GMX™ 6809 CPU III
MICROPROCESSOR BOARD

User's Manual

GIMIX Inc.
1337 W. 37th Place
Chicago, IL 60609
312-927-5510 * TWX 910-221-4055

GMX™ 6809 CPU III

The GMX 6809 CPU III is an advanced microprocessor board, specifically designed for use with multi-user/multi-tasking operating systems.

FEATURES

* 2 MHz 68B09 CPU

* Expanded memory management (DAT) supports up to 1 MByte of memory, in 2K segments.

* Allows a full 64K of RAM in a users address space with no system overhead.

* Memory segment attributes that provide:

  Trapping of out-of range memory references.
  Write protection of individual segments.
  Protection against certain illegal instructions.
  Hardware single-step.

* High-speed Memory-to-Memory and Memory-to-I/O DMA transfers at 1 byte per microsecond.

* Arbitrates contention between the on-board DMA and external DMA devices.

* Separate supervisor and user "states" to prevent unauthorized access to system information and devices.

* Automatic switching to the Supervisor State in response to hardware and software interrupts.

* Full function Time-Of-Day clock with: year, automatic leap year/daylight savings time correction, and battery.

* 6840 Programmable Timer with a separate 500 Khz precision (.0025%) time base oscillator.

* Accepts one 2K, 4K, or 8K EPROM for on-board firmware, up to 4K mapped into the address space at one time.

* Software or hardware selection of the upper or lower half of an 8K EPROM for software switching applications.

* 2K CMOS scratchpad RAM with battery backup.

INTRODUCTION

Although the 6809 is one of the most powerful 8-bit microprocessors available, it lacks some features that are important to system security and performance in multi-user/multi-tasking applications. The GMX 6809 CPU III was designed to enhance the performance of the processor and increase the security of these systems.

In a multi-user system it is vital that the actions of a particular user can not and do not affect others who may be using the system. While the operating system itself can provide a certain degree of protection, by restricting file access for example, many forms of protection can only be provided by the hardware or a combination of hardware and software. If the user is to be given direct access to the system, as is necessary in assembly language software development, he must be denied access to areas of the system that could adversly affect the operating system or other users. In particular, he should not be able to directly access memory assigned to other users, memory assigned to the operating system itself, I/O devices which may be used by others, the memory management hardware, or memory containing software that is being shared with other users. The CPU III restricts this type of access in several ways. In conjunction with the memory management hardware, each 2K segment of the 1 Mbyte address space can be assigned certain "attributes". These attributes are used to prevent unauthorized access to the memory (UnAllocated Memory attribute, UAM) and prevent unauthorized writes to the memory (Write ProTect attribute, WPT). Further protection is provided by the creation of separate Supervisor and User "States". Access to certain functions and memory areas is restricted to the Supervisor state, preventing unauthorized access.

The 6809 offers no protection against the execution of any of its undefined or "illegal" op-codes. While the CPU III does not trap all illegal op-codes, it does trap those that cause the 6809 to stop processing and enter a state in which it does not respond to any interrupts. To recover from this state the processor must be reset. (The remaining undefined op-codes generally do not effect the operation of the system as a whole. Their effects, if any, are limited to the task in which they are executed or they will be trapped by the attribute traps.) The WatchDog Counter detects this condition by monitoring the the processors response to interrupts while in the User State. If the processor does not service an interrupt within a specific number of clock cycles the WDC resets the 6809 (other devices connected to the reset line are not affected) and a special reset vector is asserted. This allows the state of the system to be preserved and normal operation (with the exception of the task that caused the trap) can be resumed. As an added benefit, the WDC limits the length of time that a users software can keep the interrupts masked. Without this limitation, a user could mask the interrupts indefinitly, effectively preventing the processor from servicing other users.

In addition to system security, system performance (speed) is an extremely important consideration in a multi-user environment. While the 6809 is a fast and efficient processor, the overall performance can be improved by the addition of hardware to increase the speed of certain operations. Multi-user systems can spend a considerable

amount of time moving large blocks of data from one area of memory to another or from memory to I/O devices.

In spite of the 6809's multiple index registers and auto-increment/decrement modes, large block moves require considerable processor time. A simple move using the 6809 (lda <addr>, sta<addr>) takes a minimum of 8 machine cycles per byte or 4 microseconds at 2 MHz. The CPU III includes on-board Direct Memory Access (DMA) circuitry to move data from memory to memory or to I/O devices at 1 byte/microsecond (2 clock cycles).

To improve interrupt handling, the CPU III automatically switches to the Supervisor State (and Task Map 0) in response to both hardware and software interrupts. This eliminates the need to map the interrupt vectors into each tasks address space. A full 64K of address space is available to each user (task) with no system overhead.

In addition to security and performance features, the CPU III provides a full function Time-Of-Day clock. The clock is maintained by a rechargeable battery supply when the system power is removed. It includes all of the normal time and date functions, including the year, as well as automatic compensation for daylight savings time amd leap years, alarm interrupt capabilities, and 50 bytes of battery backup RAM.

For timing and counting functions, the CPU III includes a 6840 Programmable Timer with a separate 500 KHz precision time base oscillator that can be easily replaced by the user to provide other frequencies up to 750 KHz.

The 28 pin EPROM socket accepts 2K, 4K, or 8K single-supply devices. When using 8K devices, only 4K is mapped into the address space. The user has the option of selecting between the upper and lower halves of the EPROM under hardware (jumper) or software control, for software selection and switching applications.

1337 W 37th Place
Chicago, IL 60609
(312) 927-5510 * TWX 910-221-4055

TABLE OF CONTENTS

(cont.)

The CPU III has several hardware options which must be configured to suit the application. In most cases these options are factory configured before the board is shipped and do not need to be reconfigured by the user. See Appendix a for a list of standard jumper configurations.

1-1: Watchdog Counter Delay Option (JA-1)

This jumper determines the number of clock cycles that will be counted between the time that an interrupt occurs and the generation of a watchdog trap (if the watchdog is enabled). The watchdog can be set to count either 32 or 128 cycles. See sheet 2 of the JUMPER OPTIONS drawings, figures GG and HH and the section on the Watchdog Counter, page 8.

1-2: 6840 External Connection Jumpers (JA-2)

JA-2 provides access to the clock inputs, gate inputs, and the outputs of the three sections of the 6840 PTM and the output from the precision time base oscillator (500 KHz. standard). Unless required by the operating system, the 6840 is available for application as determined by the user. The timer sections can be interconnected by installing the appropriate jumpers or connected to external devices using a user supplied cable. See sheet 2 of the JUMPER OPTIONS drawings, figures II, JJ, and KK for the pinout of JA-2 and interconnection examples.

1-3: Missing Cycle Detector Interrupt Select Jumper (JA-3)

JA-3 connects the interrupt input pin from P2 (the Missing Cycle Detector board connector) to one of the three system interrupt lines (NMI, IRQ, or FIRQ). The required interrupt line is selected by installing a jumper in one of three locations. See sheet 1 of the JUMPER OPTIONS drawings, figures A and B and the section on the Missing Cycle Detector, page 19.

1-4: Timer and Clock Interrupt Select Jumpers (JA-4)

This jumper area allows the interrupt outputs from the 6840 PTM and the 146818 Time-Of-Day clock to be connected to the system interrupt line(s). The two devices can be independently jumpered to generate any one of the three interrupts (NMI, IRQ, or FIRQ). See sheet 1 of the JUMPER OPTIONS drawings, figures C and D and the appropriate manual sections.

1-5: On-board Device Access Select Jumper (JA-5)

JA-5 determines when the on-board, memory mapped devices (timer, clock, PROM, RAM, and control registers) can be accessed. In the "TASK 0/SUPERVISOR" position, these devices can only be accessed when the system is in the Supervisor State. In the "All Tasks" position the devices can be accessed from any task in either supervisor or User State. See sheet 1 of the JUMPER OPTIONS drawings, figures E, F, and G.

1-6: Bus NMI-line Option Jumper (JA-6)

This jumper allows the function of the NMI line on the 50 pin bus (normally an interrupt input to the board) to be changed to an output, from the CPU board, that can indicate to other boards in the system that the CPU board is in the "TASK 0/ SUPERVISOR" state. See sheet 1 of the JUMPER OPTIONS drawings, figures H, I, and J.

1-7: Battery ON/OFF Jumper (JA-7)

JA-7 allows the on-board battery to be disconnected when servicing or transporting the board outside the system. JA-7 should always be set to the OFF position when changing parts or servicing the board (especially the clock and scratchpad RAM). CAUTION: Even when JA-7 is in the OFF position, parts of the circuit are "live". Use caution when handling or servicing the board to prevent accidental short circuits which could damage the battery and other parts of the board. The jumper must be in the ON position to provide battery backup for the clock and scratchpad RAM. See sheet 1 of the JUMPER OPTIONS drawings, figures K, L, and M.

1-8: 8K EPROM Block Select Jumper (JA-8)

When an 8Kx8 EPROM is used for on-board software, only one half of the device is mapped into the system address space at any one time. There are four options, selected by JA-8, that determine which half of the EPROM is being used. The options are: upper 4K always enabled; lower 4K always enabled; upper 4K enabled at reset, lower 4K may be selected by software switching; and lower 4K enabled at reset, upper 4K may be selected by software. See sheet 1 of the JUMPER OPTIONS drawing, figures N through R.

1-9: Sense-bit Select Jumper (JA-9)

The position of this jumper determines the status of a spare read-only status bit in the TASK STATUS REGISTER. If a jumper is installed at JA-9, bit 4 in the TSR will read back as 0 (zero), if the jumper is removed, the bit will read back as 1 (one). See sheet 1 of the JUMPER OPTIONS drawing, figures S and T.

1-10: Test Points (JA-10)

JA-10 provides access to test points for adjusting the clock oscillator for the Time-Of-Day clock and the Unsafe-voltage/power-down circuits. See Sections 7-3 and 12-1.

1-11: Time-Of-Day Clock Reset Option Jumper (JA-11)

The CPU III offers two options for resetting the 146818 Time-Of-Day clock. In the normal reset mode the clock is connected to the system reset line and is reset whenever the system is reset (on power-up or a front panel reset). In the reset on battery fail position, the clock is reset only if the battery discharges below the level where the clock can operate reliably. See sheet 2 of the JUMPER OPTIONS drawing, figures U, V, and W.

1-12: Scratchpad RAM Size Select Jumper (JA-12)

JA-12 selects the size of the scratchpad RAM. In the 1K position, only the lower half of the RAM appears in the address space. In the 2K position, the entire RAM is available. See sheet 2 of the JUMPER OPTIONS drawing, figures X, Y, and Z.

1-13: Time-Of-Day Clock Write-protect Enable Jumper (JA-13)

JA-13 enables the write protect option for the Time-Of-Day clock. If this option is enabled, the clock can only be written to if bit 3 of the Task Select Register is set to a 1 by the user. When bit 3 of the TSR is cleared (0), writes to the clock are ignored. The bit is cleared on power-up or reset, and must be set (1) before data in the clock can be modified. If this option is disabled, the clock can be written to regardless of the state of TSR bit 3. See sheet 2 of the JUMPER OPTIONS drawing, figures AA, BB, and CC.

1-14: EPROM Size Select Jumper (JA-14)

JA-14 must be set to match the size of the EPROM installed on the board (U-37). It has two positions: one for 2K devices, and one for 4 and 8K devices. See sheet 2 of the JUMPER OPTIONS drawing, figures DD, EE, and FF.

***** NOTE *****

Throughout this manual references made to specific addresses will be in one of two forms. Generally, four digit addresses (e.g. $F800) refer to logical addresses as they would appear in a section of code or on the address outputs of the 6809. Five digit addresses (e.g. $FF000) refer to physical addresses within the 1 megabyte address space. The address space is divided into 16 banks of 64K each. The first digit corresponds to the extended or bank address.

The memory management hardware on the CPU III serves a dual purpose. First, using a technique called Dynamic Address Translation, it extends the addressing capabilities of the 6809 from 16 to 20 bits, allowing the processor to access 1 megabyte of memory. Second, it allows certain "attributes" to be assigned to the memory. These memory "attributes" are used to restrict access to certain areas of the memory space and provide a hardware single-step function for software debugging.

