6809 Design:
Controller or System?

This chip is versatile enough for almost any application—from a simple black box controller to a complete disk-based business system.

Tim Ahrens
7405 Ladybug St.
Austin, TX 78744

Microprocessors have been traditionally broken up into two distinct groups: controllers and small personal systems.

Controllers can do everything from turning on lights to making better blends of gasoline. Some even count the number of French fries that go into each bag. As you can see, the microprocessor controller has many diverse applications in everyday life.

There are three elements to every controller—memory (ROM or RAM), I/O and the MPU. The ROM/RAM can be whatever size is necessary, and I/O can be either serial, parallel or both. The MPU should be easy to use, both in hardware and software. One of the best choices is the MC6809, the most advanced eight-bit microprocessor available.

A small system is an expanded controller and is used in applications ranging from hobby computers to small-business computers. They are single-user computers that run programs written in languages such as Pascal or BASIC. In some situations, software generation is the main purpose and is done with editors, assemblers and compilers. There are many more diverse controller applications than small systems, due to the nature of their environment.

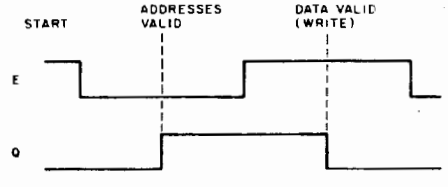


Fig. 1. E-Q relationships.

The MC6809—Hardware

The hardware features of the MC6809 make system design a snap. In traditional M6800 style, all peripherals are spoken to in a memory-mapped I/O fashion.

The MC6809 requires no complex clock generation devices: only a parallel resonant crystal across the Xtal and Extal pins with a frequency four times that of the bus. If you want an external frequency source, the Extal input will accept a TTL level of four times the bus frequency. Be sure to ground the Xtal pin when operating in this mode.

The crystal frequency is internally divided by four and then output on the E pin. In addition, a quadrature clock, Q, leads E by 90 degrees. (See Fig. 1.)

The falling edge of E signifies both the beginning and end of a cycle. On a read or write cycle, addresses, R/W and MPU status signals are valid on the rising edge of Q. This edge may be used to latch data. On a read cycle, data must be valid on the bus before the falling edge of E, which is late in the cycle. See the MC6809 data sheet for specific bus timing. Latching addresses or data is not required when using M6800 series peripherals, but interfacing to other devices may require these edges for timing purposes.

The reset input on the MC6809 is a Schmitt trigger input, which has a higher threshold voltage than standard peripherals. Peripherals thus come out of reset before the processor, and a simple R/C circuit resets the entire system. During power-on, reset should be held low until the clock oscillator is fully operational (about 100 ms). After that time, you may reset by holding the RESET line low for a minimum of one bus clock cycle.

Addresses are valid with the rising edge of Q. When the MPU doesn't need the bus for data transfer, it will output address $FFFF, R/W = 1 and BS = 0. Because of this, no VMA signal is used on the MC6809. If you want a retrofit to the MC6800 system, the VMA line may be tied high. The drive capability of the address bus in one Schottky TTL load and 90 pF. This makes single board design without buffers a reality.

The data bus provides bidirectional data transfers between peripherals and the MPU. The drive capability is one Schottky TTL load and 130 pF at related bus speed.

The HALT line will suspend program execution following the completion of the present instruction. When halted, BA goes high, indicating the address buses are in a high impedance state. Fig. 2 describes a simple single instruction stepper for the MC6809.

The MC6809 has four states that can be decoded by using the bus available (BA) and bus status (BS) pins:

BA	BS	MPU State
0	0	Normal (running)
0	1	Interrupt Acknowledge
1	0	Sync Acknowledge
1	1	Halt/Bus Grant

BA indicates that the MOS buses have been made high impedance, but does not mean that the bus will be available for more

than one cycle. BS, when decoded with BA, represents the MPU state.

The DMA/BREQ input lets you suspend execution and acquire the MPU bus for other uses, such as DMA and dynamic memory refresh.

A low level on the MRDY input pin allows E to be stretched in one-quarter bus cycle increments. This is useful when you are interfacing slow RAM, ROMs or peripherals to the bus. The maximum stretch is 10 us, due to the dynamic properties of the MPU.

The NMI, FIRQ and IRQ interrupt input pins provide the designer with methods of interrupting normal MPU operations.

NMI is the non-maskable interrupt pin. This input cannot be inhibited by the program. NMI finds general use in power-down applications, software refresh of dynamic RAM and real-time interrupt structures.

FIRQ is a fast maskable interrupt in the sense that only the program counter and condition code register are pushed upon the stack. The IRQ is an interrupt that can also be inhibited by program commands but will place all registers upon the stack when executed. For interrupt vector locations, see Fig. 3.

