DATEL

DVME-601 Series
16S or 8D-Channel 68010-based VME A/D Coprocessor Board

PRODUCT DATA SHEET

FEATURES

- **Local 8 MHz 68010 CPU plus:**
  - **— 64 Kb Private RAM**
  - **— 64/128 Kb EPROM**
  - **— 64 Kb Dual-ported RAM**
- **Various analog-to-digital front ends:**
  –12 Bits, 2,4, or 20 µSeconds
  –16 Bits, 35 µSeconds or 400 mSeconds
- **16 Single-ended or 8 differential on-board analog input channels expandable to 232 total channels using DATEL's slave MUX boards.**
- **Simultaneous A/D scanning and host transfer of previous scans. Ideal for DSP, FFT, ATE, graphics.**
- **Monitor/Executive firmware to run in "no program mode" or from user programs.**
- **RS-232-C serial port to debug optional user software. Programs may be downloaded through dual-port RAM or serial port and reprogrammed in EPROM.**
- **Programmable Peripheral I/O (68901):**
  - **— RS-232-C serial port USART.**
  - **— Three timer outputs which are software-programmable as interrupts, A/D start triggers, or pulses.**
  - **— Two 8-bit counter inputs.**
  - **— Five TTL I/O bits or interrupts.**
- **On-board +5V dc-to-dc power converter**
- **Programmable vectored interrupt.**
- **A/D Start by external trigger, timer or program.**
- **Sample-to-memory transfers at up to 300 KHz.**
- **Easily integrates with popular host multitasking Real Time Operating Systems such as UNIX, PDOS, OS-9, and VRTX.**

Today's VME environment requires that busy host processors run real-time operating systems, seriously affecting the host system's processing speed. DATEL'S DVME-601 coprocessor board integrates high-performance A/D data acquisition with a local 68010-based microcomputer. This unique single-board design lets the host handle other tasks while the DVME-601 smart A/D board simultaneously collects analog data.

GENERAL DESCRIPTION

Unlike dumb A/D boards, the DVME-601 coprocessor board automatically collects scanned data without delaying other host tasks. When A/D data is ready, the DVME-601 coprocessor board can interrupt the VME host. The host can then transfer data from previous scans to memory without halting collection of the next scan's data.

Typical applications for the DVME-601 include high-speed process control loops, analytical instruments, vehicular data recorders, ATE equipment and communications testers. The block-oriented, interrupt-controlled memory transfers of A/D scans to the host are particularly suited to digital signal processing applications. Such applications include acoustics, sonar, high-speed mapping, seismology, medical imaging, graphics, array processing, FFT's, and waveform analysis. Using the single-channel fast-throughput mode and the 4 µSec., 12-bit converter yields true speeds of up to 170,000 samples per second to memory (single channel, gain = 1).

As shown in Figure 1, the data acquisition section includes an analog input multiplexer with 16S/8D local channels expandable up to 232 total channels using DATEL's slave multiplexer boards. The board includes an instrumentation amplifier which may be resistor-programmed by the user for gains of up to 1000. A choice of pluggable A/D converter modules is offered on four different models. Resolution from 12 to 16 bits is available with 12-bit conversion speeds down to 2 µSec.

TABLE OF CONTENTS

(UNIX is an AT&T trademark. PDOS is an Eyring Research Institute trademark. OS-9 is a Microware trademark. VRTX is a Hunter and Ready trademark.)

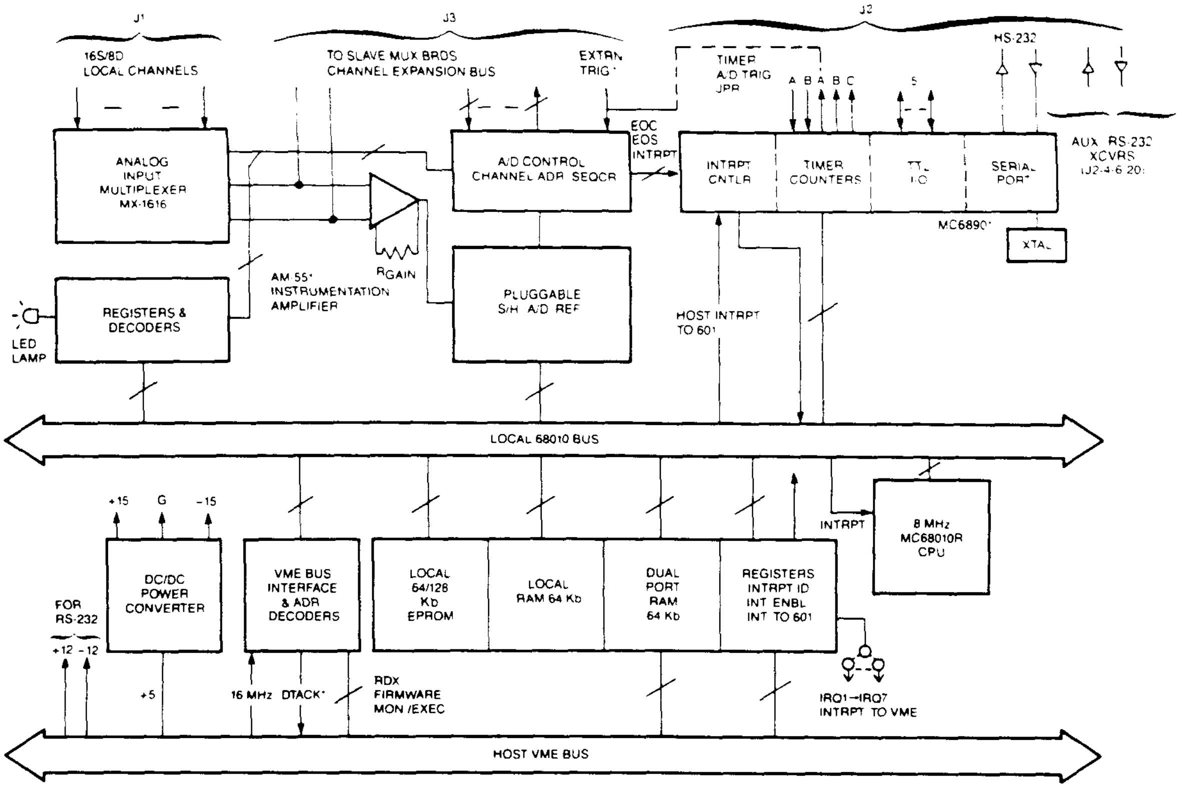

Figure 1. DVME-601 Coprocessor Board Simplified Block Diagram

External triggers, the internal programmable timer, local programs, or host commands may initiate A/D conversions. Onboard EPROM firmware manages all of these start modes. A dc-to-dc converter supplies low-noise, regulated power to the data acquisition section.

The primary interface between the DVME-601 and its VME host is a 64 Kb dual-ported random access memory (DPR). The DPR is used for commands, subroutines and parameters, control/status bits, A/D data blocks, optional downloads of user programs, and bidirectional interrupts between the DVME-601 and the host. The maskable interrupt to the VMEbus normally occurs after A/D scanning, but may also be issued by a local user program.