2-1: Dynamic Address Translator (DAT)

The DAT is a high speed, random access memory (DAT RAM) placed between the upper 5 address lines of the 6809 (A11-A15) and the system address bus. These 5 lines, along with 3 outputs from the Task Select Register (TSR), form the address input of the DAT RAM. The DAT RAM outputs 12 bits of data. The lower 9 bits of the DAT RAM output become the upper 9 bits of the system address bus (A11-A19); the remaining 3 bits are used for the memory attributes. Thus, when the processor outputs an address to access a memory location (logical address), the upper 5 bits of that address along with the three bits from the TSR define a location in the DAT RAM. The data at that location in the DAT RAM then becomes the actual address (physical address) placed on the system address bus. This translation of logical to physical address takes place each time the processor accesses a location in memory.

Since only the upper 5 bits of the processors address output are translated by the DAT, the processors 64K logical address space is divided into 32 "segments", each 2K (2048) bytes long. For each of the 32 segments there is a corresponding location in the DAT RAM. The data stored in that location becomes the physical address placed on the bus when the processor accesses that segment of it address space. The 32 locations in the DAT RAM form a "task map". By storing the appropriate data in each location in the task map it is possible to map any combination of 2K segments from the 1 megabyte physical address space into the processors 64K address space. The DAT RAM can hold up to 8 task maps simultaneously. The task map in use at any particular moment is determined by the value in the lower 3 bits of the Task Select Register.

Data is stored in the DAT RAM by writing to addresses $F800 through $F9FF. The DAT can only be accessed when the board is in the Supervisor State. Data written to these addresses is stored in the DAT RAM; reads from these addresses return data from the EPROM or other memory mapped there. Each of the 256 locations in the DAT RAM (32 segments x 8 task maps) occupies two sequential bytes in this address range. The first byte (even address) is for the 3 memory attribute bits and A19, the second byte (odd address) is for A11 through A18. (See the Dynamic Address Translator drawing, page 25.)

On power-up the DAT RAM contains random data. In order to guarantee a known condition on power-up and allow the DAT RAM to be initialized, a special state is forced on power-up. In the power-up state, address lines A11 through A19 are forced high (1), making the upper 2K of the on board EPROM accessible at $F800 through $FFFF. This power-up state exists only until the first write to an odd byte in the DAT RAM. Once data has been written to an odd byte in the DAT RAM, the power-up state ceases to exist and the DAT functions normally. Before any other initialization of the DAT is performed, the reset routine must set up the last segment of task 0 as addresses $FF800-$FFFFF as shown in the following example.

```
DAT     equ     $F800       Base address of DAT RAM

        ldd     #$1FF        Translate physical addresses
        std     DAT+62      $FF800-$FFFFF to the logical
                            segment at $F800-$FFFF in task 0
```

At the completion of the write to the second byte the power-up state ceases to exist; however, the value just stored in the DAT RAM maps the upper 2K of the EPROM into the processors address space so that the initialization can continue. At this point the only logical addresses that are valid are in the $F800-$FFFF segment of task 0. All other memory remains undefined due to the random contents of the DAT RAM. Before memory outside this 2K segment can be accessed, the appropriate sections of the DAT must be initialized to known values.

Because the system automatically switches to the Supervisor State and the task 0 map in response to interrupts (see the Task Control section), the reset and interrupt vectors are only required in the supervisor state. When the system is in the User State the entire 64K of logical address space is available to the user. It is not necessary (and normally not possible, see JA-5) to map the on-board EPROM into the users address space. However, memory containing the interrupt handlers (either the on-board EPROM or other EPROM in the system) must be mapped into the task 0 address space.

***** NOTE *****

The Reset and Interrupt vectors in the top 16 bytes of the on-board EPROM are always mapped into the top 16 bytes of the logical address space ($FFF0-$FFFF) when the system is in the Supervisor State. These vectors can not be replaced by re-mapping with the DAT. The remainder of the EPROM can be remapped if required (see above).

2-2: Memory Attributes

Three bits in the DAT RAM are used to associate certain "attributes" with each 2K segment of memory that is mapped into the processors logical address space. These attributes are: UnAllocated Memory (UAM), Write ProtecT (WPT), and Single STep (SST). With these three attributes the security and reliability of the system can be greatly improved, and powerful debugging tools can be implemented.

The attributes for a particular segment are enabled by setting the appropriate bit(s) in the DAT RAM when the segments address is written. Anytime a memory segment is accessed while the system is in the User State, the attribute bits are checked. If the access violates the conditions of the attribute, for example if a write is attempted to a segment that has its WPT attribute bit set, and the WPT enable bit in the TSR is set, a "trap" occurs. The result of a trap depends on the particular attribute.

The WPT attribute is used to inhibit writes to a segment or group of segments. If the processor attempts to write to a segment that has its WPT attribute bit set a WPT trap occurs. The write attempt is blocked by the hardware to prevent corruption of the data in the protected memory and the WPT flag bit in the Task Status Register is set to indicate the cause of the trap. The trap also generates an IRQ interrupt and if the processor's interrupt mask is not set, this interrupt will be serviced at the end of the instruction that caused the trap. Unlike a normal IRQ, an IRQ caused by a trap is vectored through a special interrupt vector at instead of the normal IRQ vector. The vector at $FFFF0, normally undefined by the 6809, must point to a trap handling routine.

The UAM attribute is used to inhibit all accesses (read and write) to a segment or group of segments. If the processor attempts to access a segment that has its UAM attribute bit set, a UAM trap occurs. The memory access is blocked by the hardware (writes have no effect, reads return invalid data) and the UAM flag bit in the Task Status Register is set. An IRQ interrupt is generated, which is processed in the same way as an interrupt caused by a WPT trap.

The SST attribute is used to allow execution of a program, one instruction at a time, for testing and debugging. After each instruction is executed the system returns to the Supervisor State so that the programs operation can be examined and traced. The system returns to the Supervisor State in a manner similar the WPT and UAM attributes. Whenever the processor executes an instruction in a segment that has its SST attribute bit set, a trap occurs. The trap has no effect on the instruction causing the trap; it completes normally. After the instruction is completed an IRQ is generated, which is processed through the special interrupt vector. Since the trap occurs only at the completion of the instruction, multi-byte instructions only cause one trap. Traps are normally caused only by an instruction fetch, unless an instruction executed in a memory segment that does not have the SST attribute set, accesses data in a segment that does.

***** NOTE *****

The SST trap flag in the TSR does not behave like the UAM and WPT trap flags described above. The UAM and WPT flags in the TSR are only set when a trap occurs. The SST trap flag is set as soon as the SST is enabled by writing a 1 (one) to the SST enable bit (TSR bit 5).

Interrupts caused by the attribute traps, and interrupts caused by the Watchdog Counter are processed through a special interrupt vector at $FFFF0-$FFFF1; asserted in place of the normal IRQ vector by the CPU III. This "trap vector" must point to a routine which identifies the cause of a trap and takes appropriate action. The four "trap flags" in the Task Status Register (TSR) are used to identify the source of a trap. The WDC, UAM, and WPT flags are set only as a result of their associated traps and should be checked first. If none of these three bits is set, the SST bit should be checked next. If the SST bit is set, indicating that the SST trap is enabled, then it can be assumed that the trap was an SST. NOTE: It is possible for a single event to cause more than one type of trap, resulting in more than one trap flag being set. In this case it is necessary for the software to determine which of the traps has priority and should be acted on.

Once set as the result of a trap, the WDC, WPT, and UAM flags in the TSR are latched and can only be cleared by disabling the associated trap(s). When one of these traps occurs, the trap handler must disable the trap by clearing the appropriate bit in the Task Select Register, to reset the flag, and then if necessary re-enable the trap. The SST flag only indicates that the SST trap is enabled and does not need to be cleared in this manner.

The three software interrupt instructions (SWI, SWI2, and SWI3) interact with all three of the attribute traps. If one of these instructions is executed in memory with its SST bit set, or if the stack overflows into a protected (UAM, WPT) segment, certain side effects occur which must be allowed for in the trap handlers.

When an SWI(2,3) causes an attribute trap, the SWI is processed through the special trap vector ($FFF0) instead of one of the normal SWI vectors. The IRQ generated by the trap is never processed. The effect is that an SWI which causes a trap is equivalent to a NOP that causes a trap. The only difference between traps caused by an SWI and traps caused by an SWI2 or 3 is that an SWI sets the IRQ and FIRQ masks and an SWI2 or 3 do not.

SECTION 3: WATCHDOG COUNTER

The prompt servicing of interrupts is vital to the overall performance of a multi-user/multi-tasking system. If a user were to mask the interrupts in the system, he could effectively keep the system from task switching and prevent other users from being serviced almost indefinitely. To prevent this, and to protect the system against the effects of certain undefined op-codes, the CPU III includes a Watchdog Counter (WDC). The WDC operates only when the system is in the User State and must also be enabled by setting the appropriate bit in the Task Select Register.

The WDC operates by monitoring interrupts and the processors response to them. When an interrupt occurs, the WDC begins counting machine cycles. If the count reaches 128 (or 32 as selected by JA-1) and the processor has not responded to the interrupt, a WDC trap occurs. When a WDC trap occurs the 6809 is reset. Unlike a normal (front panel) reset, only the 6809 is reset; other devices connected to the bus Reset line are not. A reset caused by a WDC trap is vectored through the special trap vector at $FFFF0 rather than the normal reset vector. Thus, a WDC trap can be handled by the same trap handler that processes attribute traps. A bit in the TSR is set to indicate the cause of the trap. See the preceding section on Memory Attributes for more information on trap processing.

Of the 256 possible 8-bit op-codes available to the 6809, only 223 are used, the other 33 are undefined. If one of these undefined or "illegal" op-codes is executed (usually because of a programming error), the results are generally unpredictable. However, the execution of certain undefined op-codes is known to cause the 6809 to stop normal processing, and not respond to interrupts. This condition can be detected by the Watchdog Counter. The WDC trap causes a reset because it is the only way to "interrupt" the 6809 after it has executed this type of illegal op-code.

When a WDC trap occurs, the task that caused the trap can not normally be recovered (the CPU register contents are lost during the reset). However, because only the processor is reset, the rest of the system remains intact and it is possible to resume normal operation of the remaining tasks.

NOTE: The Watchdog Counter only counts actual program execution cycles. The count is frozen during external DMA requests.

SECTION 4: TASK CONTROL

The overall operation of the CPU III is controlled through the Task Select/Task Status Register (TSR), addressed at $FE280, and the Fuse Register at $FFB00. The value stored in the TSR determines which of the 8 possible task maps is in use, and enables or disables the memory attribute traps, the Watchdog Counter (WDC) and the clock write-protect feature. Reads of the TSR return attribute flags and other system status information. The Fuse Register is used when switching from the Supervisor State to the User State. See the Task Control Register drawing on page 26 for the TSR bit definitions.

4-1: Task Switching

At any particular instant, the system is in one of two "states" as defined by the CPU III hardware. These states are called the "Supervisor State" and the "User State". In the Supervisor State, the system is always in task 0 (the task 0 address map is used). Task 0 normally contains the operating system executive software. The peripherals and registers on the CPU III are only accessible from the Supervisor State; unless jumper JA-5 is in the "All Tasks" position. Normally, access should be restricted to the Supervisor State for system security reasons.

In the User State, access to the on-board devices is restricted and other special conditions are in force. The system is always in the User State when in tasks 1-7. It may also be in the User State while in task 0.

On power-up or after a system (front-panel) reset, the system is in the Supervisor State. All bits in the TSR are cleared to 0 (zero). Usually, the operating system itself is the only program run in the Supervisor State. All other programs are executed as "tasks", in the User State. The operating system invokes a task by initializing one of the Task Maps in the DAT RAM; allocating the memory and other resources needed by the program to be run and assigning the appropriate memory attributes. Once the Task Map is initialized, the operating system can begin execution of the task as described below. Once started, execution of the task continues until an interrupt (hardware or software) occurs, causing the system to return to the Supervisor State. The operating system can then service the interrupt as required and either return to the task that was executing at the time of the interrupt or invoke a different task.

