A HIGH PERFORMANCE MC68000L12 SYSTEM WITH NO WAIT STATES

Prepared by
Trey West
Microprocessor Systems Engineer
Austin, Texas

Demands on microprocessor throughput are constantly increasing. For instance, complex graphic systems require rapid data manipulation for high speed displays. Likewise, front-end processors must rapidly transfer tremendous amounts of data via synchronous data links. The list of possible performance oriented applications is ever increasing. High performance MC68000 microprocessor systems are capable of meeting these throughput demands provided they are designed to operate without wait states.

System throughput can be increased by several methods, the simplest of which is to increase the system clock frequency. However, this method yields higher performance only if the remainder of the system is capable of operating at the higher rate. Multiprocessing increases throughput, but when used in a small system the overhead required might be prohibitive. Software optimization would certainly provide faster program execution; however, this is perhaps the most difficult and time consuming method of throughput enhancement.

This application note describes throughput enhancement by using increased clock frequency and system response. Although the design of this system is optimized for a 12.5 MHz processor clock, it is also applicable for higher speed MC68000 MPUs.

TIMING CONSIDERATIONS

Due to the asynchronous bus structure of the MC68000, simply increasing the clock frequency is not sufficient for increased throughput. Memory and peripheral devices must be able to respond to the accelerated transfer rate to avoid insertion of wait states. Speeding up the clock rate without decreasing access times will generally cause the MC68000 to "wait faster." However, system thoughput would be increased for operations not requiring bus transfers; e.g., multiply and divide.

To achieve the highest possible performance from an MC68000 system at any given clock speed, all transfers must occur without wait states. The read and write cycle timing diagrams, shown in the MC68000 data sheet, indicate that minimum read and write cycles occur within the first four clock cycles from state zero (S0). To add wait states to those transfers, it is only necessary to delay the assertion of $\overline{\text{DTACK}}$ beyond the falling edge of S4 (state 4 of the clock). Then, if $\overline{\text{DTACK}}$ is asserted prior to the falling edge of S6 (state 6 of the clock), one wait cycle (two wait states) will be inserted. The actual number of wait states inserted depends upon when $\overline{\text{DTACK}}$ is asserted; e.g., if $\overline{\text{DTACK}}$ is asserted a setup time prior to the falling edge of S4, no wait state is added, whereas, if the assertion of $\overline{\text{DTACK}}$ is delayed indefinitely, an infinite number of wait states are added. Thus, to achieve no wait state operation, $\overline{\text{DTACK}}$ must be asserted a setup time prior to the falling edge of S4. The easiest method available for meeting this criteria is to allow the select line of the peripheral to assert $\overline{\text{DTACK}}$. Thus, as soon as a valid address decode has taken place (some gate delay after the assertion of $\overline{\text{AS}}$, $\overline{\text{UDS}}$, and/or $\overline{\text{LDS}}$), $\overline{\text{DTACK}}$ will be asserted. Generating $\overline{\text{DTACK}}$ in this manner assures no wait state bus cycles.

Normally, $\overline{\text{UDS}}$ and/or $\overline{\text{LDS}}$ are included in the decoding logic for peripheral (and memory) chip select signal generation. Since both $\overline{\text{UDS}}$ and $\overline{\text{LDS}}$ are active only for a portion of the bus cycle (two clock cycles for a read and one for a write), they determine the actual access time allotted for a peripheral or memory device. For example, in a 12.5 MHz MC68000 system, the static RAM access time required for "no wait state" transfer is less than 80 nanoseconds (for a write). All signals (A1-A23, R/W, D0-D15, etc.) needed to perform a write are valid the entire time $\overline{\text{UDS}}$ and/or $\overline{\text{LDS}}$ are active; therefore, the entire 80 nanoseconds is available for access time.

NOTE

Dynamic RAMs could include $\overline{LDS}/\overline{UDS}$ in the CAS driver logic ($\overline{AS}$ and addresses would cause $\overline{RAS}$) increasing the available access time to 130 nanoseconds for a 12.5 MHz MC68000.

Connecting the M68000 Family peripherals to a "no wait state" system is relatively simple. To attain maximum speed data transfers, use similar M68000 Family clock speed devices in the system; e.g., use an MC68230L12 parallel interface/timer with an MC68000L12 MPU. The E output of the MC68000 is one-tenth of the system clock frequency and provides synchronous operation for M6800 Family peripherals. Synchronous M6800 Family peripherals can also be used asynchronously with the MC68000. In this mode, MC68BXX peripherals are recommended because of decreased access times.

HARDWARE CONSIDERATIONS

Designing an economical "no wait state" system requires careful consideration of the timing constraints. By careful selection of devices with the optimum decode logic speed and memory access times, the most cost-effective solution can be implemented. For the system design discussed in this application note, high-speed MCM2147 HMOS static RAMs were selected. These RAMs are avilable from Motorola with access times ranging from 35 to 100 nanoseconds.