Executive firmware included in the EPROM offers many ways to manage the DPR including swapped buffer ("ping-ponged") A/D scan transfers while the host reads the alternate buffer. The user may run the DVME-601 either in the "no-programming" mode or may load and execute their own programs. The no-program mode uses fast A/D routines supplied in the EPROM. The firmware also includes a serial port monitor program for developing optional user programs.

The DVME-601's full power and flexibility is realized with user-written software. As a high-performance, general-purpose microcomputer, the DVME-601 is ideal for automatic A/D data collection and arithmetic pre-processing of A/D data before sending it to the host. Transferring pre-processed results rather than raw data enhances total system performance while the host continues with disk, display, or control activities. Programs may be developed in the host, saved on disk, then downloaded to local RAM via the DPR or serial port. The pluggable EPROM may be reprogrammed by the user or by DATEL under special order. Any language may be used such as Assembly, BASIC, FORTRAN or C if it can be compiled to 68010 code.

The local microcomputer consists of an 8 MHz MC68010 microprocessor, 64 Kb of pluggable EPROM (socketed to 128 Kb), 64 Kb of DPR and 64 Kb of private RAM. Total local storage of A/D data may approach 62,000 samples using the DPR plus private RAM if no other program is using RAM space.

A programmable 68901 I/O peripheral controller is also included as part of the board design. This device offers an interrupt controller, an RS-232-C serial port, four timer/counters, and five I/O bits. Some of the timers and the serial port are used by the monitor/executive firmware but may be reprogrammed by the user to include external interrupts. A second RS-232-C serial port is available using a software UART and the I/O bits. A green front panel LED lamp lights to confirm power-up self-test and may be programmed by the user for alarms, etc.

Using only the serial port, a 16 MHz clock, and a +5V dc power supply, it is even possible to operate the DVME-601 in stand-alone mode, not connected to the VMEbus. Commands and A/D data pass through the port at rates of up to 19.2 Kb.

The board uses +5V dc at 2.8 A and ±12V dc at 2 mA (typical) from the VMEbus. Connections are made only to P1 (P2 is not used to assure compatibility with most hosts). Data transfer is 16 bits wide. The DVME-601 is a D16 A24 slave board using 24 address lines and 6 address modifier lines. The board occupies 64 Kb of host memory. A single interrupt to the host asserts a programmable interrupt vector. Three front panel D-type connectors provide physical interfacing for local analog inputs, analog slave-mux channel expansion, and for the serial/parallel/timer I/O.

The DVME-601 is a 9.19"W x 6.3"D x 0.6"H (233,5 x 160 x 15,2 mm) 6U board. It includes a comprehensive user's manual with programming information. Access to the EPROM source code in several formats is available to customers upon special request.

FUNCTIONAL SPECIFICATIONS

(Typical at +25°C, Gain=1, unless noted.)

DATA ACQUISITION SECTION

Number of On-board Channels . . . . 16 single-ended or 8 differential

Number of Off-board Channels . . . Up to 224 single-ended or 160 differential using DATEL's external channel multiplexer boards. (10 MUX brds. max.)

Total Addressable Channels . . . . . . Up to 232 single-ended or differential

Input Voltage Range . . . . ±10 V full scale. (±5V, 0 to +10V, and 0 to +5V may also be available. See chart page 8.)

Common Mode Voltage Range . . . . ± 10V, maximum, non-isolated

Common Mode Rejection (dc to 60 Hz, CMV = ±10V) . . . . 80 dB, gain=1 with 1 KΩ source unbalance

Input Bias Current . . . . . ±200 pA

Overvoltage Protection . ±30V dc, maximum sustained

Input Impedance . . . . . . Power on: 1000 MΩ, differential or to ground. Power off: 1.5 KΩ

A/D Output Coding (All models) . . . . . Bipolar 2's complement, or bipolar offset binary. The DVME-601A is jumperable for unipolar straight binary.

Instrumentation Amplifier Type and Gain Range . . . . AM-551, supplied as gain = 1. May be resistor-programmed by user up to gain=1000 with increased settling delay.

Instrumentation Amplifier Settling Delay (gain = 1) . . . . 3 µSec. to 0.01% of FSR

Adjustments . . . . Instrumentation Amplifier offset, A/D offset and A/D gain.

LOCAL MICROCOMPUTER

CPU Type and Clock Speed . . . MC68010R8, 8 MHz (requires VMEbus SYSCLK)

Local Data Bus Width . . . 16 Bits

Local Read/Write Memory . . . . . . 64 Kilobytes static RAM (no VMEbus access)

Local Read-only Memory . . . . UV-erasable EPROM, 64 Kb supplied as two 27C256's but is socketed for two 27C512's totaling 128 Kb.

Dual-ported Read/Write Memory . . . . 64 Kb (VMEbus and local access)

Front Panel LED Lamp . . Green LED lamp is lit if local CPU power-up self-test succeeds. User software may use the lamp for alarms, etc.

A/D-S/H RESOLUTION AND CONVERSION PERIOD OPTIONS

(See Technical Note 1)

Model	Resolution	Convert time	Thruput to RAM
DVME-601A	12 Bits	20 µSec.	40 KHz
DVME-601B	12 Bits	4 µSec.	100 KHz
DVME-601C	16 Bits	35 µSec.	25 KHz
DVME-601D	16 Bits	400 mSec.	2.5 Hz
DVME-601E	12 Bits	2 µSec.	see notes
DVME-601F	14 Bits	4 µSec.	100 KHz

SYSTEM PERFORMANCE

Specifications	A/D Type 601A 12 bit,	 601E,B 12 bit,	 601C 16 bit,	 601D 16 bit,	 601F 14 bit,
Speed	20 µS	2 or 4 µS	35 µS	400 mS	4 µS
Accuracy, min	0.025% of FSR	0.05% of FSR	0.01% of FSR	0.0063% of FSR	0.01% of FSR
Non-linearity & noise, max.	1/2 LSB	1/2 LSB	2 LSB	2 LSB	2 LSB
Zero Tempco max.	±20 ppm/°C	±20 ppm/°C	±20 ppm/°C	±10 ppm/°C	±15 ppm/°C
Gain Tempco max.	±20 ppm/°C	±20 ppm/°C	±20 ppm/°C	±10 ppm/°C	±15 ppm/°C

External A/D Start Trigger . . . . . Negative-going TTL input with 4.7 KΩ pullup to +5V. Pulse width is 100 nSec. minimum, 2 µSec. max.

Local Pacer Clock . . . . . Software-programmable to cause either an A/D scan start or a single conversion.

Pacer Clock Interval Range . . . . 3.255 µSec. to 41.667 mSec. using one timer.

Software Reset . . . . Write to BASE + $FFF9, bit 0

PERIPHERAL I/O CONTROLLER

Controller Type . . . . MC68901 multifunction peripheral, crystal-controlled, 2.4576 MHz, user-programmed.

Interrupts

Local Hardware Interrupts to 68010 CPU (Maskable) . . . . . . A/D End of Conversion, A/D End of Scan, VMEbus host command request

Local Software Interrupts (Programmable) . . . . . . Any timer count reached or I/O bits 0 through 4.

Digital I/O

Number of I/O Lines . . 5 lines, individually programmable as inputs, outputs, or interrupts.

Logic Levels . . . . TTL levels, 1 load maximum with 10 KΩ pullups to +5V.