Switching to the User State and a different task is accomplished by first writing the desired task number to the TSR. Writing a task number to the TSR causes no immediate change in the state of the system, except that the "Not Task 0 Flag" in the TSR is set if a non-zero task number was written. The actual switch is accomplished by writing a value to the Fuse Register. On each subsequent machine cycle, the Fuse Register is incremented. When it reaches eight, the CPU III switches to the User State and the task map for the selected task number. Values from 0 to 7 may be written to the Fuse Register, so the delay before task switching may be from 1 to 8 machine cycles. The following section of code shows the normal sequence for switching to a user task. Note: the masking of the interrupts is critical to proper operation and must not be omitted.

```
lda     #%11010011      enable UAM,WPT,& WDC; task 3
orcc    #%01010000      mask IRQ and FIRQ
sta     TSR             setup TSR
andcc   #%10101111      unmask interrupts
lda     #4              setup delay; 8-4=4 cycles
sta     FUSE            start count
JMP     USERPGM         jump to user program (4 cycles)
```

This example transfers to the user program using a jump (JMP) instruction. The proper value for the Fuse Register depends on the transfer instruction used JMP, JSR, etc. The fuse register is determined by subtracting the number of machine cycles required to execute the instruction from 8 (eight). The remainder is the Fuse Register Count. For example, an LBRA takes 5 cycles, 8 - 5 = 3, the Fuse Register should be set to 3 for an LBRA.

Resuming execution of a program that was interrupted while in the User State is more complicated than simply transfering to a user program directly. The normal method of resuming an interrupted program is to execute an RTI instruction. However, in this case the registers were saved by the interrupt in the users memory (using the users task map) and the RTI will be executed in the Supervisor task (0). The task switch must be timed so that the RTI instruction is fetched from the task 0 memory and the stack is recovered from the user's memory. To accomplish this the Fuse Register must be loaded with a value of 6 (six) by the instruction immediately preceding the RTI, as shown in the following example. This value causes the proper sequence of events and the RTI will be executed properly. Note: While it would appear that the fuse register should be set to a value of 7, a close examination of the RTI instruction shows that an extra read (of the byte following the RTI instruction) is performed. To prevent this extra read from causing an attribute trap or other adverse effects, one cycle is subtracted from the fuse count to cause the extra read to occur before the task switch.

```
orcc    #%01010000      mask IRQ and FIRQ
sta     TSR
lds     USERSP          recover user stack pointer
lda     #6              setup delay; 2 cycles for RTI
sta     FUSE            start count
rti                     return to interrupted program
```

Notice that in the case of an RTI, it is not necessary to unmask the IRQ and FIRQ as in the previous example. The Condition Code register will be restored to it original (pre-interrupt) value by the RTI.

NOTE: the Fuse Counter only counts program execution cycles. When processing is suspended by external DMA requests, the Fuse Counter value is frozen to prevent premature expiration of the count.

The CPU III automatically switches from the User State to the Supervisor State and task 0 when any of the following occurs: IRQ, FIRQ, SWI, SWI2, SWI3, or any of the memory attribute traps. This is the only way that the system can be switched to the Supervisor State (with the exception of a system Reset). Note: an NMI (non-maskable interrupt) can not cause a switch to the Supervisor State because NMIs are masked by the hardware and are ignored when the system is in the User State.

The switch to the Supervisor State does not occur immediately. The system remains in the User State until the registers have been pushed on the users stack by the interrupt. The actual switch to the Supervisor State occurs just before the 6809 fetches the interrupt or trap vector. The vector fetch, and subsequently the interrupt or trap handlers occur in the Supervisor State, using the task 0 address map.

4-2: Memory Attribute Control

In addition to the bits in the DAT RAM that enable and disable memory attributes for individual memory segments, there are three bits in the TSR that enable and disable the attributes for an entire task. An attribute trap can only occur if the attribute is enabled in both the DAT RAM and the TSR, and the system is in the User State. The CPU III automatically disables all traps when in the Supervisor State.

To use the memory attribute traps, the appropriate bit(s) must be set in the DAT RAM when building the task map and the appropriate bit(s) must be set in the TSR when the task number is written.

4-3: Automatic Interrupt Masking

Under certain conditions the CPU III automatically masks (through hardware) all hardware interrupts (IRQ, FIRQ, and NMI). This hardware interrupt masking is in effect anytime the system is in the Supervisor State AND the task number in the TSR is a non-zero value. When the system switches to the User State, the IRQ and FIRQ masks are removed; however, the NMI remains masked at all times when in the User State. The only time the NMI is un-masked is when the system is in the Supervisor State AND the task number in the TSR is zero (0).

Referring to the previous software example; notice that before the RTI can be executed, the users stack pointer (which was saved by the interrupt handler) must be loaded into the the 6809's SP register. If an interrupt were to occur after the SP had been loaded but before the actual task switch occurred, the registers would be saved at the address currently in the SP, which is the address of the stack in the users task map, not the address of the stack in the current task map!

A similar situation occurs when returning to the Supervisor State from a user task. The SP register points to the stack area in the user task map but the system is using the supervisor task map. Until the users stack pointer is saved, and the stack pointer for the supervisor task is loaded into the SP register, interrupts can not be allowed.

To protect the system in these situations, the interrupts are masked by the hardware as described above. The hardware masking is removed automatically when the system switches to the User State but must be removed manually, by setting the task number to 0 in the TSR, when switching to the Supervisor State.

CAUTION: In spite of this automatic protection, a possibility exists for an interrupt to be recognized by the processor if it occurs

at the same time that the task number is being written to the TSR. For this reason it is necessary, as shown in the sample code, to mask the IRQ and FIRQ in the processor before the task number is written.

SECTION 5: ON-BOARD DMA CONTROLLER

In many applications the processor is required to spend a great deal of time moving data from one area of memory to another. The CPU III includes a Direct Memory Access Controller (DMAC) that is capable of moving data between memory locations at 1 byte every 2 machine cycles (1 byte/microsecond at 2 MHz.) In addition to speed, the DMAC offers another advantage in its ability to directly manipulate the DAT task maps. The addresses output by the DMAC are translated by the DAT just as addresses output by the CPU. However, the task map is not selected by the TSR during DMA transfers. Instead, control bits in the DMAC registers select the task map(s) to be used. The source and destination for the DMA transfer can be in different task maps. In this way data can be directly transferred from memory in one task map to memory in another map.

5-1: DMAC Programming

The DMAC is controlled by five write-only registers at $FFC00. (See the DMA control register drawing on page 27.) These registers are: the Source Control Register (SCR), the Source Address Register (SAR), the Destination Control Register (DCR), the Destination Address Register, and the Byte Count Register. The address and byte count registers are two-byte (16 bit) registers, the control registers are single-byte registers. The SCR and DCR are cleared by power-on or front-panel Reset. Before a DMA transfer can take place these registers must be initialized.

***** CAUTION *****

Once initialization of the DMAC registers is begun, it should not be interrupted, if the interrupting process may also use the DMAC. If the registers are partially initialized and the process is interrupted, the registers that are already initialized may be changed by the interrupting process and not contain the correct values when the original process resumes. For this reason, interrupts must be masked while the DMAC registers are being initialized.

The SCR, along with the SAR, determines the source address of the data to be transferred and the type of transfer. The least significant three bits (b0,b1,b2) determine the Task Map to be used as the source of the data. The most significant bit (b7) determines whether the source address (in the SAR) will change or remain fixed after each byte is transferred. If b7=0 (zero) the address will change after each byte to allow blocks of data to be transferred. If b7=1 (one) the address will remain fixed to allow multiple bytes to be read sequentially from a single address, such as the data port on an I/O board (e.g. GMX intelligent serial and parallel I/O boards). If

b7=0 (address changes) then b5 is used to determine whether the address increments (b5=0) or decrements (b5=1) after each byte is transferred; allowing blocks of data to be transferred in either forward or reverse order. If b7=1, then b5 has no effect on the transfer. Bits 3,4, and 6 of the SCR are not used.

The SAR is a 16-bit register that sets the initial logical address for the source of the data to be transferred. Depending on the contents of the SCR, the SAR will be incremented, decremented, or remain unchanged during the transfer. Note: the SAR contains the logical address of the data, which is modified by the DAT and the task map selected in the SCR to form the physical address.

The BCR determines the number of bytes that will be transferred. It is a 16-bit register and any number of bytes from 1 to 65535 may be moved by a single DMAC operation. The BCR is decremented each time a byte is transferred and, when the last byte has been moved, the DMA transfer ends.

The DCR and DAR determine the destination for the DMA transfer, and are functionally equivalent to the corresponding source registers. The DCR also contains an additional bit (b6) that starts the DMAC. Like the source address, the destination address may be incremented, decremented, or remain fixed during the transfer. The source and destination addresses are totally independent of each other. One can remain fixed while the other increments, both may be incremented, etc.

Writing a 1 (one) to bit 6 of the DCR starts the DMAC. Once started, the DMAC halts the 6809 and performs the transfer specified by the DMAC register contents. The other DMAC registers (and if necessary the appropriate task maps in the DAT) must be initialized before the transfer is started!

Due to the manner in which the 6809 processes the "halt" request from the DMAC, the instruction following a write to the DCR, that starts the DMAC, is executed before the transfer takes place. The instruction that follows the write to the DCR must not modify the contents of the DMAC registers or the data being transferred. It is recommended that a NOP (no operation) instruction immediately follow the write to the DCR. The following code illustrates a typical DMA transfer of the type that could be used to move a block of data into the FIFO on one of the GMX intelligent I/O interfaces.

```
DMAMOVE     orcc    #%01010000   mask interrupts
            ldd     #SRCADDR
            std     SAR          set up source address
            ldd     #DESTADR
            std     DAR          set up dest. address
            ldd     MOVESIZE
            std     BCR          set up byte count
            lda     #%00000010   increment address, task 2
            sta     SCR          set up source address
            lda     #%11000000   single address, task 0
            sta     DCR          start the DMAC
            nop
* The processor is halted and the DMAC starts here
```

The DMAC can be used to quickly move an entire task map into the DAT RAM, as illustrated in the following example.

```
DMAMOVE     orcc    #%01010000   mask interrupts
            ldd     #NEWMAP      point to the new map
            std     SAR          set up source address
            ldd     #DAT+4*64    point to the DAT RAM
            std     DAR          area for task 4
            ldd     #64
            std     BCR          64 bytes per map
            lda     #%00000111   increment address, task 7
            sta     SCR          set up source address
            lda     #%01000000   increment address, task 0
            sta     DCR          start the DMAC
            nop
            ...
```

CAUTION: be sure that the task being loaded with a new map is NOT THE SAME as the source task.

These examples illustrate two of the many uses for the DMAC. Almost any code that moves data from place to place in memory can be replaced by the DMAC. Because of the overhead in setting up the DMAC registers, moves involving only a few bytes, are less efficient than the same move done by the processor; except in cases where the move is between two separate task maps.

NOTE: when the 6809 does an indexed read or write in the auto-decrement mode, the index register is decremented BEFORE the read or write. The DMAC, in the decrement address mode, decrements the address AFTER the read or write. When replacing code that moves data using auto-decrement with the DMAC this difference must be taken into account.

5-2: DMA Contention

If an external DMA device, such as a disk controller, makes a DMA request while the DMAC is in operation, the CPU III arbitrates the resulting DMA contention as follows. If the external device uses cycle-steal DMA (e.g. the GIMIX DMA disk controller), the external device takes priority. The DMAC will stop for three cycles each time the external device transfers one byte. However, the DMAC has priority over external devices using halt/burst DMA. Any external device using halt/burst is forced to wait until the DMAC is finished before it is given control of the bus.