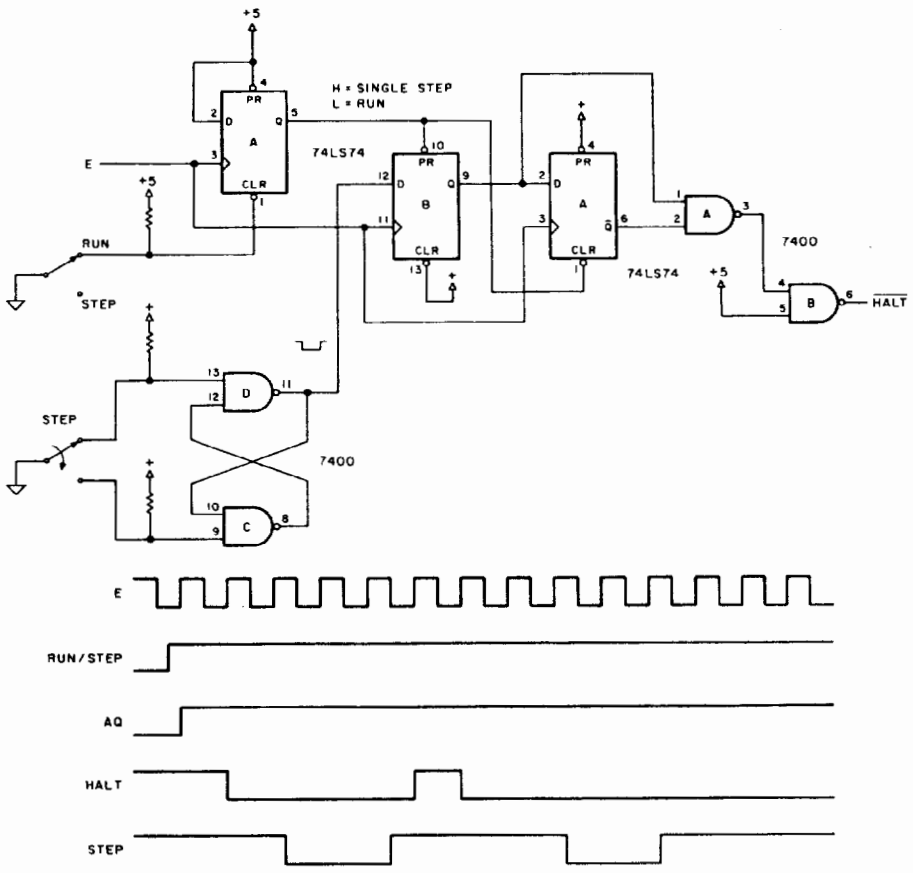


Fig. 2. Single stepper for the MC6809.

Software

While the MC6809 has hardware attributes, software is its forte.

The MC6809 gives you the following registers:

- two eight-bit accumulators, which can be concatenated into a single 16-bit wide register;
- two 16-bit indexable general-purpose registers;
- two 16-bit indexable-stack-type registers;
- one eight-bit direct page register; and
- one eight-bit condition code register.

See Fig. 4 for the MC6809 programming model.

Converting from 6800 to 6809 software is done by running the source code through a 6809 assembler or cross-assembler. Although the object codes for the 6800 and 6809 are noticeably different in most areas, numerous op codes have remained the same.

The addressing modes for the MC6809 are upward-compatible with the MC6800. The old modes have been kept and new ones added.

Direct addressing had previously been only in the lower 256 bytes of the memory map. This mode has been expanded to put that page anywhere in memory through the use of the direct page register (DPR). This register may be loaded with any value that will be the page in memory used for direct addressing.

For example, if the DPR contains $02, then any instruction that uses direct addressing will have the value of 02 put on the address bus as the most significant byte. Following a system reset, this register is cleared to be compatible with the MC6800.

Relative branching had been limited on the MC6800 to -125 or +127 bytes. In many cases, this restricted some programming applications and made position independent code (PIC) difficult without many alternate branches. The MC6809 allows relative branches to anywhere in the memory map (−32768 to +32767).

Another type of relative addressing is program counter relative. By using this mode, you can easily write position-independent code. For example, if you wanted to print a text string with the MC6800, the common method was:

```
LDX #MSG
JSR PRINT
```

Print is a routine within the code that prints text until you encounter an EOT character. This type of code is difficult to make position independent, but with the MC6809, PIC becomes very easy:

```
LEAX MSG,PCR
LBSR PRINT

MSG FCC/PRINT THIS/
```

The load effective address (LEAX) instruction takes the current offset from the program counter to the message, adds it to the PC and places it into the X register. Then, by doing a long branch to subroutine, the message gets printed. This code is fully position independent and thus executes properly anywhere within the memory map. The LEA instruction is available with any of the four indexable registers (X, Y, U or S).

The MC6809 has expanded index addressing modes, which include 0-, 5-, 8- and 16-bit constant offset, 8/16-bit accumulator offsets and auto increment/decrement. In addition, these indexing modes may have an extra level of indirection.

Indirect addressing is useful in many applications where addresses of parameters are taken on and off of the stack pointers. Such applications include higher-level languages such as Pascal and BASIC.

An example of how indirect addressing helps out when writing position independent code follows:

```
LDX#$E014  Loads X register with $E018 which is the address of the ACIA
PSHU X     Places $E018 on the U stack pointer
```

Now, any time data is to be loaded from

MS Byte	LS Byte	Function
FFFE	FFFF	RESET
FFFC	FFFD	NMI
FFFA	FFFB	SWI
FFF8	FFF9	IRQ
FFF6	FFF7	FIRQ
FFF4	FFF5	SWI2
FFF2	FFF3	SWI3
FFF0	FFF1	RESERVED

Fig. 3. Memory map for vector locations.

