we’ll need wait states, maybe even with RAM components
- Watchdog for non-decoded memory addresses

How can we make it better?

- CONTROL OF INTERRUPTS
  ❖ Use 74LS148 priority encoder to provide 7 levels of interrupt
- EXTERNAL BUS INTERFACE
  ❖ CPU can supply only the limited current to drive the bus
  ❖ SOLUTION: Bus drivers (buffers)

DTACK* Generation

- DTACK* generator based on a shift register

![DTACK* Generation Diagram]

DTACK* Generation

Shift register and its timing diagram

![DTACK* Generation Diagram]
DTACK* Generation
Shift register and its timing diagram

DTACK* Generation
DTACK* generator based on a counter

Figure 4.74