SECTION 6: ON-BOARD EPROM/ROM

The CPU III has a single 24/28 pin socket that will accept a 2Kx8, 4Kx8, or 8Kx8 EPROM/ROM. The socket occupies the upper 4K of the 1 megabyte address space $FF000 through $FFFFF. This address space is shared with the DAT RAM and the CPU III control registers (see the memory map on page 24). Normally the ROM is only accessible when the system is in the "Supervisor State"; unless JA-5 is in the "all tasks" position. When the system is first powered up or after a Reset, the upper 2K of the ROM is the only available memory; until the DAT has been initialized. (see the description of the power-up state on page 5) The upper 2K of this ROM must contain the restart and interrupt vectors as well as initialization routines for the DAT.

6-1: EPROM Size Selection

Jumper area JA-14 must be set to correspond to the size of the ROM installed on the board. JA-14 has two positions: one position configures the socket for 2K devices, the other configures it for 4K or 8K devices. NOTE: When a 2K ROM is used it appears twice in the 4K address space allocated to the rom socket ($FF800-$FFBFF and $FFC00-$FFFFF).

6-2: Software Switching with 8K ROMs

Since the ROM socket occupies a maximum of 4K of address space, only half of an 8K device is available at any one time. The user has the option of choosing between the upper or lower half of a 8K device, under hardware (jumper) and/or software control (see the description of JA-8 on page 2). This allows switching between two operating systems, two versions of a program, etc. When software switching is enabled by JA-8, one half of the ROM will always be selected on reset or power-up. The other half of the ROM is selected by writing to address $FE281 while the system is in the Supervisor State.

***** CAUTION *****

Writing to this address causes the ROM to switch immediately. The code that performs the switch normally must reside outside the address range of the ROM.

Subsequent writes to $FE281 cause the ROM to switch back and forth between the upper and lower halves. NOTE: the data written to this address is unimportant. Any write to this address causes the switch.

SECTION 7: TIME-OF-DAY CLOCK

Time-Of-Day clock and calendar functions are provided by a 146818 Real-Time Clock (RTC). Once set, the RTC provides accurate time and date information that is maintained by a rechargeable battery supply when the system power is removed. The RTC is also capable of providing the periodic interrupts required for "time-slicing" in multi-user/multi-tasking systems.

The RTC has many features and functions, which are described in detail in the 146818 data sheet. The CPU III hardware allows full access to the RTC; however, the implementation of certain features will depend on the particular software used.

***** CAUTION *****

The RTC and its associated circuitry, as well as the 2K scratchpad RAM, are powered anytime the battery jumper (JA-7) is in the ON position. To prevent possible damage, the battery must be turned OFF when replacing components in these circuits.

7-1: RTC Addressing

The RTC occupies 64 bytes of address space from $FE240 to $FE27F. The lower 14 bytes are the clock data and control registers (see the the 146818 data sheet). The remaining 50 bytes are general purpose RAM, which may be used as scratchpad storage, etc. Normally the RTC is only accessible when the system is in the Supervisor State. When in the User State, the RTC is not accessible unless JA-5 is in the "all-tasks" position. See the 146818 data sheet for a functional description of the clock registers.

7-2: RTC Write-protection

To protect against tampering or unintentional modification of the RTC registers and the 50 bytes of RAM in the RTC, the CPU III includes a write-protection option for the RTC. When write-protection is enabled (see JA-13), all writes to the RTC are inhibited, unless the write enable bit in the Task Select Register (TSR) is set (1). This bit must be set (1) before writing to the RTC, and should be cleared (0) after, to enable the write-protection.

7-3: RTC Time-base

The time-base for the RTC is provided by a separate, crystal controlled, 32.768 KHz. oscillator. User written software that initializes the RTC must take this into account when initializing the "DV" bits (b6,5 & 4) in Register A.

The RTC time-base oscillator is factory adjusted and should not normally require readjustment unless one of the parts in the oscillator circuit is replaced. If the RTC does not keep accurate time, the oscillator may be checked and adjusted (if necessary) by monitoring the oscillator output at the 32.768 KHz. test point on JA-10, and adjusting C-17 to obtain the correct frequency (32.768 KHz.). NOTE: The accuracy of the clock depends on the accuracy of the frequency counter used to adjust the oscillator. Slight trial-and-error adjustment may be required to achieve the desired accuracy.

7-4: RTC Reset Options

The CPU III provides two options for generating a reset signal to the RTC. A reset only affects certain interrupt and alarm functions; described in the 146818 data sheet. Resetting the RTC does not affect the time-keeping and calendar functions or the RAM. JA-11 selects either the "normal reset" mode, or the "reset on battery fail" mode. In the normal mode the RTC is reset each time a front-panel or power-on reset occurs in the system. This clears the interrupt flags so that the RTC does not generate unwanted interrupts. The "reset on battery fail" mode resets the RTC if the input voltage to the clock falls below the level at which it will operate reliably. This mode only resets the RTC if the battery discharges below a set level (while the clock is being powered by the battery) or if system power is removed with the battery disconnected. This mode is for special applications where it is desirable to maintain interrupt or alarm settings while the system is powered down.

7-5: RTC Power-fail Indication

One of the features of the RTC is a power-fail flag that indicates power to the clock was lost and its contents are no longer valid. This flag (the VRT bit in register D) is cleared when power is removed from the device. When power is first applied to the RTC or restored after a power failure, the first read of the register returns this bit as 0, indicating that the power was off; subsequent reads return a 1.

7-6: RTC Interrupt Selection

One half of jumper area JA-4 is used to select the type of interrupt that the RTC will cause when it is programmed to generate interrupts. The interrupt output of the RTC can be connected to any one of the three interrupt lines (IRQ, FIRQ, or NMI) by JA-4.

SECTION 8: 6840 PROGRAMMABLE TIMER MODULE (PTM)

The PTM has three 16-bit binary counters that may be used in a variety of applications. The 6840 data sheet provides information on the application and programming of the PTM.

8-1: PTM Addressing

The 6840 requires 8 bytes of address space for its control registers and latches. These 8 bytes are mapped into the address space twice; from $FE210 to $FE217, and again from $FE218 to $FE21F. Either set of addresses can be used to access the PTM. Like the other memory mapped devices on the CPU III, the PTM is only accessible while the system is in the Supervisor State; unless JA-5 is in the "all tasks" position.

8-2: PTM Time Base

The 6840 has provisions for using either the internal time base (the processor "E" clock) or an external clock source. While the internal time base is adequate for some applications, it is subject to inaccuracies cause by MRDY cycles which stretch the processor clock. For applications that require a more accurate time base, the CPU III includes a separate oscillator (Y3), that provides a 500 KHz. (.0025%) time base. It is a self-contained unit, in a DIP-package, that may be replace by the user to obtain a other frequencies. NOTE: Due to restrictions imposed by the 68B40 when using an asynchronous external time base, the MAXIMUM allowable frequency for the external time base is 750 KHz. The specifications for Y3 are as follows:

Package	14-pin DIP
Power	+5 VDC (pin 14)
	GND (pin 7)
Output	TTL level Square Wave (pin 8)
Frequency	500 KHz. Std. (750 KHz. Maximum)
Tolerance	0.0025% Std. or as required

8-3: PTM External Connections

Jumper area JA-2 provides access to the clock and gate inputs and the timer outputs for all three counters, and the 500 KHz time base oscillator output. The counters and the time base may be interconnected as required using the jumpers provided or by wire-wrapping directly to JA-2. When connection to off-board devices is required, a user supplied cable with the appropriate 20 pin transition connector may be connected to JA-2.

8-4: PTM Interrupt Selection

One half of jumper area JA-4 is used to select the type of interrupt that the PTM will cause if it is programmed to generate interrupts. The interrupt output of the PTM can be connected to any one of the three interrupt lines (IRQ, FIRQ, or NMI) by JA-4.

SECTION 9: SCRATCHPAD RAM

The CPU III includes a 2Kx8 static RAM that may be used for scratchpad storage. When the battery ON/OFF jumper (JA-7) is in the ON position, data in the scratchpad RAM is maintained, during power down, by the same rechargeable battery that maintains the Real-Time Clock.

9-1: Scratchpad RAM Addressing

The scratchpad RAM can be configured as either 1K or 2K of memory by jumper area JA-12. When JA-12 is in the 2K position, the entire RAM is accessible at addresses $FE400 to $FEBFF. When JA-12 is in the 1K position, only the lower half ($FE400-$FE7FF) of the RAM is accessible. Normally, the scratchpad RAM is only accessible when the system is in the Supervisor State. In the User State the RAM is not available unless JA-5 is in the "all tasks" position.

SECTION 10: MISSING CYCLE DETECTOR INTERFACE

A GIMIX Missing Cycle Detector (MCD) board can be connected to the CPU III at P2. The connector provides power (8 Volts-unregulated) for the MCD board and inputs for the MCD interrupt and status outputs. An MCD board can be used to monitor the AC power input to the system and generate an interrupt if the power fails (see the documentation supplied with the MCD board). The MCD connector can be used in other applications that require generation of interrupts and/or a status bit that can be read by the system, provided that the input signals are properly conditioned.

10-1: MCD Interrupt Selection

The type of interrupt caused by the MCD is selected by jumper area JA-3. The interrupt output from the MCD can be connected to any one of the three interrupt lines (IRQ, FIRQ, or NMI).

10-2: MCD Status Bit

A separate output from the MCD is used as a status bit. This bit would normally be tested by the interrupt polling routines to determine if the MCD was the source of an interrupt. The status bit

input from the MCD connector appears at bit 1 of the Task Status Register ($FE280). When the status bit input is at a TTL low level, bit 1 of the TSR reads 0 (zero); if it is high, bit 1 reads 1 (one).

10-3: MCD Connector, Input Requirements

The interrupt input at P2 may be connected to any open-collector (or open-drain) output that is suitable for direct connection to the system interrupt bus. The status bit input may be connected to a TTL level signal. The input is non-latching and the signal must remain stable while the TSR is being read to guarantee valid data.

SECTION 11: FRONT PANEL RESET/NMI CONNECTOR (P1)

The front panel Reset/Abort(NMI) switch on GIMIX mainframes connects to the CPU III at P1. P1 provides separate inputs for Reset and NMI (Non-Maskable Interrupt), and circuit ground. To generate a Reset or NMI the appropriate input must be grounded. The inputs are conditioned on the CPU III so that a simple, normally open switch can be used.