Due to the 80 nanosecond access constraint mentioned previously, MCM2147 RAMs with a 70 nanosecond access time were chosen. Refer to the hardware circuitry shown in the schematic diagram of Figure 1. For a write cycle, address decoding for chip select (E, CS) must be accomplished during the 80 nanoseconds between assertion of $\overline{AS}$ and assertion of $\overline{LDS}$ and/or $\overline{UDS}$. Four levels of logic are required before the RAM SELECT line is asserted at the LDS/UDS OR gates (U40). When S-TTL logic is used for these four levels of logic, the delay time is a maximum of 28 nanoseconds; however, if LS-TTL logic is used, the delay time is a maximum of 88 nanoseconds. In order to economically accomplish this decode function, it is obvious that some combination of S and LS TTL could be used. (Some of the newer high-speed CMOS devices [74HCXX] could also be used with a considerable reduction in power requirements; however, a pullup resistor [typically 10 k] must be used to guarantee input level requirements for CMOS devices.) The schematic diagram of Figure 1 uses all S-TTL devices, for accomplishing the decode function, to allow operation beyond 12.5 MHz. If the $\overline{RAM\ SELECT}$ decoding is accomplished properly, this signal will be asserted (low) before the assertion of $\overline{LDS}/\overline{UDS}$. When $\overline{UDS}$ and/or $\overline{LDS}$ are asserted, one gate delay remains before the $\overline{RAMH}$ or $\overline{RAML}$ chip selects (MCM2147 E inputs) are asserted. Since S-TTL OR gates switch state in a maximum of seven nanoseconds, this leaves approximately 70 nanoseconds for access to RAM contents. Note that an MCM2147 static RAM latches data when its write ($\overline{W}$) line goes to the read state (logic high). Since MC68000 data is no longer valid when the $R/\overline{W}$ line goes high, $R/\overline{W}$ should be ORed with $\overline{AS}$. The output of this OR gate is then connected to the MCM2147 $\overline{W}$ lines and data is latched on the trailing edge of $\overline{AS}$. By using decoded address

to provide the $\overline{DTACK}$ signal, $\overline{DTACK}$ will remain asserted well into the next cycle. This violates the t_{SHDAH} specification in the data sheet. This decoded address signal, $\overline{RAM\ SELECT}$, is applied together with $\overline{AS}$ to an OR gate which provides a $\overline{DTACK\ RAM}$ input to AND gate U49. Thus, the $\overline{DTACK}$ input to the MC68000 is generated only when $\overline{AS}$ and $\overline{RAM\ SELECT}$ are both low, meeting the t_{SHDAH} timing specification.

Read cycles present no problems once the write cycle timing is met. The $\overline{UDS}$ and/or $\overline{LDS}$ data strobes fall at the same time as $\overline{AS}$ is asserted. The $\overline{RAML}$ and $\overline{RAMH}$ inputs are active as soon as the addresses are decoded. This decoding should provide ample times for the RAMs to respond with valid data and meet the 15 nanosecond setup time (t_{DICL}) before the falling edge of S6.

The ROM storage used in this application note is implemented using MCM68764s ($8K \times 8$) EPROMs with an access time of 350 nanoseconds. Since these EPROMs are much slower than the MCM2147 RAMs discussed above, several wait states must be inserted to read data from these devices. The decoding scheme used to address these ROMs is very similar to that used for the RAMs. However, in this case with the slower devices, the S-TTL parts are unnecessary. The ROM $\overline{DTACK}$ signal for the MC68000 is accomplished using a 74LS161 4-bit counter (U45) as shown in Figure 1. When both EPROMs are disabled, the 74LS161 counter is continually parallel loaded with the value 1011 (hexadecimal B). If one or both of the EPROMs become enabled, the 74LS161 counter then begins to count MPU clock cycles from 1011. When the count reaches 1110, the next input clock causes a logical 1 at the ripple carry out (RPC) output (pin 15). This rising edge clocks a 1 into the ROM DTACK flip-flop (U12, 74S74). The $\overline{Q}$ output of U12 applies a low to DTACK AND gate U49 which asserts the $\overline{DTACK}$ line of the MC68000. When $\overline{AS}$ is negated at the end of the read cycle, the 74S74 flip-flop is cleared, negating the processor $\overline{DTACK}$ input. A parallel load value of 1011 in the 74LS161 counter will cause three (or possibly four) wait states. Asynchronous input setup time t_{ASI} for the MC68000 $\overline{DTACK}$ recognition is 20 nanoseconds; however, the ROM DTACK circuit discussed above, under worst case delays (MPU-$\overline{AS}$ & TTL delays), does not meet the 20 nanosecond setup time. Thus, four wait states could be added by using a parallel load value of 1011; although, typically 1011 causes three wait states for ROM accesses. To change the number of wait states added to ROM accesses, simply reconfigure the parallel load value of the 74LS161.