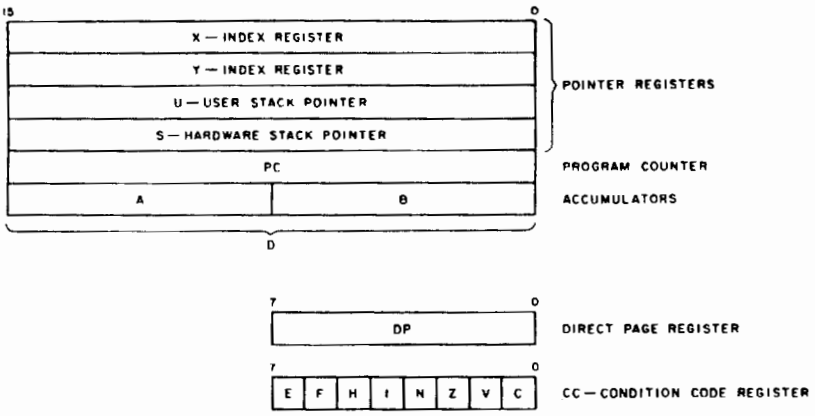


Fig. 4. Programming model of the MC6809.

the ACIA, only the following instruction is needed:

```
LDA [0,U]   Get data from ACIA
```

Note that many "addresses" may be placed on the stack and called in this indirect manner.

Miscellaneous

In the MC6809, any or all registers may be pushed onto the stack with a single instruction.

A multiply instruction multiplies the unsigned binary numbers in the A and B accumulator and places the unsigned result into the 16-bit D accumulator. This unsigned multiply also allows multiple-precision multiplications and takes only 11 machine cycles (5.5 us in a 2 MHz system).

The Basic Controller Design

What is required for a controller?

As mentioned earlier, the minimum is a microprocessor, program storage and I/O. The basic controller in this article contains an MC6809, two MC6821 parallel interface adapters (PIA), one MC6850 serial port (ACIA) and one EPROM of any desired density (MCM2708, 2716, 2532 or MCM68764). Also included is the necessary decoding and baud rate generation for the serial interface.

The bus frequency is 1.2288 MHz, corresponding to a cycle time of 813 ns. This frequency was chosen for one reason. 1.2288 × 4 = 4.9152 MHz, which is a common frequency and can be divided down by an MC14040 ripple counter to give most desired baud rates for the ACIA. Note that the bus speed is higher than that specified as the maximum rate for a standard MC6809. To be within specifications, an MC68A09 as well as A series peripherals are required.

To use standard 1 MHz parts, choose a 2.457 MHz crystal, which is still usable with the 14040. If the ACIA is not required or a different baud rate generation scheme is used, any crystal within frequency specifications may be used.

The decoding of this system is straightforward. If you anticipate no expansion over the original design, the 74LS42 may provide all necessary chip selects for the peripherals. The outputs of this 7442 are eight blocks of 8K. For a minimum parts count, tie each chip select of the RAM, ROM and peripherals to one of these outputs. The ROM must be the highest-order decode line ($E000 – FFFF).

Due to the incomplete decoding, each peripheral will occupy many locations within its respective block of memory. Here is an example of a decode scheme:

PIA 1	4000-5FFF
PIA 2	6000-7FFF
ACIA	8000-9FFF
RAM 1	A000-BFFF
RAM 2	C000-DFFF
ROM	E000-FFFF

Although incomplete decoding is used, it can be to your advantage. By addressing the lower bank of RAM in software as BC00 to BFFF and the upper bank of RAM as C000 to C3FF, you have 2K of contiguous RAM. This is possible due to the many mirror images that occur with incomplete decoding.

The R/W signal must be conditioned for use with 2114 RAMs. This conditioning effectively delays the valid R/W signal until the rising edge of E, which is halfway into the memory cycle.

The 74LS139 two- to four-line decoder is shown for those users who desire a more complete decoding scheme. By using the 139, these additional blocks may be decoded (see Fig. 5).

As mentioned earlier, the baud rates for the MC6850 are derived from the E clock through a CMOS counter. All common baud rates are available from 300 baud to 19.2 kilobaud, and if using the lower 2.45 MHz crystal, slide the taps down one for the correct baud rate.

RS-232 for the ACIA is provided by simple transistors, thus reducing cost over the traditional MC1488/89 receiver/transmitter devices.

A power-on reset circuit is provided in the 4.7k and 10 uF capacitor.

All unused inputs on the MC6809 are pulled up with 3.3k resistors for a wire-or capability. If you don't anticipate using these inputs, you can use a direct Vcc connection, further reducing cost and parts count.

You may further reduce the number of parts by using a device such as the MC6846, which includes 2K of mask programmed ROM, an eight-bit parallel I/O port and a 16-bit timer. Although this controller uses only a minimum of parts, its capabilities are great because of the flexible instruction set of the MC6809. See Fig. 6 for the complete schematic.