SECTION 12: USER-DEFINED STATUS BIT

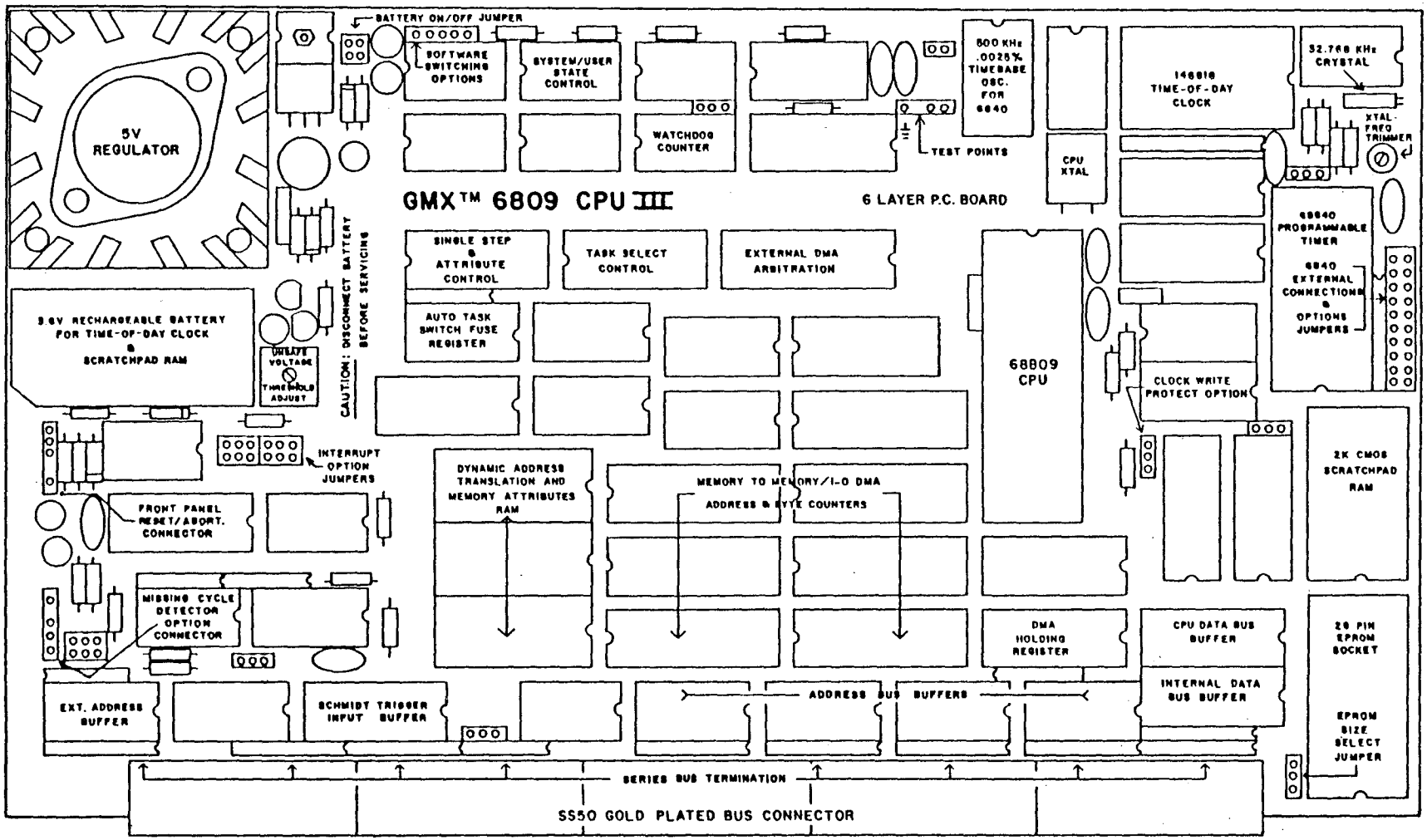
The Task Status Register ($FE280/read) contains an unused bit that can be user defined. The status of this bit (TSR bit 4) is set by jumper area JA-9. If a jumper is installed at JA-9, bit 4 of the TSR will be a 0 (zero); if the jumper is removed, bit 4 = 1 (one). This bit may also be used to monitor an external TTL level signal. The input at JA-9 is non-latching and the signal must remain stable while the TSR is being read to guarantee valid data.

SECTION 13: UNSAFE BUS VOLTAGE DETECTION

To prevent accidental modification of the data in the Clock and the Battery Back-up RAM, during the transition between system and battery power, the CPU III includes a circuit that monitors the +8V supply to the board. When the bus voltage falls below the level required for proper operation of the board, the 6809 is held in Reset and all accesses to the Time-Of-Day clock and the 2K Scratchpad RAM are inhibited.

13-1: Voltage Threshold Adjustment

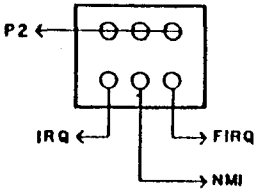
The Unsafe Voltage Threshold is factory set and should not require adjustment in the field. The threshold is set (by R26) to a point just above the minimum voltage required by the on-board voltage regulator (approx. 7 volts). The "Power Fail" test point at JA-10 is used during checkout and adjustment of the circuit.



FUNCTION LAYOUT

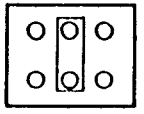
JA 3

FIG. A



MCD INTERRUPT SELECT

FIG. B

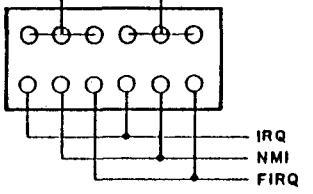


NMI SELECTED

JA 4

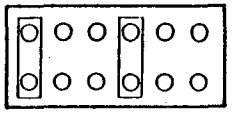
6840 146818

FIG. C



TIMER & CLOCK INTERRUPT SELECT

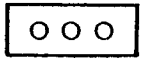
FIG. D



6840-IRQ SELECTED
146818-IRQ SELECTED

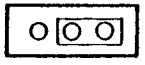
JA 5

FIG. E



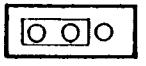
ACCESS SELECT FOR ON BOARD MEMORY MAPPED DEVICES

FIG. F



TASK Ø/SUPERVISOR STATE ONLY

FIG. G



ALL TASKS

JA 6

FIG. H

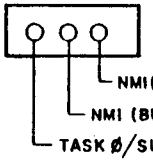
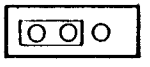
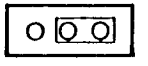


FIG. I



STATE INDICATION to BUS

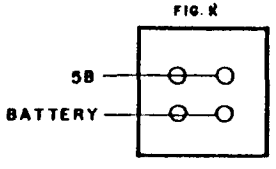
FIG. J



NMI to BUS

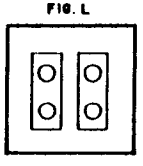
JA 7

FIG. K



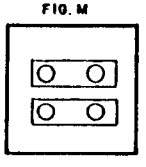
BATTERY ON/OFF JUMPER

FIG. L



BATTERY ON

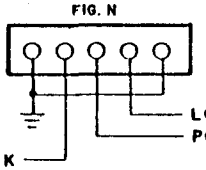
FIG. M



BATTERY OFF

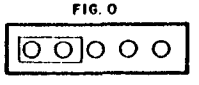
JA 8

FIG. N



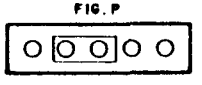
8K EPROM 4K BLOCK SELECT FOR SOFTWARE

FIG. O



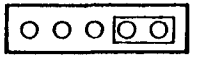
UPPER 4K
ALWAYS SELECTED
SWITCHING DISABLED

FIG. P



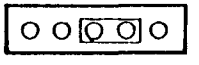
UPPER 4K
SELECTED ON RESET
SWITCHING ENABLED