Obviously, "no wait state" ROM access cannot be accomplished at 12.5 MHz with 350 nanosecond EPROMs (although bipolar ROMs could be used). However, one possible solution for faster ROM access would be a memory system in which, upon initialization, EPROM contents are transferred to "no wait state" RAM. This transfer could be accomplished quite rapidly using the block move routine shown below. This routine moves 48 bytes of data for each pass through the loop. Assuming a transfer from "three wait state" ROM to "no wait state" RAM, the MACSbug monitor ($6K \times 16$) could be moved in approximately 3.2 milliseconds. (Note that the routine shown below moves data in multiples of 48 bytes only. Any remaining bytes should be moved separately.)
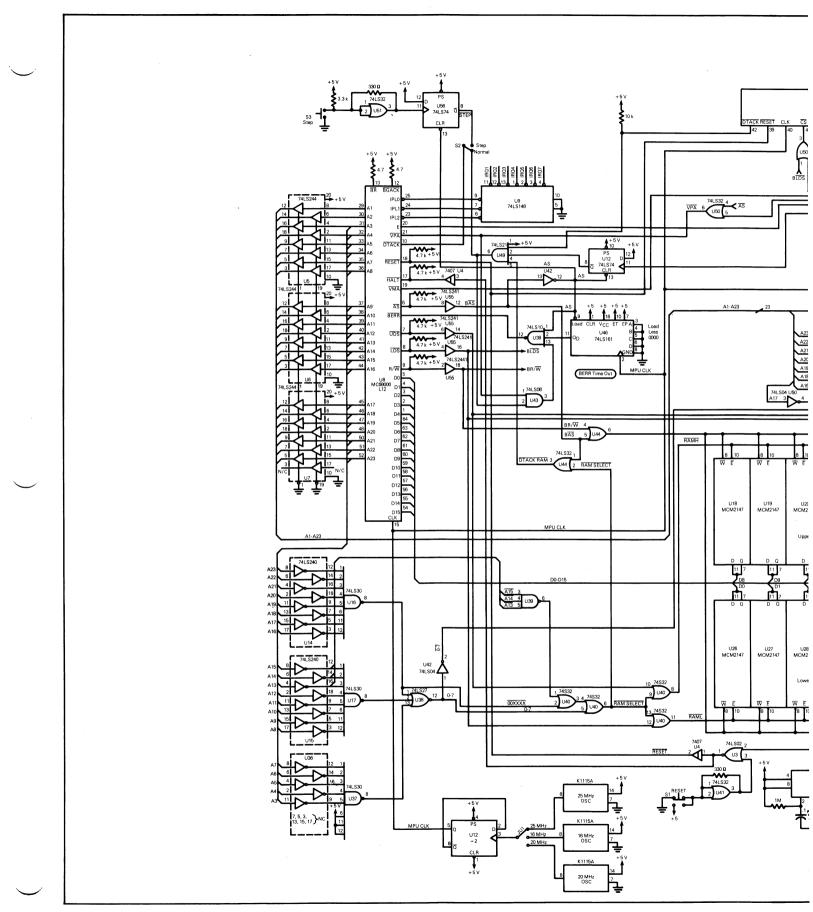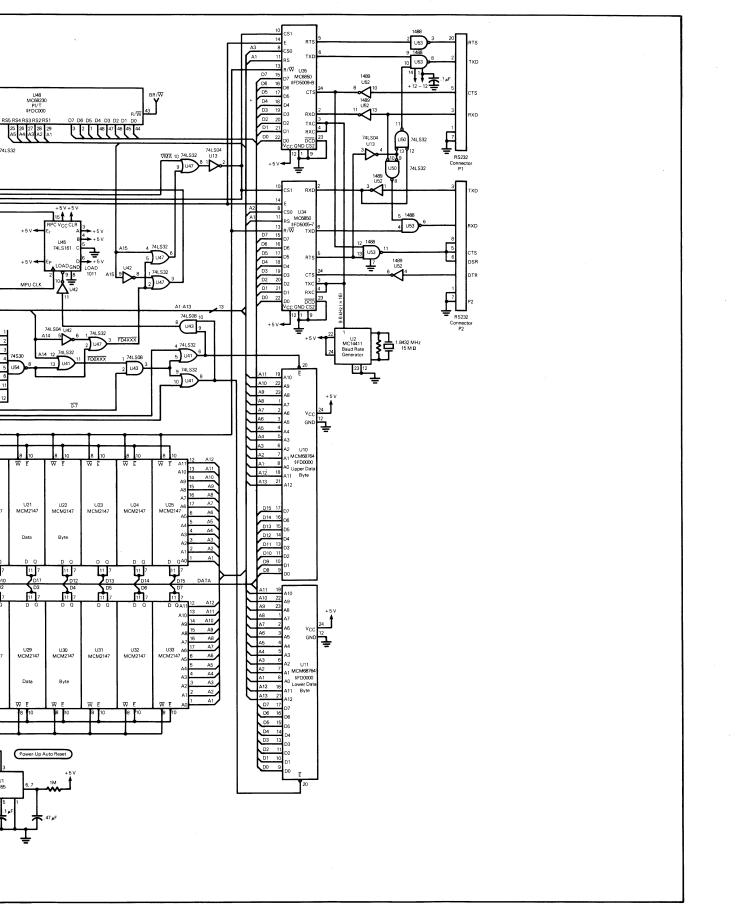
MACSbug is a trademark of Motorola Inc.