PERIPHERAL I/O CONTROLLER (cont.)

Timer/Counters

Number of Timers	4 8-bit timers with pre-scale up to divide-by-200.
Timer Outputs	3 outputs, (Timers A,B,C), 1 TTL load maximum. Timer D is the USART baud clock but may be reprogrammed.
Timer/Counter Inputs	2 inputs, TTL levels with 10 KΩ pullups to +5V.

Serial Port (See Technical Note 2)

Number of Serial Ports	1 USART, full duplex, RS-232-C levels, DTE pinout.
RS-232-C Handshakes	DTR, DSR, RTS, CTS programmed by the user. (See J2 pinout).
Modes	Synchronous or asynchronous.
Number of Stop Bits	0, 1, 1.5, or 2
Number of Data Bits	5, 6, 7, or 8
Parity	Odd, even, none for receiver. Transmitter is user-coded.
Baud Rates	Up to 19.2 Kilobaud.

VMEbus INTERFACE

Standards Compliance	IEEE P1014/D1.0
Data Bus Width	16 Bits
Address Bus	A24 D16 slave, 24 address lines (A23-A01) plus 6 address modifiers, jumper selected.
Address Modifier Codes	39 hex or 3D hex, jumperable.
Architecture (See memory map)	Dual-ported 64 Kb block mapped on 64 Kb boundaries.
VME Bus Interrupter (See Note 3)	1 line, jumper-selectable IRQ1* through IRQ7*.
Data Transfer	16 bits using P1. Generates DTACK* derived from 16 MHz bus clock.

CONNECTORS

VMEbus, P1	96-pin male DIN connector. P2 connector not used.
Local Analog Input, J1	25-pin DB-25S female on front panel.
Multifunction I/O Peripheral (68901), J2	25-pin DB-25S female on front panel.
Analog Input Channel Expansion Bus, J3	25-pin DB-25S female on front panel, compatible to DATEL DVME-64X series multiplexer boards.

MISCELLANEOUS

Power Required	+5V dc ±5% at 3.1 A max. and ±12V dc at 10 mA max. from VMEbus for serial port. A local ±15V dc-to-dc converter is included for linear circuits
Operating Temperature	0 to +60°C
Storage Temperature	−20°C to +80°C
Relative Humidity	10% to 90%, non-condensing.
Outline Dimensions	Double-height VME, 6U outline. 9.19" W x 6.3" D x 0.6" H, (233,5 x 160 x 15,24 mm).
Weight	17 ounces (482 grams)

TECHNICAL NOTES

1. The typical throughput rate is an aggregate time within a multichannel scan and does not include subroutine setup time. The rate includes times for channel sequencing, MUX, Inst. Ampl., S/H acquisition/settling, A/D conversion, and 68010 software times. The polled EOC STSNSC subroutine is used, triggering A/D conversion on each data read. Higher single-channel speed is available using the "fast-throughput" mode (delayed DTACK*). The DVME-601E offers 300 KHz in a long burst to local RAM, single channel, gain = 1. In multichannel, all modes require at least 6 microseconds (gain = 1) from channel sequencing to A/D start plus the A/D conversion time. Data transfer time may overlap A/D conversion.
2. EPROM Monitor firmware uses Timer D and the serial port. Communications format is 9600 baud, 8 data bits, no parity, and 1 stop bit. A 4800-baud software UART is formed with I/O bit 0 for optional serial S record downloads. Firmware also uses Timer A as an A/D start clock. The user may re-program all functions.
3. Asserts one interrupt ID code which is programmable from the host. A DVME-601 local register generates the interrupt. The host may mask this interrupt by writing to a DVME-601 DPR register.
4. For model DVME-601F, allow 20 minutes warmup to rated specifications.

DVME-601 FIRMWARE OVERVIEW

Memory Mapping

Table 1 shows the relationship between the local DVME-601 memory and the host DPR window. Arbitration circuits prevent simultaneous access to the DPR by delaying either the local DVME-601 DTACK* or the VMEbus DTACK*. Three addresses in the DPR are hardware-mapped from the host. They enable interrupts to the host, select the interrupt ID vector that is asserted and force a local 68010 interrupt to execute a Function Block Command or local subroutine.

A portion of the top of Read/Write DPR is reserved for control/status bits defined by the DVME-601 firmware. A portion of the local RAM is reserved for the DVME-601 system control, vectors and stacks. Data acquisition and 68901 registers, as well as a register to interrupt the host from a local program, are hardware-mapped in local memory.

Table 1. DVME-601 Memory Map

Local Memory		Host Memory	
$000000- $01FFFF	64/128 Kb Local EPROM (READ only)		
$020000- $02FFFF	64 Kb Local RAM (READ/WRITE)		
$040000- $04FFF7	64 Kb Shared Dual Port RAM (READ/WRITE)	Base + $0000- Base + $FFF7 [$FFF0-FFF7 are soft-mapped R/W cmd/stat. See manual.]	Shared 64 Kb DPR (RD/WR)
$04FFF8 $04FFF9	Not used Not used	Base + $FFF8 Base + $FFF9	Enable interrupt to VMEbus (RD/WR) (Bit 7)
$04FFFA $04FFFB	Not used Not used	Base + $FFFA Base + $FFFB	Host interrupt ID Vector (RD/WR)
$04FFFC $04FFFD	Not used Not used	Base + $FFFC Base + $FFFD	Force command interrupt to DVME-601 (WR-only word from host)
$04FFFE $04FFFF	Not used Not used	Base + $FFFE Base + $FFFF	Not used Not used
$06XXXX	A/D Start Channel Address Reg. (WR)	Notes: 1. Unlisted addresses are redundant or not defined. All addresses are in hexadecimal. "XXXX" bytes are not decoded and are don't care. Factory-jumpered base addressing is $FA0000. 2. Hardware registers from $6XXXX to $10XXXX require MOVE.W instructions. Local RAM from about $20000-21000 is reserved. 68901 registers require MOVE.B instructions or Read-Modify-Write. 3. BASE + $FFFC is write-only from VMEbus and should be located beyond power-up memory testing. 4. *Software Reset is available at B + $FFF9, bit 0*	
$08XXXX	A/D Final Channel Address Reg. (WR)		
$0AXXXX	Command/Status Reg. (R/W) [EOC,EOS, LED, A/D, etc.]		
$0CXXXX	Start A/D Convert (WRITE only)		
$0EXXXX	Force interrupt to VMEbus Host (WR)		
$10XXXX	A/D Data Register (READ only)		
$120000- $12002F	68901 Intrpt/Timers/ USART/Parallel Port (READ/WRITE)		

A/D Converter Command Modes

Location $0AXXXX in the DVME-601's local memory map contains a command register. The settings of individual bits in this register determine:

- A/D converter triggering,
- modes of channel sequencing, and
- how the converter transfers data.

Executive firmware manages this register; however, it may be directly controlled by user-written programs as well. The addresses below refer to the local 68010 memory map (see Table 1). The command register controls the following hardware functions:

1. A/D conversion starts with a short settling delay before the actual conversion. Three events can trigger the delay:

   A. An external TTL trigger or timer input arrives. (The logic enables the trigger and waits for the falling edge.)

   B. The local CPU reads the A/D data register ($10XXXX).

   C. A write to location $0CXXXX occurs.

These modes may be partially combined. All modes must test the End of Conversion (EOC) bit to confirm that data is ready.