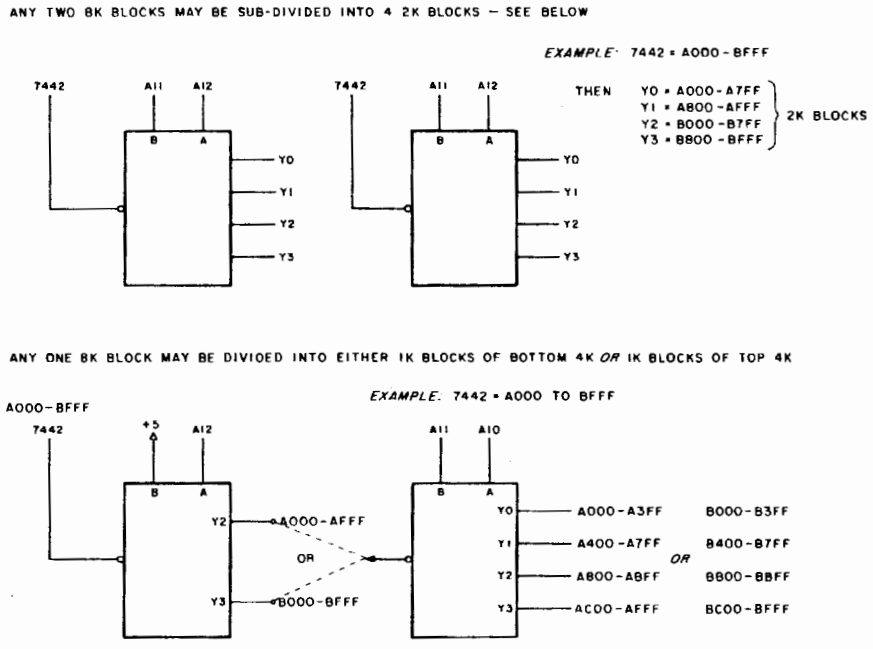


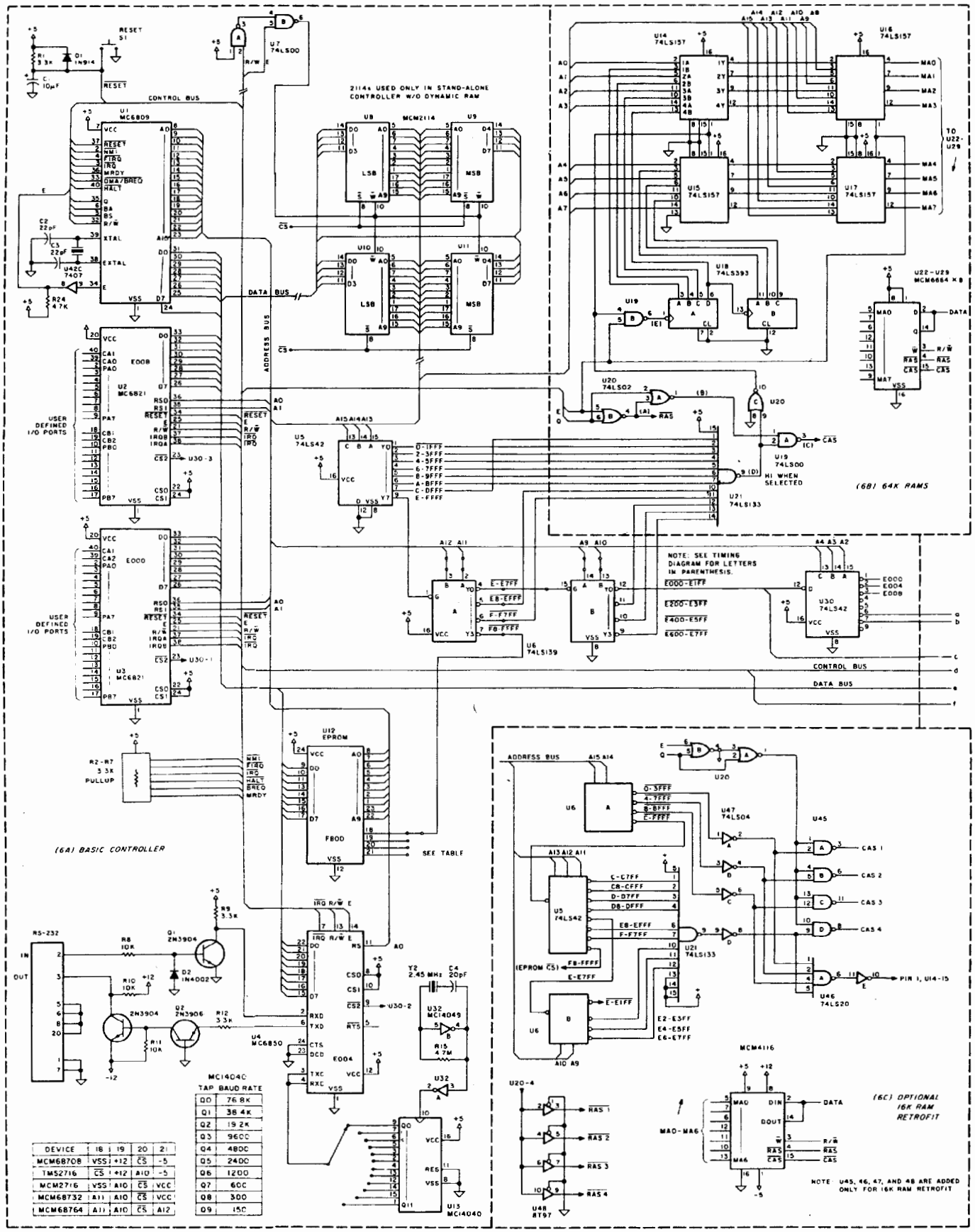
Fig. 5. 74LS139 additional decoding.