FIG. Q



LOWER 4K
ALWAYS SELECTED
SWITCHING DISABLED

FIG. R



LOWER 4K
SELECTED ON RESET
SWITCHING ENABLED

JA 9

FIG. S

NO JUMPER
B4=1

FIG. T

JUMPER
B4=Ø

SENSE BIT SELECT JUMPER
TASK STATUS REGISTER BIT 4

GIMIX INC.
1337 W. 37th PL, CHICAGO, IL, 60609

3-8-83	JUMPER OPTIONS SHEET 1	

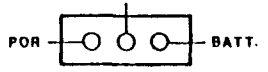
GMX™ 6809 CPU III

C-24-00668-1

JA11

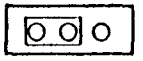
FIG U

146818 RST



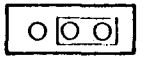
TIME-OF-DAY CLOCK
RESET OPTIONS

FIG.V



NORMAL RESET

FIG.W

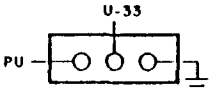


RESET ON
BATTERY FAIL

JA12

FIG.X

U-33



SCRATCHPAD RAM
SIZE SELECT

FIG.Y

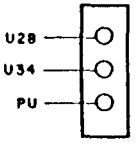
2 K
$E400-$EBFF

FIG.Z

1 K
$E400-$E7FF

JA13

FIG.AA

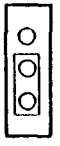


TIME-OF-DAY CLOCK
WRITE PROTECTION ENABLE

FIG.BB

WRITE PROTECTION
ENABLED

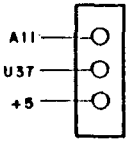
FIG.CC



WRITE PROTECTION
DISABLED

JA14

FIG.DD

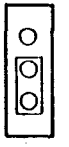


EPROM SOCKET (U-37)
SIZE SELECT

FIG.EE

4/8K
EPROMS

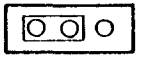
FIG.FF



2K
EPROMS

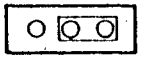
JA1

FIG.GG



32 CYCLES

FIG.HH

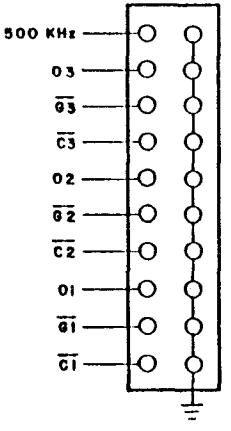


128 CYCLES

WATCHDOG COUNTER
DELAY COUNT OPTION

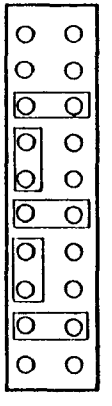
JA2

FIG.II



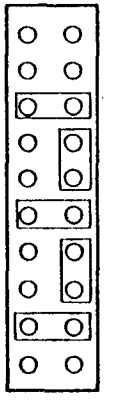
6840 PTM
EXTERNAL CONNECTIONS
& JUMPERS

FIG.JJ



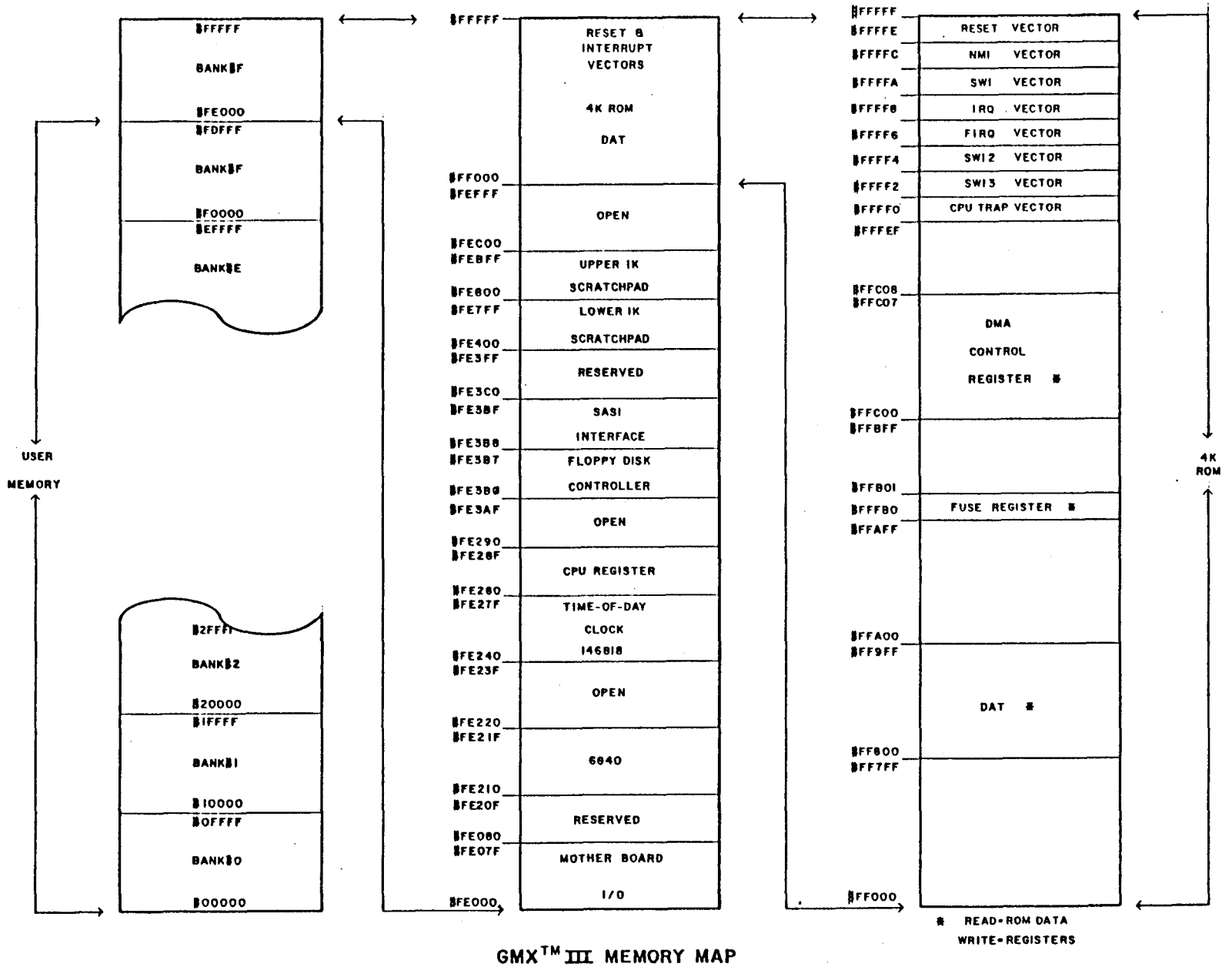
CASCADED
1—2—3

FIG.KK

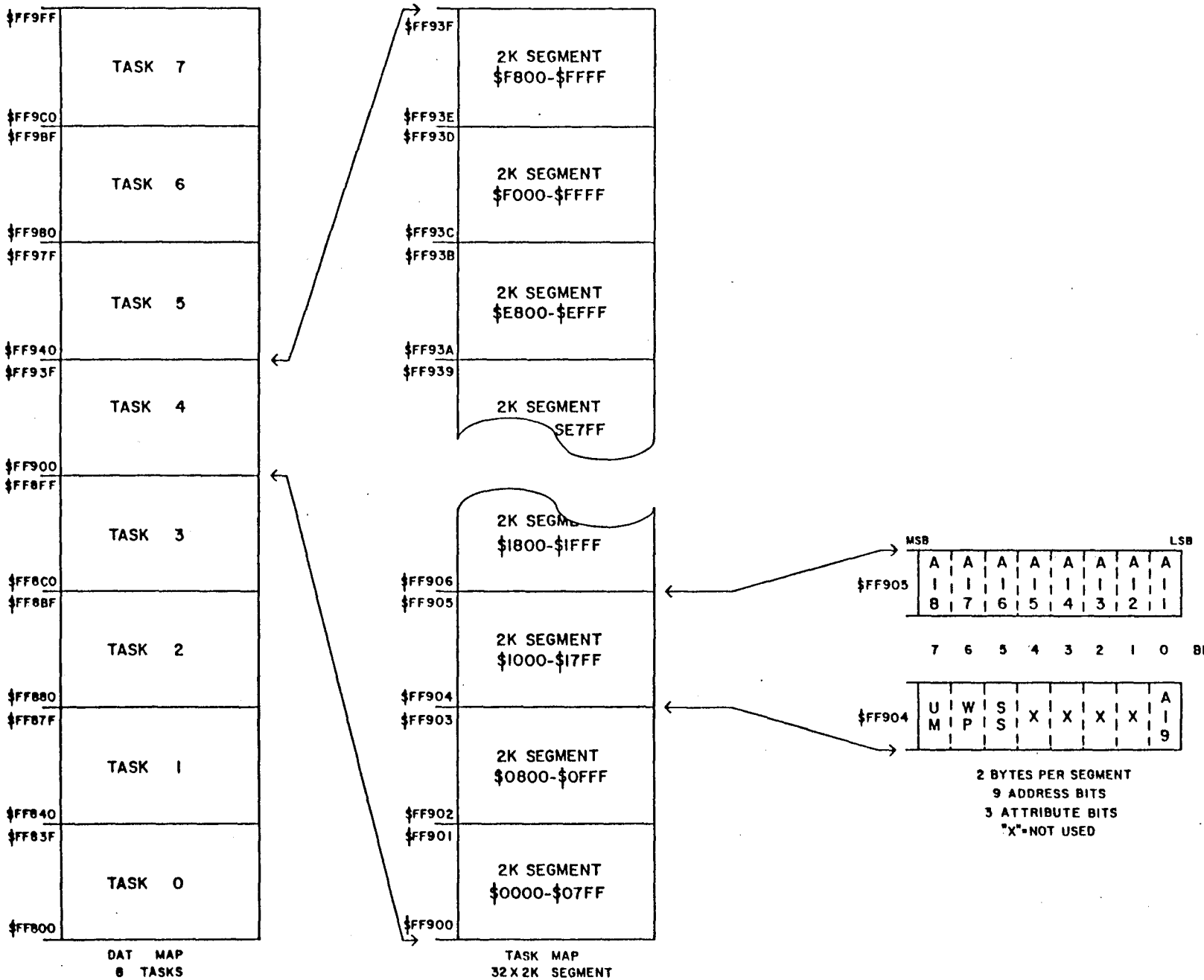


CONTINUOUS
OPERATION

GIMIX INC. 1337 W. 37th PL, CHICAGO, IL. 60609		
3-10-83	JUMPER OPTIONS SHEET 2	
GMX 6809 CPU III		
PROPRIETARY MAT'L. ALL RIGHTS RESERVED © 1983 GIMIX INC.		C-24-0066B-2