2. The start pulse may start either a single conversion or a scan of N channels as defined by the start and final channel address registers ($06XXXX and $08XXXX). If the start defines a SCAN, then individual samples must still be started by reading the A/D data register.

3. A "fast throughput" mode is offered where the A/D converter logic holds off the local DTACK* input to the 68010 until the End of Conversion output. This mode offers higher speeds by eliminating EOC polling. When used with a block move instruction loop which auto-increments the destination pointer, this acts like a DMA transfer. During DMA transfers, the instruction remains in the 68010's queue and requires no opcode fetch cycle.

4. A "re-scan mode" is offered which automatically reloads the start channel register when the End of Scan is reached. This mode yields higher speeds for repeated scans by eliminating the address register write.

5. The channel address sequencer may be inhibited from incrementing after each sample. This would be used for very high-speed single channel applications and is similar to loading the same address in both the start and final addresses registers.

6. The A/D start logic may be inhibited by local 68010 command. This would be used to ignore external triggers until ready or for other usage.

NOTE

EOC/EOS interrupts are always sent to the local 68901 interrupt controller but are normally masked off by programming the 68901 registers.

EPROM Firmware

The 64 Kb EPROM includes Monitor and Executive software. The Monitor helps in developing and debugging optional user-written programs and is available only via the RS-232-C serial port. This port is on the DVME-601's front panel J2 connector. *The user simply connects a "dumb" terminal to this connector.*

The Monitor is not required if the user chooses to control the DVME-601 only from the DPR using the Executive. After successful power-up testing, the LED is lit and the DVME-601 automatically enters the Monitor. This LED illuminates to indicate successful memory and I/O tests. The DVME-601 attempts to run the Monitor even if self-test fails. The Executive will be off, insuring that the DVME-601 will ignore any power-up bus activity in the DPR. The user switches the board from the Monitor to the Executive state via either the serial port or the DPR.

The Executive accepts function blocks loaded from the DPR by the host. The blocks may contain reserved command words, local subroutine addresses, and optional input or output parameters. The host tells the DVME-601 to execute a previously loaded command by writing to a reserved location in the DPR. This forces a local 68010 interrupt to branch to the command or routine.

A typical subroutine would set the start and final analog channel address registers. It would include these addresses as longword parameters in a sequential list after the subroutine memory address. Another subroutine would start A/D scanning. Its "output parameter" would be blocks of digitized A/D data transferred to the DPR.

Subroutine addresses may also include the addresses of user-written code which was previously downloaded into local memory or reprogrammed in the EPROM. A function command is reserved to perform the DPR download of S records. Since the S records contain destination addresses and may be of any length, the code may be sent anywhere in usable local RAM. It can overlay previous code and can be repeated indefinitely at high speed. It can include tables as well as executable binary image since it is not executed as part of the download.

All of this activity is controlled by VMEbus commands via the Executive. While a subroutine runs, a reserved control/status area of DPR indicates when command execution is in progress and when a command block is done. Any subroutine (including a long multiple A/D scan) may be interrupted to return to the command level or execute a different routine. Thus, the user has an extremely powerful, general purpose means of controlling the DVME-601.

Monitor Commands

A conventional hexadecimal firmware Monitor allows full access to the 68010 CPU registers, data acquisition registers, the 68901 peripheral, and to memory locations (including the DPR). Programs may be run under breakpoint and/or trace control. The Monitor also includes several A/D diagnostic commands to aid in calibration or hardware troubleshooting. A list of monitor command functions appears in Table 2.

Table 2. Summary of Monitor Command Functions

Monitor Commands
Read/Write CPU or memory registers.
Display memory block in hexadecimal and ASCII.
Fill memory block with a constant.
Set or display breakpoint.
Trace one or more instructions.
Start execution until optional breakpoint.
Turn Executive ON or OFF (enable/disable DPR commands).
Transition to/from Executive or Monitor.
Auxiliary serial port download.
Start A/D sampling to serial port.

Executive Commands

The Executive accepts commands and optional parameters through the DPR. A major benefit of the Executive is a uniform sequential list method of passing subroutine parameters. User programs may also use this syntax or may develop their own parameter-passing method in another part of the DPR. If the user's serial port terminal is connected, the execution of Executive Function Blocks may be analyzed at the Monitor level. The Executive also traps non-executable subroutine addresses by performing a soft reset.

Executive Commands
Select the memory destination address of A/D scans as either local RAM, single DPR or swapped DPR buffers.
Select the source of A/D start triggers as external TTL, local timer, last A/D read or host command.
Select A/D triggering per conversion or per scan.
Select channel address sequencer to increment or not after each conversion.
Select start/final sequential channel addresses.
Select timer channel, period and control.
Transfer A/D scans from local RAM to DPR (after data pre-processing by a user-downloaded program).
Define whether scan transfers will wait for host Ready status or transfer without waiting.
Select how scanning will stop (N samples, N scans, buffer full, stop by host).
Download S records via DPR or auxiliary serial port and flag checksum errors but do not start execution.
Load subroutine addresses or function command block with optional parameter list and await execution.
Execute previously loaded commands or subroutine(s).
Memory block transfer.

The Executive allows for repeating or alternating blocks of sequential subroutines. They may be selected once, N number of times, or until stopped by the host. The DPR Download is one of the Executive commands. Most of the subroutines available through the Executive manage the data acquisition section. Table 3 lists the functions of executive commands available.

Speed by Architecture

The primary difference between the DVME-601 smart A/D board and "dumb" A/D boards is the increased total system throughput achieved by offloading the host. This is a result of simultaneous A/D scanning and concurrent host processing, plus using the DPR as a programmable buffer. Even greater system bandwidth may be possible by having the DVME-601 do arithmetic pre-processing of A/D blocks, delivering final results rather than raw input data.

Figure 2 shows the wasted idle times in the host for dumb A/D boards because an interrupt, polling, or DTACK* delay must occur with each sample.

The interrupt processing takes many microseconds to save stacks and registers and arbitrate with the Real Time Operating System. To realize the full speed of a dumb board, the host must be fully dedicated to data acquisition, leaving no time for non-A/D tasks. With fast converters and high bandwidth inputs, the short sample intervals make it inefficient to run the host in an interrupt mode, thereby locking out other host software tasks during data acquisition. The typical lack of memory on dumb boards for sample storage also means that the last sample is saved in the A/D converter and new sampling cannot start until the old sample is read.

The DVME-601 efficiently runs long blocks of thousands of samples, allowing ample time for host disk and display activity between blocks.