Due to its functions, almost every system design goes through many changes or even a total redefinition of its intended use. The basic controller circuit described earlier can be easily launched into the small business/personal computer market with a few expansions. These include full 64K RAM enhancement and a floppy-disk controller for program storage/recall.

The RAM

The RAM expansion circuit uses the new MCM6664 64K X 1 dynamic RAMs, but the techniques employed may also be used with the more common MCM4116 16K X 1 dynamic chips. Also included is an easy

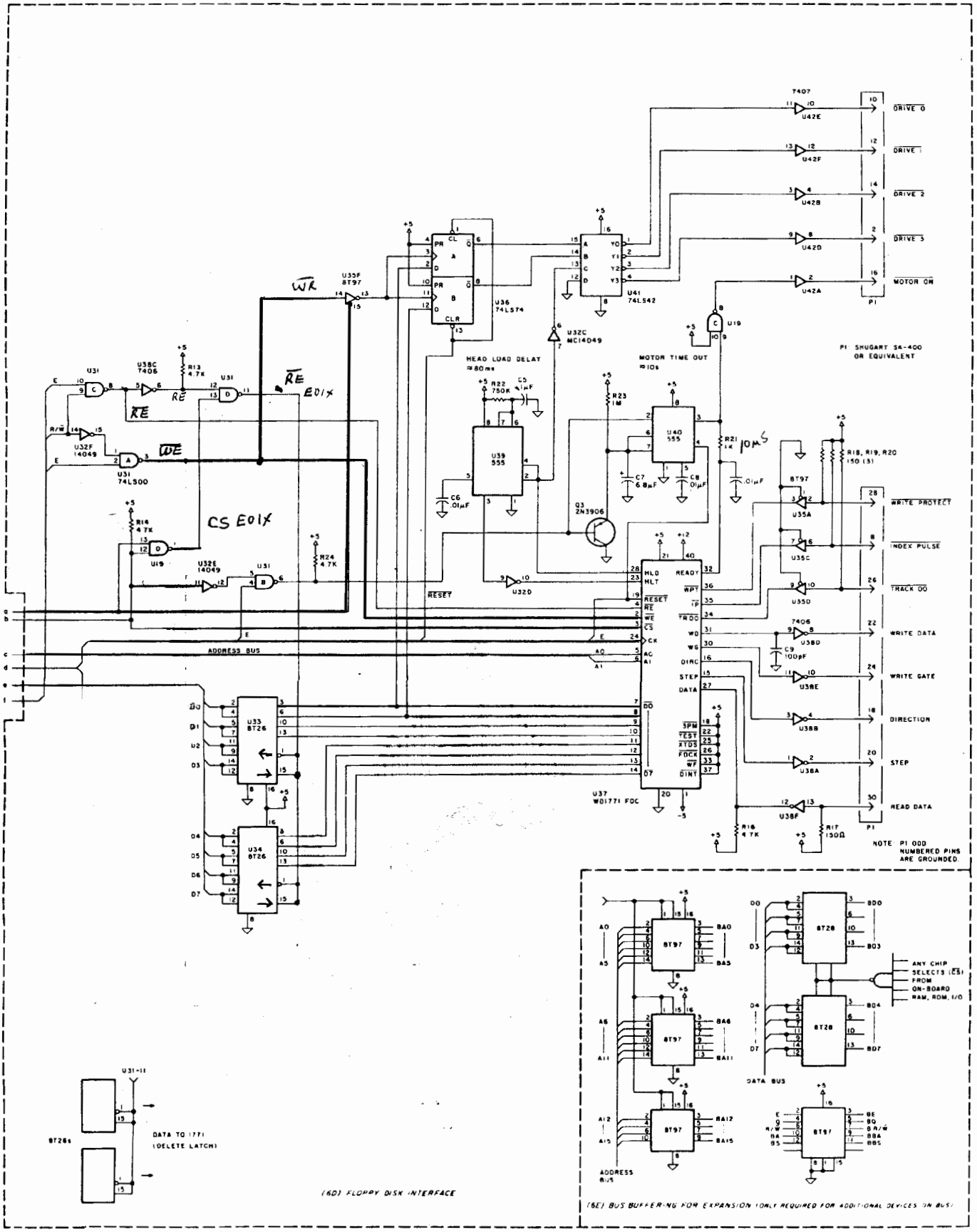
Fig. 6. Main schematic.



retrofit with 16K RAMs.

Dynamic RAMs, unlike their static counterparts, require a periodic "refreshing" to maintain integrity of the stored data. This refreshing can take on several different forms, one of which is discussed here – RAS only refresh. Fig. 7 shows how the RAS only refresh technique is used with the 6664s.