FIGURE 1 — High Performance MC68000L

3

```
 2                               *
 3                               *
 4                               ************************************
 5                               * THIS CODE SEQUENCE MOVES 48 BYTES OF *
 6                               * DATA FOR EACH PASS THROUGH THE LOOP. *
 7                               ************************************
 8                               *
 9          00FD0000     SOURCE   EQU   $FD0000         MACSBUG ROM LOCATION
10          00001000     DESTINTN EQU   $1000           RAM LOCATION
11          00001800     LENGTH   EQU   6144            NUMBER OF BYTES TO MOVE
12                               *
13 0 00000000 4BF900FCFFD0       LEA    SOURCE-48,A5     POINTER TO SOURCE
14 0 00000006 4DF82800          LEA    DESTINTN+LENGTH,A6  POINTER TO DESTINATION
15 0 0000000A 303C1800          MOVE.W #LENGTH,D0        # OF BYTES TO XFER
16 0 0000000E 4CF51FFE0000 LOOP MOVEM.L 0(A5,D0),D1-D7/A0-A4  MOVE DATA IN
17 0 00000014 48E67FF8          MOVEM.L D1-D7/A0-A4,-(A6)  SEND DATA OUT
18 0 00000018 04400030          SUB.W  #48,D0           48 BYTES EACH TIME
19 0 0000001C 66F0              BNE.S  LOOP             DONE?
20                               END
```

SYSTEM MONITOR

The system monitor used for this design is MACSbug 3.0 which is available from Motorola Microsystems in Phoenix, Arizona. The decoded addresses for ROM, RAM, and I/O are defined by this program. The address decoding provided for ROM access begins at location $FD000. The RAM space is decoded from $0 to $1FFF; however, to access the restart vectors, the first eight locations of ROM are decoded from $0 to $7. This decoding scheme disables the first 8 RAM locations.

SUMMARY

Enhanced throughput from an MC68000 system can be obtained by operating with no wait states. This application note has described one method for accomplishing these transfers. This method should prove to be an effective solution for the system designer.

List of ICs used in MC68000L12 System With No Wait States

IC Number	Part Number	IC Number	Part Number
U1	MC1455 Timing Circuit	U30, U31, U32, U33	
U2	MC14411 Baud Rate Generator	U34, U35	MC6850 Asynchronous Communication Interface Adapter
U3	74LS02 Quad NOR Gate		
U4	7407 Hex Buffer with Open Collector Outputs	U38	74S27 3-Input NOR Gate
		U39	74S10 3-Input NAND Gate
U5, U6, U7	74LS244 Octal Line Driver Non-Inverting Outputs	U40, U41, U44, U47	74S32 Quad OR Gate
		U50, U51	74LS32 Quad OR Gate
U8	MC68000 16 Bit Microprocessor	U43	74S08 Quad AND Gate
U9	74LS148 8 to 3 Encoder	U45, U46	74LS161 Sync. 4-Bit Counter
U10, U11	MCM68764 8K × 8 EPROM	U48	MC68230 Parallel Interface/Timer
U12	74S74 Dual D Latch	U49	74LS21 Dual AND Gate
U13, U42	74S04 Hex Inverter	U52	MC1489 Level Shifter/Line Driver
U14, U15, U36	74S240 Octal Buffer Inverted Outputs	U53	MC1488 Level Shifter/Line Driver
		U55	74S241 Octal Buffer Non-Inverting
U16, U17, U37, U54	74S30 8-Input NAND Gate	U56	74LS74 Dual D Latch
U18, U19, U20, U21	MCM2147 4K × 1 Static RAM	U57	24 MHz Oscillator
U22, U23, U24, U25	70 Nanosecond Access Time	U58	16 MHz Oscillator
U26, U27, U28, U29			