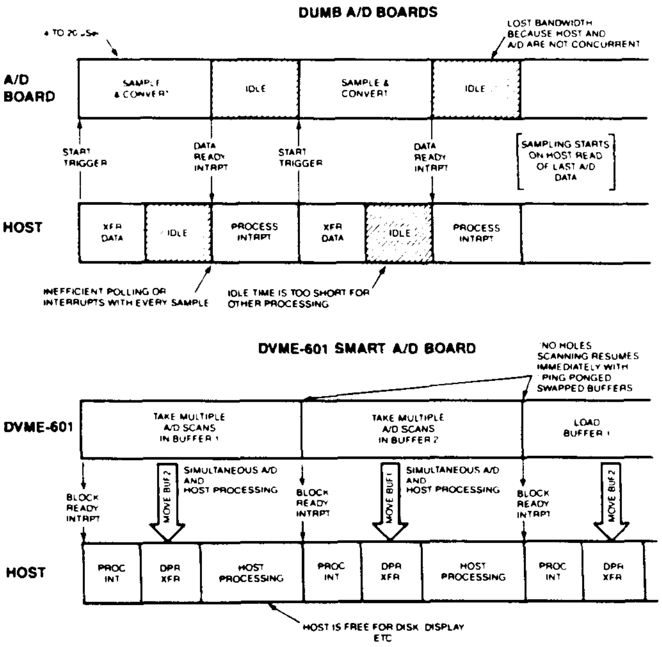
DUMB A/D BOARDS
4 TO 20 uSec
LOST BANDWIDTH BECAUSE HOST AND A/D ARE NOT CONCURRENT
A/D BOARD
SAMPLE & CONVERT
IDLE
SAMPLE & CONVERT
IDLE
START TRIGGER
DATA READY INTRPT
START TRIGGER
DATA READY INTRPT
SAMPLING STARTS ON HOST READ OF LAST A/D DATA
HOST
XFR DATA
IDLE
PROCESS INTRPT
XFR DATA
IDLE
PROCESS INTRPT
INEFFICIENT POLLING OR INTERRUPTS WITH EVERY SAMPLE
IDLE TIME IS TOO SHORT FOR OTHER PROCESSING
DVME-601 SMART A/D BOARD
NO HOLES SCANNING RESUMES IMMEDIATELY WITH PING PONGED SWAPPED BUFFERS
DVME-601
TAKE MULTIPLE A/D SCANS IN BUFFER 1
TAKE MULTIPLE A/D SCANS IN BUFFER 2
LOAD BUFFER 1
BLOCK READY INTRPT
MOVE BUF 2
SIMULTANEOUS A/D AND HOST PROCESSING
BLOCK READY INTRPT
MOVE BUF 1
SIMULTANEOUS A/D AND HOST PROCESSING
BLOCK READY INTRPT
MOVE BUF 2
HOST
PROC INT
DPR XFR
HOST PROCESSING
PROC INT
DPR XFR
HOST PROCESSING
PROC INT
DPR XFR
HOST IS FREE FOR DISK DISPLAY ETC

Figure 2. Speed by Architecture

I/O SIGNALS AND CONNECTIONS

A/D Channel Expansion Bus

(Please refer to the Channel Address Map, Table 4)

The DVME-601's J3 front panel connector accepts a flat cable assembly, such as DATEL Part Numbers DVME-C-01 or DVME-C-02, to form a channel expansion bus. The cable assembly plugs into DATEL's slave multiplexer boards installed in slots adjacent to the DVME-601 or in a nearby VME chassis. Available DATEL channel expansion boards include:

- DVME-641 A 32 Single-ended/16 Differential channel high-speed MUX;
- DVME-643 An 8 Differential channel low-level isolated MUX (for sensors such as thermocouples, RTD's, 4-20 mA loops, etc.); and,
- DVME-645 A 16 Single-ended/8 Differential channel simultaneous sample/hold MUX.

The DVME-645 is especially suited to array processing and DSP applications. This channel expansion bus allows the DVME-601 to directly control each slave MUX board and carries three classes of signals. They are:

1. Eight-bit channel address outputs from the DVME-601's address register, offering up to 256 total channels. This autosequencing register is software-controlled by the user's host program or DVME-601 firmware.
2. Buffered high-level switched differential analog signals into the DVME-601's instrumentation amplifier.
3. Control and handshake lines, an external A/D start trigger, and grounds.

Channel addresses are distributed to all MUX boards along the bus. The first 8 or 16 addresses are for local DVME-601 channels. Address selection logic and base address switches on each MUX board allows it to respond to a range of addresses. All other de-selected boards disconnect their analog outputs from the bus until addressed. For diagnostics, each MUX board has a LED lamp which turns on when that board is addressed. DATEL offers 2- and 3-connector cables to connect one or two slave MUX boards and users may fabricate flat cables for connecting up to 10 boards.

One of the control lines on the expansion bus is a TTL-compatible, open-collector A/D trigger. This lets the DVME-601's A/D start input be initiated from the trigger input on any multiplexer board. A single external event hardware trigger will start either one A/D sample and host interrupt or a scan of channels and interrupt on one or more boards. Alternatively, automatic channel sequencing and A/D conversion may be started from a DVME-601 timer by jumpering the timer output to the trigger input on the J3 connector. Software commands from the host will also start the A/D conversion process.

A settling delay control line output from each MUX board will delay the actual A/D conversion to synchronize settling times of low-level pre-amplifiers on the MUX slave boards. Through appropriate host A/D software, board addressing and input range selection, it is even possible to mix high- and low-level expansion input boards on the same bus.

Figure 3 shows how a DVME-601 Coprocessor board physically links to DATEL's slave multiplexer boards and field equipment.

Table 4. Analog Channel Expansion Address Map

Unused addresses	($FFh)
Expansion MUX Board N : : Expansion MUX Board 2	
Expansion MUX Board 1	($0Fh or $1Fh)
DVME-601 16S/8D Channels	($00)

Input/Output Connections

Figures 4, 5, and 6 show the signals present on the DVME-601's J1, J2, and J3 connectors respectively. The connectors are dedicated as follows:

J1	Local Analog Inputs
J2	Multifunction Peripheral I/O Signals
J3	Analog Channel Expansion Bus