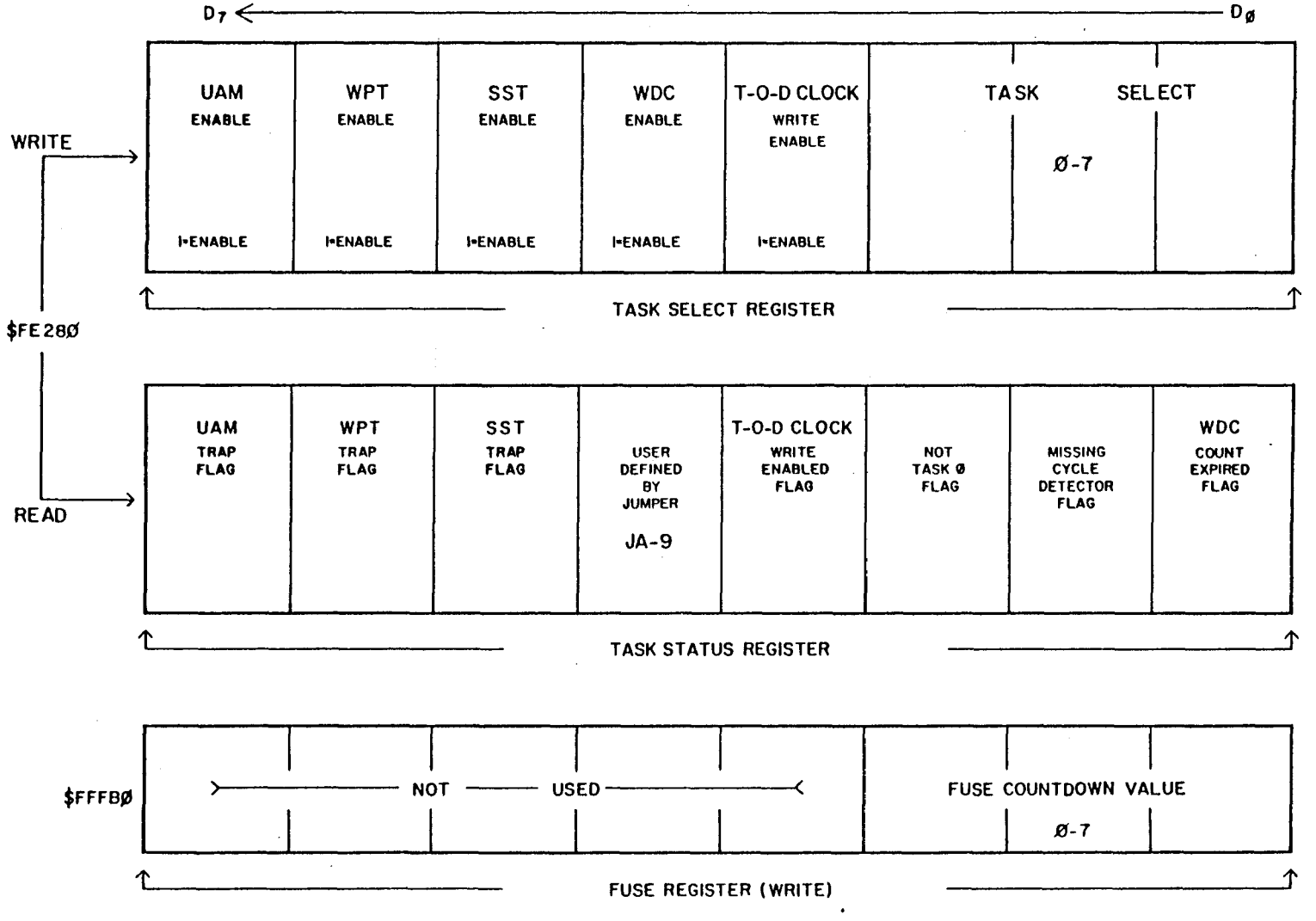
GMX™ III MEMORY MAP



GMX™ 6809 CPU III DYNAMIC ADDRESS TRANSLATOR

GMX™ 6809 CPU III

TASK CONTROL REGISTERS

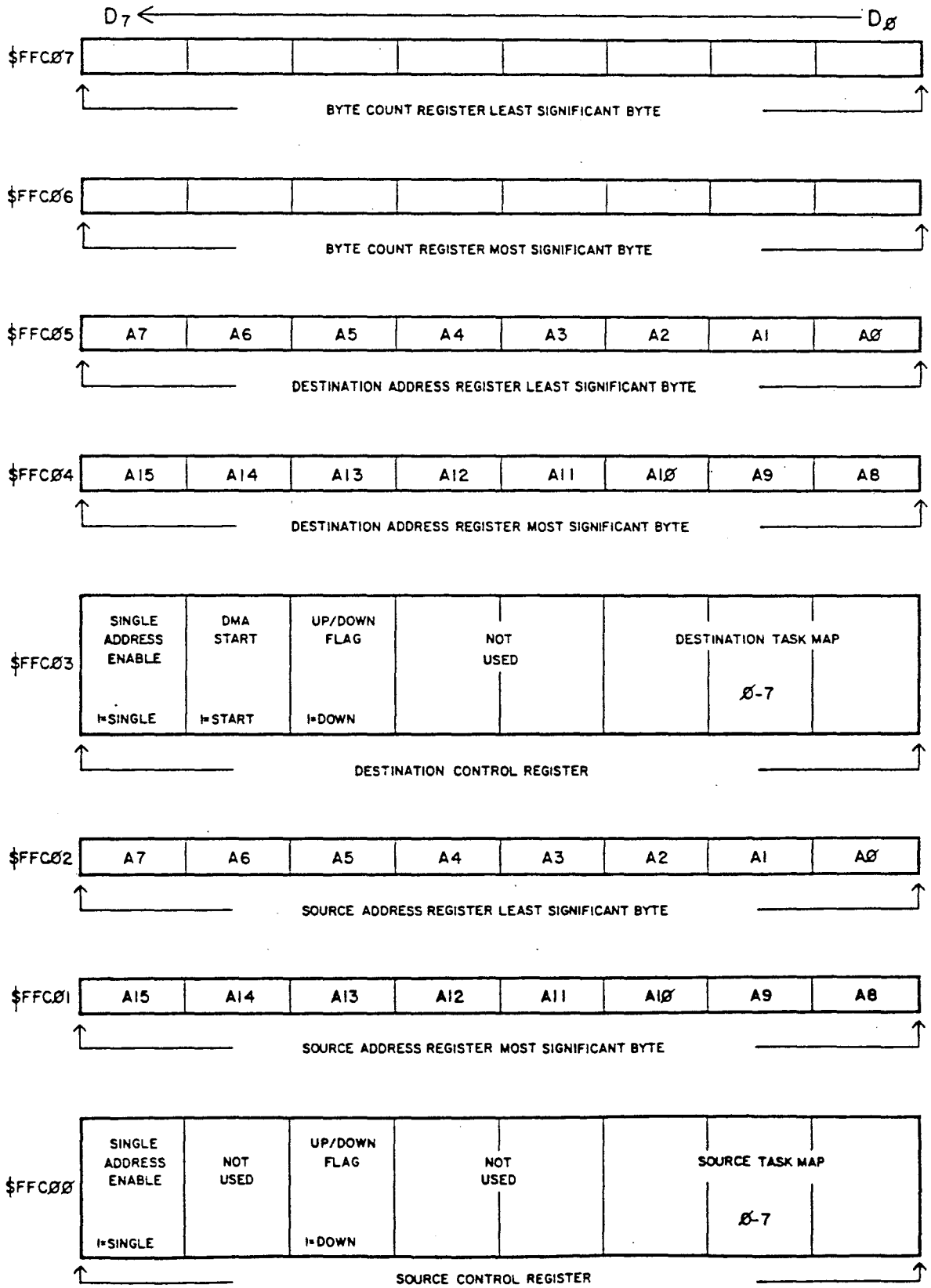


NOTE: READS OF THE FUSE REGISTER RETURN DATA FROM THE ROM AT THAT ADDRESS

DRAWN BY: CLARK P

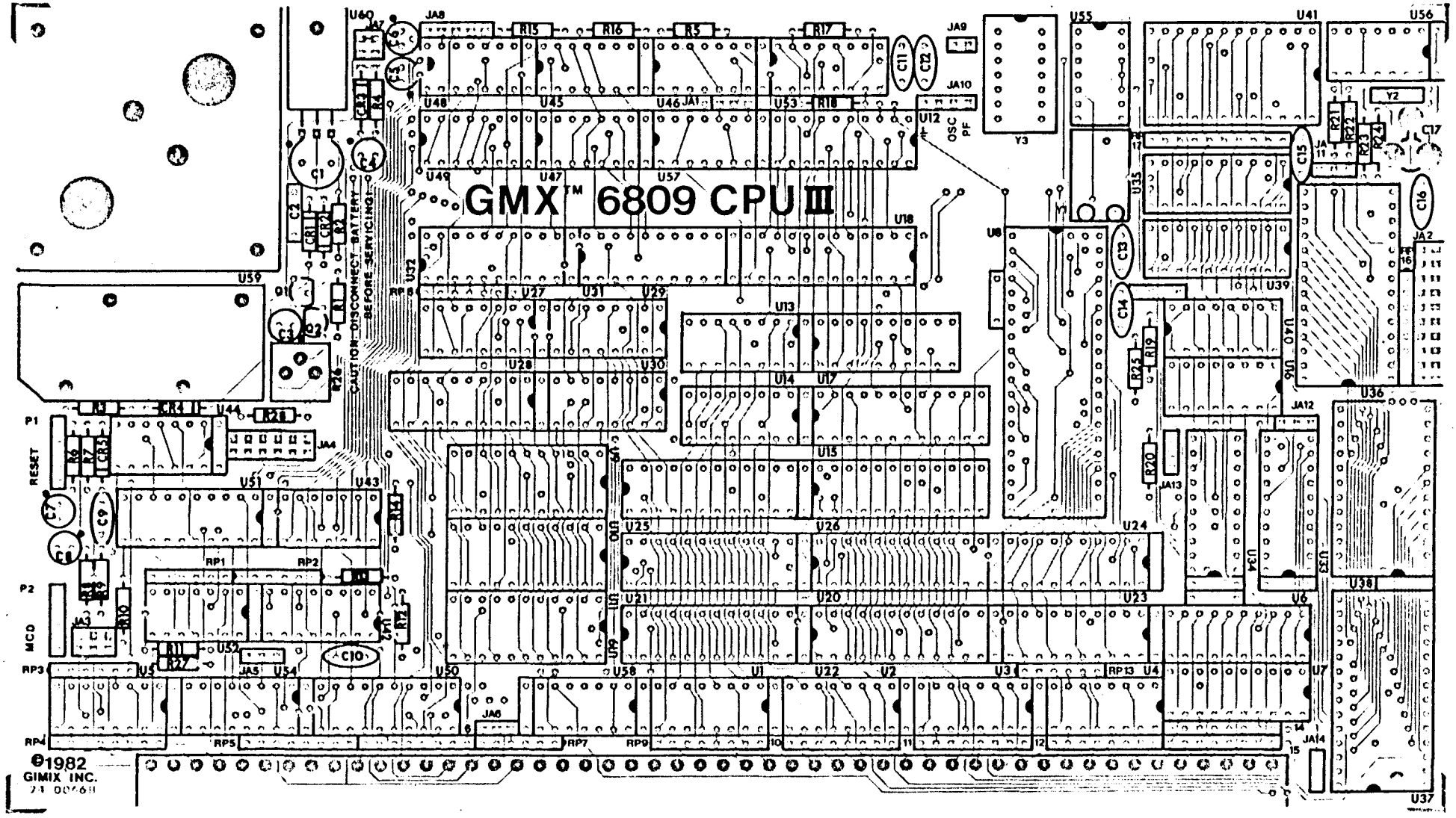
GMX™ 6809 CPU III

DMA CONTROL REGISTERS

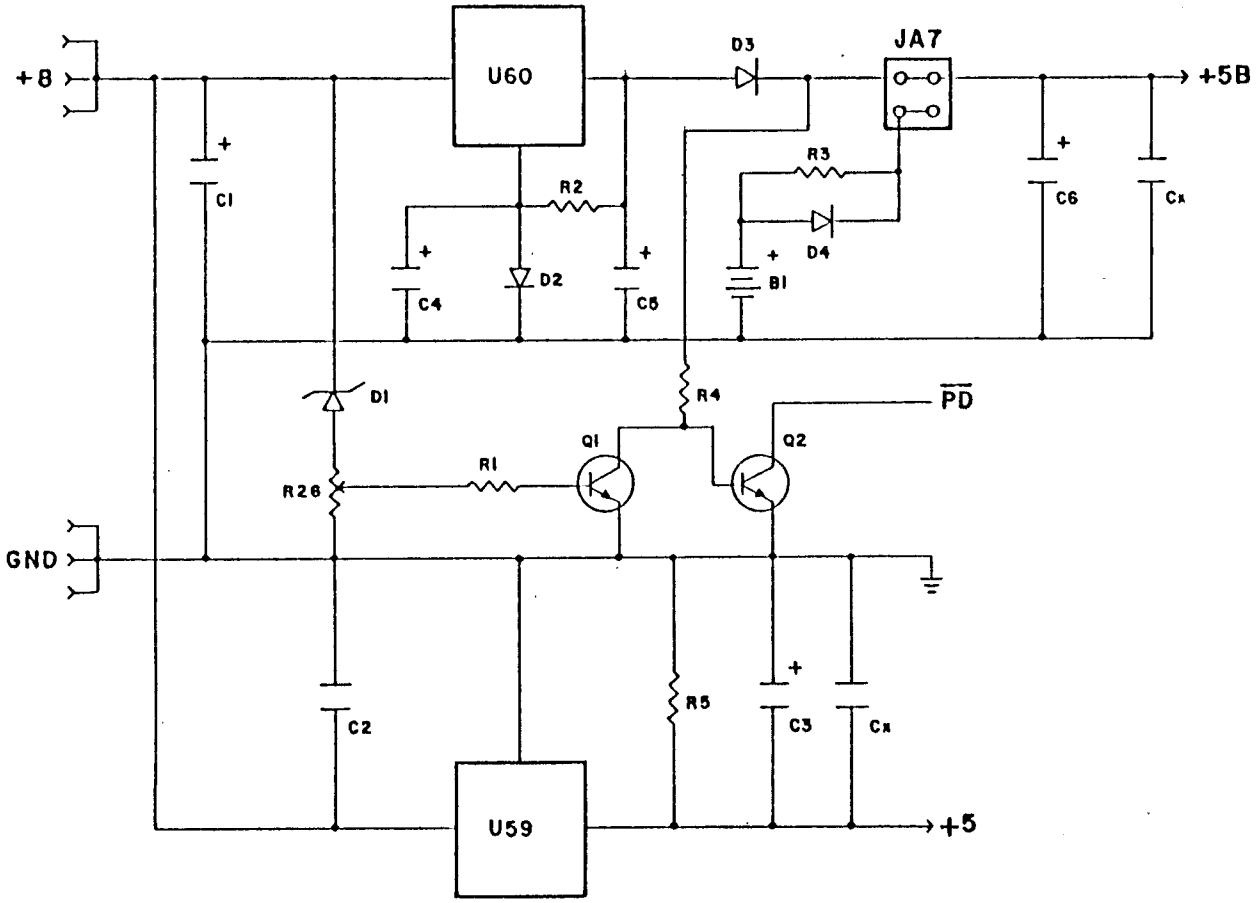


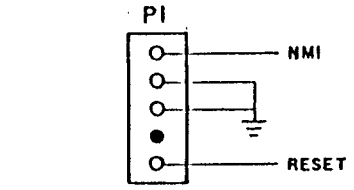
NOTE: ALL REGISTERS ARE WRITE ONLY. READS FROM THESE LOCATIONS RETURN DATA FROM THE 4K ROM SPACE.

DRAWN BY: CLARK P.