The dynamic RAMs have only eight address input lines, which select the desired memory cell within the chip. These address lines are multiplexed; that is, half of the addresses (the rows) are "strobed" in during the first part of the cycle, and the other half (the columns) are strobed in later in that same cycle. The waveforms in Fig. 8 show their relationship in the cycle.



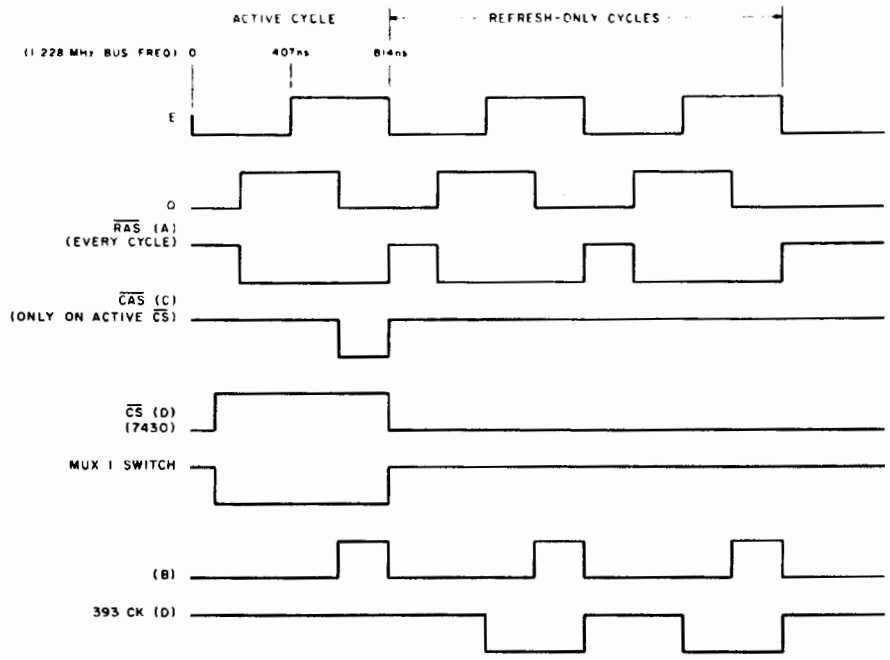


Fig. 7. RAS-only refresh timing.

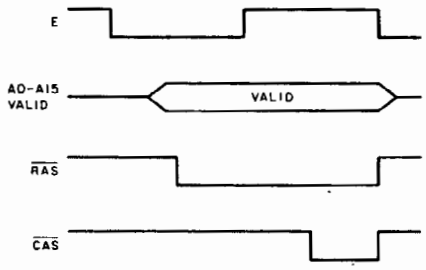


Fig. 8. RAS and CAS relationships.

As shown, the row addresses are strobed in on the falling edge of RAS, and the column addresses are strobed in on the falling edge of CAS. On a write cycle, the data should be valid on the falling edge of CAS, and on a read cycle, data comes out of the RAMs on the rising edge of CAS.

To retain the stored data, the RAMs must have every row accessed within 2 ms. Since a program execution generally does not access these rows within the required time period, you must use a hardware design to help out the refresh. The refresh schematic shows a pre-multiplexer, which selects either the regular row addresses from the MPU or a pseudo row address supplied by an external counter.

When the RAM is selected for a memory operation, the normal rows pass. At all other times, these pseudo addresses are supplied and are continually counting through the 128 rows. During the time that real addresses are being multiplexed, the clock signal going to the binary counter is held high until the pseudo addresses are required. Operation in this fashion ensures that all rows are accessed in an increasing manner, and no rows will be passed over during an access.

Following the decision point of normal or pseudo addresses (early in the cycle), the row addresses are multiplexed with the upper column addresses. These addresses, which come from the second set of 74LS157 multiplexers, are fed directly into the MCM6664s, which are decoded into actual memory cell locations within the RAM. Fig. 7 also shows the relationship between the multiplex switches and RAS and CAS.

The CAS signal is supplied by a chip select signal and the combination of E and Q. A chip select signal is obtained from a 13-wide NAND gate. The inputs to this gate come from appropriate address decoders. This CAS signal controls the actual data going into and coming out of the RAMs. Data must be valid on the falling edge of CAS (for a write), and data is valid on the rising edge of CAS (for a read). See Fig. 6b for the entire 64K RAM schematic.

16K RAMs

The design used for the 64K RAMs can also be applied to standard 16K X 1 dynamic RAMs. If you need only one bank (16K) of memory, you'll only need to modify the chip select circuitry to be more in keeping with a 16K block. Don't forget to put the appropriate voltages on the 4116s. (The MCM6664 is a single voltage part.) If you need additional banks of RAM, you must use separate CAS selections to differentiate which bank is selected.

All RAS lines may be tied together. Although more power will be used in this configuration, no additional circuitry is required for refresh generation. See Fig. 6c for CAS generation circuitry. Fig. 6c shows how standard 16K dynamic RAMs may fit into the expanded system. The decoding portion of the schematic uses the same number of devices—one 74LS42 and one 74LS139—but they are arranged in a different fashion than that of the controller schematic. Portions of the CAS selection circuitry have been kept, and others have been 16K RAM retrofit.