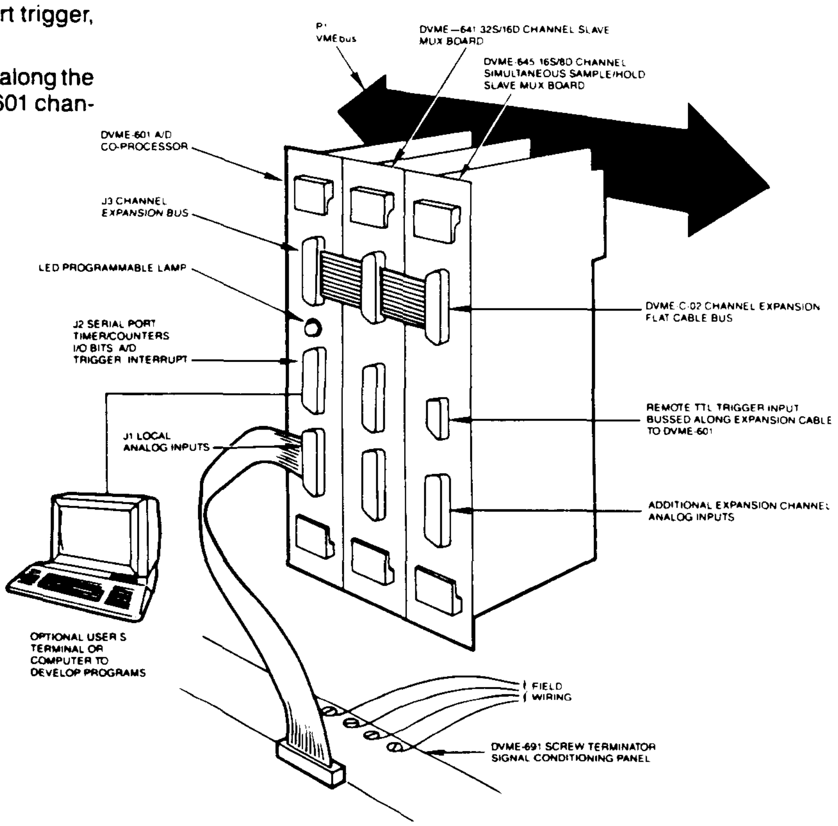
P1
VMEbus
DVME—641 32S/16D CHANNEL SLAVE MUX BOARD
DVME-645 16S/8D CHANNEL SIMULTANEOUS SAMPLE/HOLD SLAVE MUX BOARD
DVME-601 A/D CO-PROCESSOR
J3 CHANNEL EXPANSION BUS
LED PROGRAMMABLE LAMP
J2 SERIAL PORT TIMER/COUNTERS I/O BITS A/D TRIGGER INTERRUPT
J1 LOCAL ANALOG INPUTS
DVME-C-02 CHANNEL EXPANSION FLAT CABLE BUS
REMOTE TTL TRIGGER INPUT BUSSED ALONG EXPANSION CABLE TO DVME-601
ADDITIONAL EXPANSION CHANNEL ANALOG INPUTS
OPTIONAL USER'S TERMINAL OR COMPUTER TO DEVELOP PROGRAMS
FIELD WIRING
DVME-691 SCREW TERMINATOR SIGNAL CONDITIONING PANEL

Figure 3. Front Panel Cabling and Wiring Connections

J1 (VIEWED FROM FRONT OF BOARD)

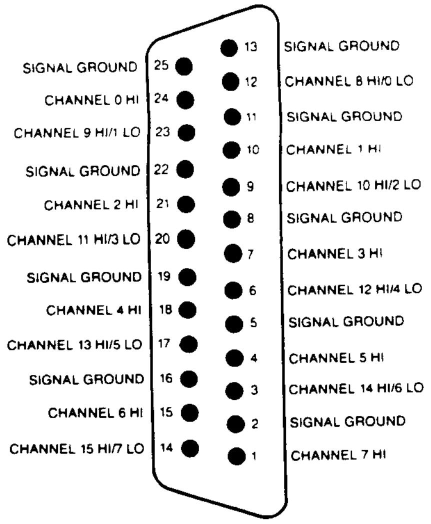

Figure 4. Local Analog Input Connector Analog (J1)

J2 (VIEWED FROM FRONT OF BOARD)

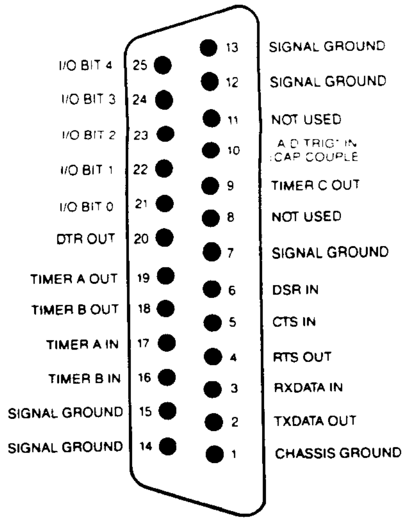

Figure 5. Multifunction Peripheral I/O Connector (J2)

J3 (VIEWED FROM FRONT OF BOARD)

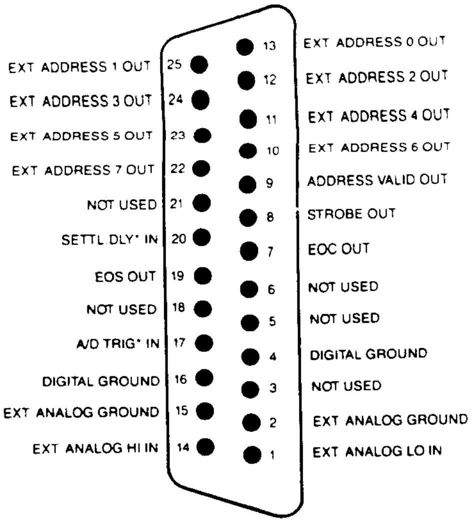

Figure 6. Analog Channel Expansion Bus Connector (J3)

ORDERING GUIDE

Model Number	A/D Bits, Conversion Speed, and Input Configuration
DVME-601A	12 Bits, 20 µSec., unipolar or bipolar
DVME-601B	12 Bits, 4 µSec., unipolar or bipolar
DVME-601C	16 Bits, 35 µSec., bipolar
DVME-601D	16 Bits, 400 ms, bipolar
DVME-601E	12 Bits, 2 µSec., unipolar or bipolar
DVME-601F	14 Bits, 4 µSec., ±10V

All models include a 64 Kb EPROM with Monitor/Executive firmware, a User's Manual, and a software disk. A substantial amount of unused EPROM is available to the user. DATEL will review custom software requirements under special order.

HARDWARE ACCESSORIES

Part Number	Description
DVME-691A	Rack-mount screw terminator panel with signal conditioning pads for 32S/16D input channels. Includes flat signal cables compatible with the DVME-601.
DVME-C-01	Channel expansion flat cable with 2 DB-25P connectors for use with one slave MUX board.
DVME-C-02	Channel expansion flat cable with 3 DB-25P connectors for use with two slave MUX boards.

CHANNEL EXPANSION SLAVE MUX BOARDS

Part Number	Description
DVME-641	32S/16D Channel high speed, high-level non-isolated MUX board.
DVME-643	8D Channel low-level, isolated MUX board.
DVME-645	16S/8D Channel simultaneous sampling MUX board.

Contact DATEL for additional DVME-601 software on disk.

Input Range	Model DVME-601A	DVME-601E,B	DVME-601C	DVME-601D	DVME-601F
0 to +5V	X				
0 to +10V	X	X			
±5V	X		X		
±10V	X	X	X	X	X

DATEL

DATEL, Inc. 11 CABOT BOULEVARD, MANSFIELD, MA 02048-1194 TEL. (508) 339-3000 / TELEX 174388 FAX (508) 339-6356
OVERSEAS: DATEL (UNITED KINGDOM) Basingstoke Tel. (256) 469-085 • DATEL (FRANCE) Tel. (1) 3460.0101
DATEL (GERMANY) Tel. (89) 53-0741 • DATEL (JAPAN) Tokyo Tel. (3) 779-1031 • Osaka Tel. (6) 354-2025

For Applications Assistance, dial 1-800-233-2765

APPENDIX B

MC68901 Programming Information

The following information was reproduced from the MC68901 Multi-Function Peripheral Advance Information, dated January 1984. Publication ADI-984. Courtesy of Motorola, Inc.

Refer to Section 5.13 for local addresses of the 68901 registers.

Timer Crystal

This is a standard "microprocessor" crystal with tolerance at +25 °C of ±50 ppm and ±50 ppm temperature coefficient from -30 to +70 °C. The crystal expects a 300 ohm load paralleled with 32 pF. The user may replace this crystal with a higher tolerance type if the warranty is not voided by damaging the board. Do not change the crystal frequency unless you also change the EPROM baud clock divisor used for timer D.