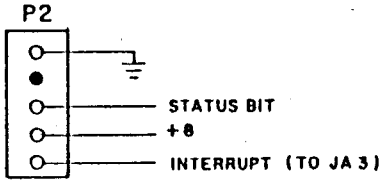


COMPONENT LAYOUT

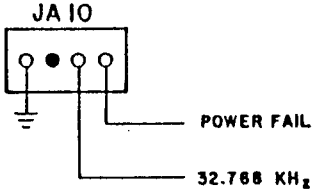




FRONT PANEL RESET/NMI

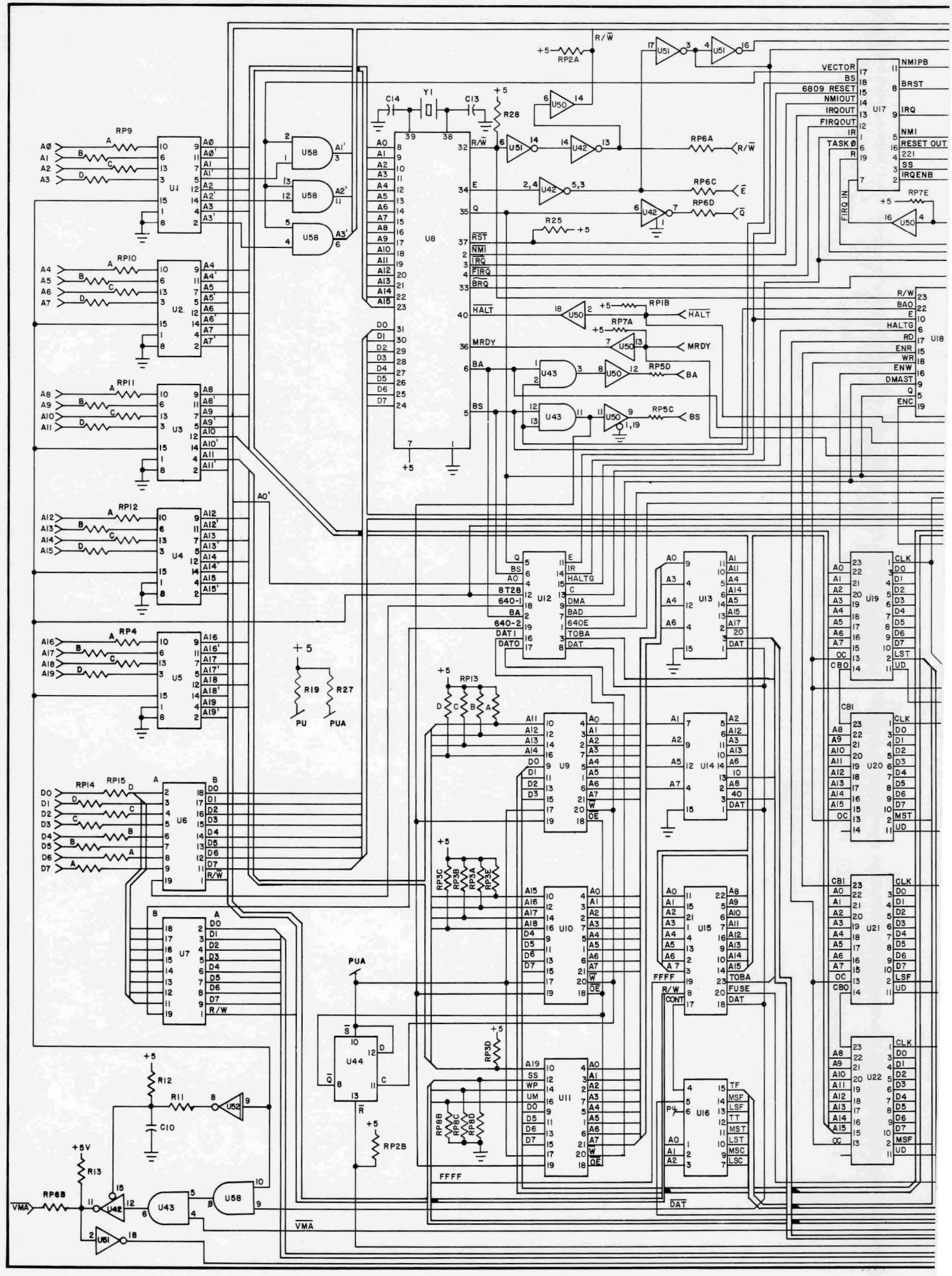


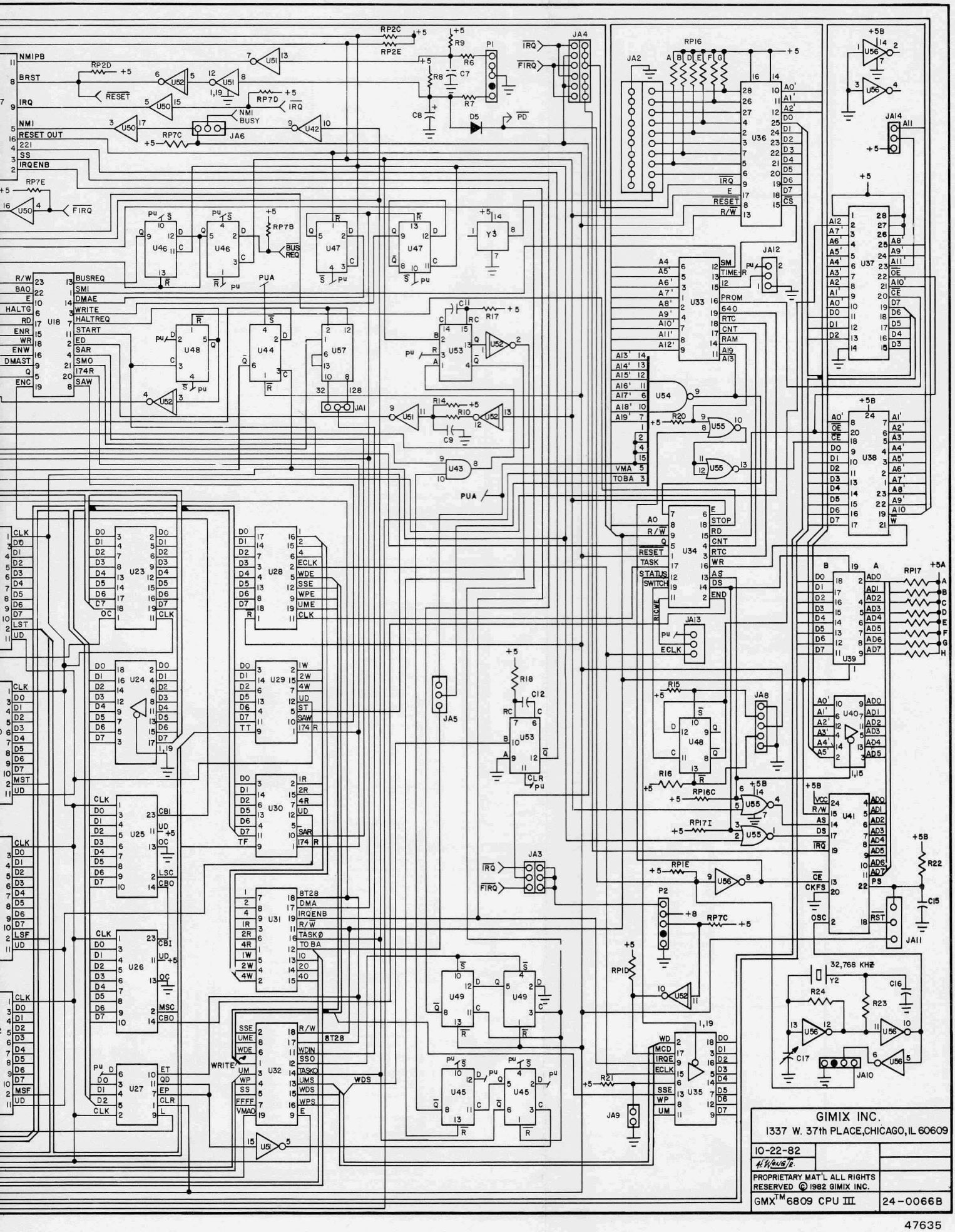
MISSING CYCLE DETECTOR



TEST POINTS

GIMIX INC. 1337 W. 37th PLACE, CHICAGO, IL. 60609		
2-16-83	BATTERY POWER SUPPLY	
GMX™ 6809 CPU III		
PROPRIETARY MAT'L. ALL RIGHTS RESERVED © 1983 GIMIX INC.		24-0066


RP9
RP10
RP11
RP12
RP4
RP14
RP15
U1
U2
U3
U4
U5
U6
U7
U8
U9
U10
U11
U12
U13
U14
U15
U16
U17
U18
U19
U20
U21
U22
U42
U43
U44
U50
U51
U52
U58
Y1
C13
C14
C10
R11
R12
R13
R19
R25
R27
R28
RP2A
RP3C
RP3D
RP5C
RP5D
RP6A
RP6C
RP6D
RP7A
RP7E
RP8B
RP8C
RP8D
RP8E
RP13
RPIB
RP2B
R/W
HALT
MRDY
BA
BS
E
Q
RST
NMI
IRQ
FIRQ
BRQ
PUA
VMA
FFFF
DAT
VECTOR
6809 RESET
NMIOUT
IRQOUT
FIRQOUT
TASK Ø
FIRQ IN
NMIPB
BRST
IRQ
NMI
RESET OUT
SS
IRQENB
R/W
BAO
HALTG
RD
ENR
WR
ENW
DMAST
ENC


NMIPB
BRST
IRQ
NMI
RESET OUT
221
SS
IRQENB
RP7E
FIRQ
RP2D
RESET
RP7D
NMI BUSY
JA6
RP7C
RP2C
RP2E
R9
R6
R8
C7
R7
C8
P1
D5
PD
IRQ
FIRQ
JA4
JA2
RP16
U36
RESET
R/W
JA12
U33
PROM
640
RTC
CNT
RAM
U37
U38
U54
U55
R20
VMA
TOBA
U34
STOP
RD
CNT
RTC
WR
AS
DS
END
STATUS
SWITCH
RIGWE
JA13
ECLK
U39
RPI7
U40
U41
R22
C15
JA11
P2
RP7C
RPID
32,768 KHZ
Y2
C16
R24
R23
C17
JA10
R21
JA9
U35
WD
MCD
IRQ
ECLK
SSE
WP
UM
U18
BUSREQ
SMI
DMAE
WRITE
HALTREQ
START
ED
SAR
SMO
174R
SAW
R/W
BAO
E
HALTG
RD
ENR
WR
ENW
DMAST
Q
ENC
RP7B
BUS REQ
U46
U47
U48
U44
U57
Y3
PUA
JA1
R17
R14
R10
C9
C11
U53
U43
U51
U52
U23
U24
U25
U26
U27
U28
U29
U30
U31
U32
U45
U49
R18
C12
JA5
JA3
JA8
R15
R16
RP16C
RPI7I
RPIE
WRITE
WDS
8T28
GIMIX INC.
1337 W. 37th PLACE, CHICAGO, IL 60609
10-22-82
A. WONG JR.
PROPRIETARY MAT'L ALL RIGHTS RESERVED © 1982 GIMIX INC.
GMX™ 6809 CPU III
24-0066B

47635

GMX™ 6809 CPU III

Standard Configuration for OS-9™ GMX III

The following table lists the standard jumper configuration for the CPU III when used with OS-9 GMX III. Figure designations refer to the JUMPER OPTIONS DRAWINGS on pages 22 and 23 of the CPU III hardware manual.

Jumper	Figure	Remarks
JA-1	HH	Solder jumper
JA-2	--	Not currently required by OS-9 may be user defined
JA-3	--	Not currently required by OS-9 may be user defined
JA-4	D	146818 must be jumpered for IRQ as shown 6840 not required, may be user defined
JA-5	F	Recommended for maximum system security
JA-6	J	Required if NMI from bus is needed
JA-7	L	Turn battery OFF when servicing board
JA-8	Q	OS-9 GMX III is contained in the lower 4K of the supplied EPROM
JA-9	--	Not currently required by OS-9 may be user defined
JA-10	--	Test points only
JA-11	V	Interrupts cleared on system reset
JA-12	--	Not currently required by OS-9 may be user defined
JA-13	BB	OS-9 GMX III supports write-protect
JA-14	EE	OS-9 GMX III is currently supplied in one half (4K) of an 8K EPROM

GMX™ 6809 CPU III

Standard Configuration for UniFLEX™

The following table lists the standard jumper configuration for the CPU III when used with UniFLEX™. Figure designations refer to the JUMPER OPTIONS DRAWINGS on pages 22 and 23 of the CPU III hardware manual.

Jumper	Figure	Remarks
JA-1	HH	Solder jumper
JA-2	Fig. 1 Page a-3	Special configuration for UniFLEX™
JA-3	--	Not currently required by UniFLEX may be user defined
JA-4	D	146818 must be jumpered for IRQ as shown 6840 must be jumpered for IRQ as shown
JA-5	F	Recommended for maximum system security
JA-6	J	Required if NMI from the bus is needed
JA-7	L	Turn battery OFF when servicing board
JA-8	--	Don't Care - UniFLEX uses 2K EPROM
JA-9	--	
JA-10	--	Test points only
JA-11	V	Interrupts cleared on system reset
JA-12	--	Not currently required by UniFLEX may be user defined
JA-13	BB	
JA-14	FF	UniFLEX™ Bootstrap is currently supplied in one 2K EPROM

(cont.)

6840 PTM Configuration for UniFLEX™

Jumper area JA-2 must be configured as shown below for use with UniFLEX. For proper operation, the time-base for the 6840 (Y3) must be the standard (500 KHz.) version.

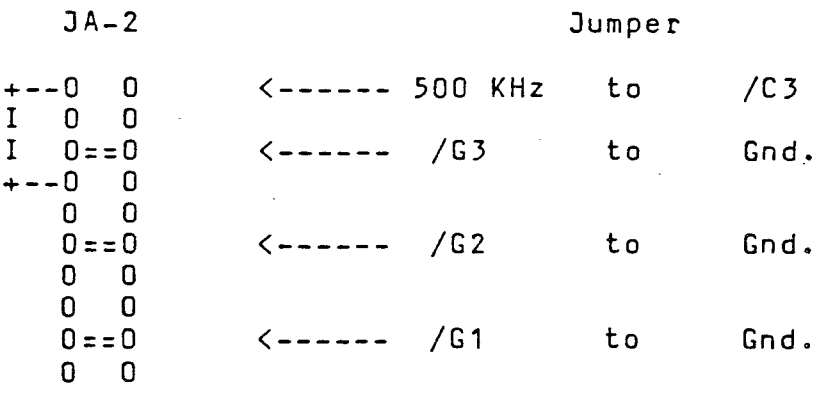


Figure 1

Other Changes to the CPU III for UniFLEX™

The UniFLEX™ configuration requires special versions of three of the PALs (Programmable Array Logic) on the CPU III. This results in the following variations from the information in the CPU III hardware documentation.

1: The EPROM/ROM socket (U-37) only occupies 2K of address space. $FF000-$FF7FF is open for UniFLEX.

2: The upper 512 bytes of the LOGICAL ADDRESS SPACE ($FE00-$FFFF) are ALWAYS read and write protected in the User State. Attempts to access this area from the User State are blocked by the hardware (Writes have no effect, reads return invalid data) but no traps are generated.