Floppy Disk Interface

In most applications with more than a few K of RAM, some type of high-speed mass storage system is used. Many times this is cassette tape, hard or floppy disks.

Most microcomputer systems use floppy disks of either the 5¼- or 8-inch variety. I'll describe an interface for a minifloppy drive, although an eight-inch drive could be used with an external data separator and a processor speed greater than 1.5 MHz.

Most of the interface involves standard decoding and buffering of necessary buses, although the FD1771 does require some

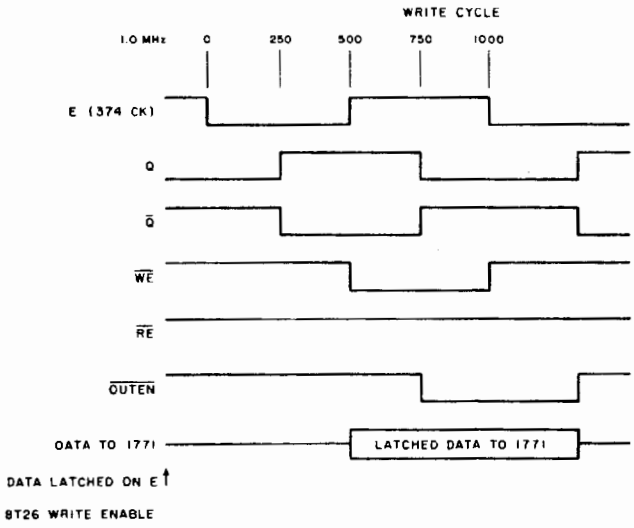


Fig. 9. Floppy disk controller timing.

strange circuitry to work with the MC6800 or MC6809.

The first is that of the R/W line. The 1771 uses separate read and write enable signals. These are derived from the R/W line and E. Each of these signals is valid for the entire E high time.

The other circuit is required for latching data into the FDC on a write cycle. The 1771 data sheet states that data must stay valid on the data bus for at least 150 ns after the WE pulse goes high. This data is valid for only 30 ns on the MC6809, or about 10–20 ns on the MC6800/6802. Because of this, a latch is needed to hold the data on the bus. In the read mode, no circuitry is required because the 1771 holds the data for more than the minimum that is specified by the MC6809.

This data-hold specification of the MC6809 denotes how long the bus drivers of the MPU are actually turned on, and not how long the data will be held on the bus. This time on the bus can vary, mainly with the amount of loading that is present. If TTL or other heavy-load devices are present on the bus, the decay time will be shortened.

But, if all that are present are MOS parts and other high-impedance devices, this hold time will traditionally be until the MPU bus drivers are driven to their opposite states (as early as the next cycle). In the given example, U44, U43, U45A, U20D and U7D may be taken out (see Fig. 6d). To be within the guaranteed specifications of both the 1771 and MC6809, these parts must be installed, but, in this application, the characteristics of a MOS bus may be used to your advantage to save PC board space and parts count (see Fig. 9).

Fig. 6d shows the schematic used as the floppy disk interface. You can use standard Shugart SA-400 or equivalent disk drives.

Minifloppies generally use a dc motor for the diskette drive motor, thus shortening their useful operating life. To make more efficient use of this time and to save the oxide on the diskette, you can turn off the drive motor when not accessed.

U40 (MC1455) turns on the drive motor when location $E018 is accessed. This is the base location of the 1771, so any access on the FDC will restart the drive motor. During any nonactive 1771 time, the 555 will hold the drive motors on for about ten seconds before shutting down. This time is determined by the value of C6.

Another 1455 provides the head load timing delay. This time is about 80 ms, which gives the head enough time to settle before signaling the 1771 that data transactions may take place.

Drive selection is determined by U36 and U41. U36 provides a way to latch information from the data bus. This information is the drive number and is sent to U41, which