For very exact triggering in spectral, DSP and FFT applications, consider using an external precision TTL trigger source or the local 68901 timer. The normal setup using the local timer for A/D triggering is the delay mode. The smallest prescale using a control byte of $01 is divide-by-4. An additional divide-by-2 is inherent in the output toggle, giving divide-by-8 as the smallest divisor of the 2.4576 MHz 68901 timer. Therefore the smallest period is about 3.255 microseconds using a $01 data byte. A $02 data byte will give about 6.5 microseconds, etc.

External Trigger

IMPORTANT! The external A/D trigger input should be a negative-going pulse with a 100 nS minimum to 2 microseconds maximum width. Triggering occurs on the falling edge. If the trigger is held low too long, it may interfere with the A/D logic. The trigger should return HI ("1") within 2 microseconds after the falling edge.

Note that since the output of the on-board 68901 timer is a square wave, it must be converted to a pulse before use as an A/D trigger. This may be accomplished with an external one shot circuit or simply a capacitor coupling between the timer output and trigger input. (A 1000 pF ceramic cap may be used.) The internal 4700 ohm pullup resistor (R11) to +5V will keep the trigger HI when no pulse is input. Use a back-biased clamp 1N4148 diode across R11. Connect the cathode to +5V. (These components may have already been added to your board with the capacitor connected between J2-10 and J3-17.) If preferred, user written software may link the timer to the A/D through a local software interrupt from the timer. This avoids the external connection but will add interrupt service routine delays.

SECTION 3
INTERRUPT STRUCTURE

In an M68000 system, the MFP will be assigned to one of the seven possible interrupt levels. All interrupt service requests from the MFP's 16 interrupt channels will be presented at this level. As an interrupt controller, the MFP will internally prioritize its 16 interrupt sources. Additional interrupt sources may be placed at the same interrupt level by daisy-chaining multiple MFPs. The MFPs will be prioritized by their position in the chain.

3.1 INTERRUPT PROCESSING

Each MFP provides individual interrupt capability for its various functions. When an interrupt is received on one of the external interrupt channels or from one of the eight internal sources, the MFP will request interrupt service. The 16 interrupt channels are assigned a fixed priority so that multiple pending interrupts are serviced according to their relative importance. Since the MFP can internally generate 16 vector numbers, the unique vector number which corresponds to the highest priority channel that has a pending interrupt is presented to the processor during an interrupt acknowledge cycle. This unique vector number allows the processor to immediately begin execution of the interrupt handler for the interrupting source, decreasing interrupt latency.

3.1.1 Interrupt Channel Prioritization

The 16 interrupt channels are prioritized from highest to lowest, with General Purpose Interrupt 7 (I7) being the highest and I0 the lowest. The priority of the interrupt is determined by the least-significant four bits in the interrupt vector number which are internally generated by the MC68901. Pending interrupts are presented to the processor in order of priority unless they have been masked. By selectively masking interrupts, the channels are in effect re-prioritized.

3.1.2 Interrupt Vector Number

During an interrupt acknowledge cycle, a unique 8-bit interrupt vector number is presented to the system which corresponds to the specific interrupt source that is requesting service.

7	6	5	4	3	2	1	0
V7	V6	V5	V4	IV3	IV2	IV1	IV0

V7-V4 — Copied from the vector register.

IV3-IV0 — Determine highest priority channel requesting interrupt.

IV3	IV2	IV1	IV0	Description
1	1	1	1	General Purpose Interrupt 7 (I7)
1	1	1	0	General Purpose Interrupt 6 (I6)
1	1	0	1	Timer A
1	1	0	0	Receiver Buffer Full
1	0	1	1	Receive Error
1	0	1	0	Transmit Buffer Empty
1	0	0	1	Transmit Error
1	0	0	0	Timer B
0	1	1	1	General Purpose Interrupt 5 (I5)
0	1	1	0	General Purpose Interrupt 4 (I4)
0	1	0	1	Timer C
0	1	0	0	Timer D
0	0	1	1	General Purpose Interrupt 3 (I3)
0	0	1	0	General Purpose Interrupt 2 (I2)
0	0	0	1	General Purpose Interrupt 1 (I1)
0	0	0	0	General Purpose Interrupt 0 (I0)

3.1.3 Vector Register (VR)

This 8-bit register determines the four most-significant bits in the interrupt vector format and which end-of-interrupt mode is used in a vectored interrupt scheme. The vector register should be written to before writing to the interrupt mask or enable registers to ensure that the MC68901 responds to an interrupt acknowledge cycle with a vector number not in the range of allowable user vectors. For information refer to **3.4.1 Selecting the End-Of-Interrupt Mode**.

	7	6	5	4	3	2	1	0	
$17	V7	V6	V5	V4	S	•	•	•	VR
Reset:	0	0	0	0	0	U	U	U	

V7-V4 — Written by user to set the most-significant four bits of the interrupt vector number.
S — In-Service Register Enable
1 = Software end-of-interrupt mode and in-service register bits enabled.
0 = Automatic end-of-interrupt mode and in-service register bits forced low.
2–0 — Not Used

3.2 DAISY-CHAINING MFPs

As an interrupt controller, the MC68901 MFP will support eight external interrupt sources in addition to its eight internal interrupt sources. When a system requires more than eight external interrupt sources to be placed at the same interrupt level, sources may be added to the prioritized structure by daisy-chaining MFPs. Interrupt sources are prioritized internally within each MFP, and the MFPs are prioritized by their position in the chain. Unique vector numbers are provided for each interrupt source.

The IEI and IEO signals implement the daisy-chained structure. The IEI of the highest priority MFP is tied low and the IEO output of this device is tied to the next highest priority MFP's IEI. The IEI and IEO signals are daisy-chained in this manner for all the MFPs in the chain with the lowest priority MFP's IEO left unconnected. Figure 3-1 shows a diagram of the interrupt daisy-chain. Daisy-chaining requires that all parts in the chain have a common IACK. When the common IACK is asserted during an interrupt acknowledge cycle, all parts will prioritize interrupts in parallel. When the IEI signal to an MFP is asserted, the part may respond to the IACK cycle if it requires interrupt service. Otherwise, the part will assert IEO to the next lower priority device. Thus, priority is passed down the chain via IEI and IEO until a part which has a pending interrupt is reached. The part with the pending interrupt passes a vector number to the processor and does not propagate IEO.

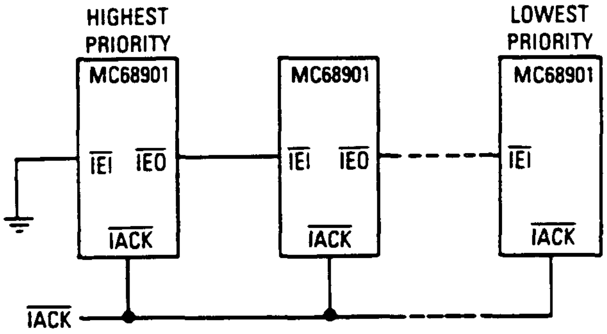

Figure 3-1. Daisy-Chained Interrupt Structure

3.3 INTERRUPT CONTROL REGISTERS

MFP interrupt processing is managed by the enable registers A and B, interrupt pending registers A and B, and interrupt mask registers A and B. These registers allow the programmer to enable or disable individual interrupt channels, mask individual interrupt channels, and access pending interrupt status information. In-service registers A and B allow interrupts to be nested as described in **3.4 NESTING MFP INTERRUPTS**. The interrupt control registers are shown in the following paragraphs.