✓ Reader Service index—page 241

```
                    *  THIS 'MINI-MONITOR' IS ALL THAT IS REQUIRED
                    *  IN THE EXPANDED SYSTEM TO PROVIDE THE USER
                    *  WITH A BOOT FOR THE 'FLEX' DISK OPERATING SYSTEM.
                    *
                    *  THE SYSTEM AUTOMATICALLY BOOTS UP FROM RESET TO
                    *  THIS ROUTINE.  OTHER ROUTINES MAY BE PUT INTO
                    *  ROM, PROVIDING THE USER WITH MORE CAPABILITIES.
                    *
                    *  THIS CODE IS COURTESY OF TSC INC.
                    *
               E014 DRVREG  EQU    $E014     DRIVE REGISTER
               E018 COMREG  EQU    $E018     COMMAND REGISTER OF 1771
               E01A SECREG  EQU    $E01A     SECTOR REGISTER OF 1771
               E01B DATREG  EQU    $E01B     DATA REGISTER OF 1771
                    *
F800                        ORG    $F800
                    *
F800 B6   E018      START   LDA    COMREG    TURN MOTOR ON
F803 86   00                LDA    #0
F805 B7   E014              STA    DRVREG
F808 8E   0000              LDX    #0000
F80B 3D             OVR     MUL               DELAY FOR SPEED UP
F80C 30   1F                LEAX   -1,X
F80E 26   FB                BNE    OVR
F810 C6   0F                LDB    #$0F      RESTORE
F812 F7   E018              STB    COMREG
F815 8D   2B                BSR    RETURN
F817 F6   E018      LOOP1   LDB    COMREG
F81A C5   01                BITB   #1
F81C 26   F9                BNE    LOOP1
F81E 86   01                LDA    #1
F820 B7   E01A              STA    SECREG
F823 8D   1D                BSR    RETURN
F825 C6   8C                LDB    #$8C      READ WITH LOAD
F827 F7   E018              STB    COMREG
F82A 8D   16                BSR    RETURN
F82C 8E   C000              LDX    #$C000
F82F C5   02        LOOP2   BITB   #2         DRQ?
F831 27   05                BEQ    LOOP3
F833 B6   E01B              LDA    DATREG
F836 A7   80                STA    0,X+
F838 F6   E018      LOOP3   LDB    COMREG
F83B C5   01                BITB   #1         BUSY?
F83D 26   F0                BNE    LOOP2
F83F 7E   C000              JMP    $C000
F842 8D   00        RETURN  BSR    RTN
F844 39             RTN     RTS
                    *  RESTART VECTORS
FFFE                        ORG    $FFFE
FFFE F800                   FDB    START
                            END

0 ERROR(S) DETECTED

SYMBOL TABLE:

COMREG E018   DATREG E01B   DRVREG E014   LOOP1  F817   LOOP2  F82F
LOOP3  F838   OVR    F80B   RETURN F842   RTN    F844   SECREG E01A
START  F800
```

Mini-monitor listing.

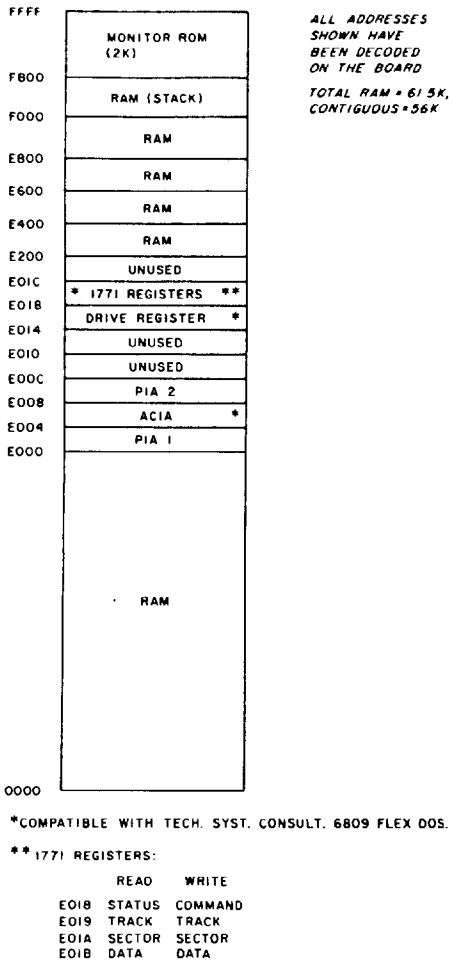


Fig. 10. System memory map.

decodes which drive is to be selected.

System Thoughts

The system timing signal E is used by all peripherals, including the 1771 for data transfers. For a controller or other small system, clock rates of up to 2 MHz may be used with the MC6809. Note, however, that the 1771 will not work much above 1.25 MHz when used in the shown configuration. I am using a 4 MHz crystal on the MC6809 and 2.45 MHz crystal for baud rate generation. The memory map for the entire expanded system is shown in Fig. 10.

Expansion

Although RAM expansion for this system over 64K is not practical, except with address translation circuits, other devices such as EPROM programmers, I/O cards, graphics cards and printer driver cards may be necessary in an expanded or small-business system configuration. Fig. 6e shows how the address/data buses may be buffered to supply the necessary signals for other cards on the bus. This bus may be anything that is close at hand, or it may be the standard Exorciser or SS-50 bus. Note that no buffering of the address or data buses is required on the single board expanded system because of the drive capabilities of the MC6809. With no software, the most elaborate piece of hardware is reduced to a pile of junk.

Rather than write an entire disk operating system (DOS), which might take me forever, I looked into the systems already available for the MC6809. Flex from Technical Systems Consultants proved to be the best choice as a DOS from both a cost and capabilities viewpoint. TSC has consistently featured excellent software at an affordable price ever since the advent of the MC6800. The new 6809 Flex has kept all of the capabilities of the standard 6800 Flex, so a conversion from an existing MC6800 system would not be too great. TSC also offers a wide range of Flex-compatible software, which includes an extended BASIC and an extensive debug package.

Since almost all operations use Flex, a small monitor ROM is all that is required. Any software debugging operations may be done with the debug package. The monitor ROM contains the following functions:

INITIALIZE FLEX

Now, that's a small monitor program! The monitor may be put in almost any type of ROM but must be placed at the top of memory so the MC6809 may get the appropriate restart vectors. See the monitor listing.

Conclusion

In these days and times, it doesn't take much to make a complete system. Whether 64K or 16K RAMs are used, this design can fill many requirements of either the controller or small systems market.■