3.3.1 Interrupt Enable Registers (IREA, IERB)

The interrupt channels are individually enabled or disabled by writing a one or a zero, respectively, to the appropriate bit of interrupt enable register A or B (IERA or IERB). The processor may read these registers at any time.

When a channel is enabled, interrupts received on the channel will be recognized by the MFP, and IRQ will be asserted to the processor indicating that interrupt service is required. On the other hand, a disabled channel is completely inactive; interrupts received on the channel are ignored by the MFP.

Writing a zero to a bit of interrupt enable register A or B will cause the corresponding bit of the interrupt pending register to be cleared. This will terminate all interrupt service requests for the channel and also negate IRQ unless interrupts are pending from other sources. Disabling a channel, however, does not affect the corresponding bit in interrupt in-service registers A or B. So, if the MFP is in the software end-of-interrupt mode (see **3.4.3 Software End-Of-Interrupt**) and an interrupt is in service when a channel is disabled, the in-service bit of that channel will remain set until cleared by software.

	7	6	5	4	3	2	1	0	
$07	GPIP7	GPIP6	Timer A	RCV Buffer Full	RCV Error	XMIT Buffer Empty	XMIT Error	Timer B	IERA
	7	6	5	4	3	2	1	0	
$09	GPIP5	GPIP4	Timer C	Timer D	GPIP3	GPIP2	GPIP1	GPIP0	IERB
Reset:	0	0	0	0	0	0	0	0	

3.3.2 Interrupt Pending Registers (IPRA, IPRB)

When an interrupt is received on an enabled channel, the corresponding interrupt pending bit is set in interrupt pending register A or B (IPRA or IPRB). In a vectored interrupt scheme, this bit will be cleared when the processor acknowledges the interrupting channel and the MFP responds with a vector number. In a polled interrupt system, the interrupt pending registers must be read to determine the interrupting channel, and then the interrupt pending bit is cleared by the interrupt handling routine without performing an interrupt acknowledge sequence.

	7	6	5	4	3	2	1	0	
$0B	GPIP7	GPIP6	Timer A	RCV Buffer Full	RCV Error	XMIT Buffer Empty	XMIT Error	Timer B	IPRA
	7	6	5	4	3	2	1	0	
$0D	GPIP5	GPIP4	Timer C	Timer D	GPIP3	GPIP2	GPIP1	GPIP0	IPRB
Reset:	0	0	0	0	0	0	0	0	

3.3.3 Interrupt Mask Registers (IMRA, IMRB)

Interrupts are masked for a channel by clearing the appropriate bit in interrupt mask register A or B (IMRA or IMRB). Even though an enabled channel is masked, the channel will recognize subsequent interrupts and set its interrupt pending bit. However, the channel is prevented from requesting interrupt service ($\overline{IRQ}$ to the processor) as long as the mask bit for that channel is cleared. If a channel is requesting interrupt service at the time that its corresponding bit in IMRA or IMRB is cleared, the request will cease, and $\overline{IRQ}$ will be negated unless another channel is requesting interrupt service. Later, when the mask bit is set, any pending interrupt on the channel will be processed according to the channel's assigned priority. IMRA and IMRB may be read at any time. Figure 3-2 shows a conceptual circuit of an MC68901 interrupt channel.

	7	6	5	4	3	2	1	0	
$13	GPIP7	GPIP6	Timer A	RCV Buffer Full	RCV Error	XMIT Buffer Empty	XMIT Error	Timer B	IMRA
	7	6	5	4	3	2	1	0	
$15	GPIP5	GPIP4	Timer C	Timer D	GPIP3	GPIP2	GPIP1	GPIP0	IMRB
Reset:	0	0	0	0	0	0	0	0	

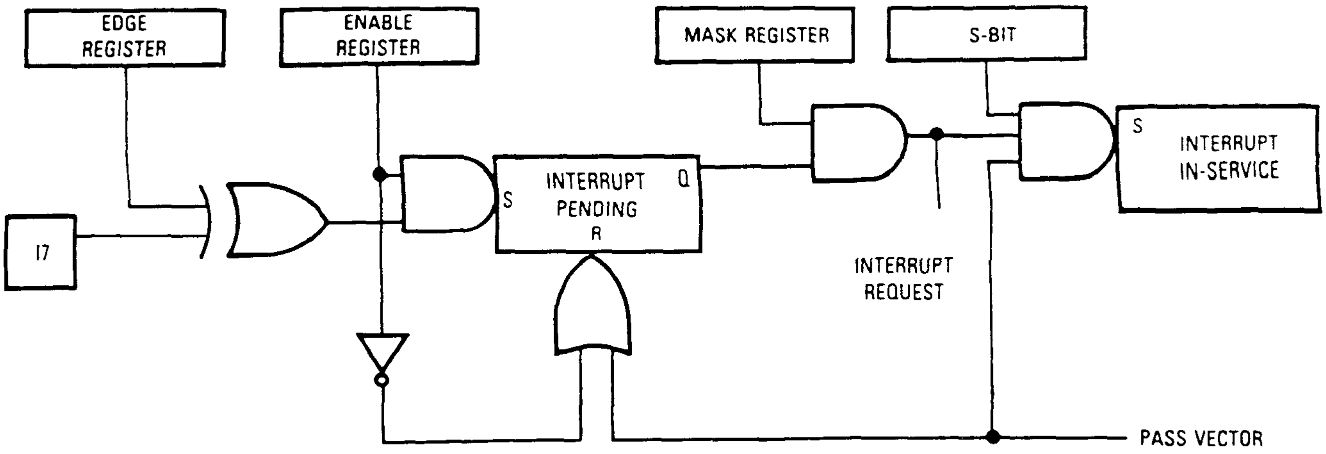

Figure 3-2. Conceptual Circuits of an Interrupt Channel

3.3.4 Interrupt In-Service Registers (ISRA, ISRB)

These registers indicate whether interrupt processing is in progress for a certain channel. A bit is set whenever an interrupt vector number is passed for a interrupt channel during an IACK cycle and the S bit of the vector register is a one. The bit is cleared whenever interrupt service is complete for an associated interrupt channel, the S bit of the vector register is cleared, or the processor writes a zero to the bit.

	7	6	5	4	3	2	1	0	
$0F	GPIP7	GPIP6	Timer A	RCV Buffer Full	RCV Error	XMIT Buffer Empty	XMIT Error	Timer B	ISRA

	7	6	5	4	3	2	1	0	
$11	GPIP5	GPIP4	Timer C	Timer D	GPIP3	GPIP2	GPIP1	GPIP0	ISRB
Reset:	0	0	0	0	0	0	0	0	

3.4 NESTING MFP INTERRUPTS

In an M68000 vectored interrupt system, the MFP is assigned to one of seven possible interrupt levels. When an interrupt is received from the MFP, an interrupt acknowledge for that level is initiated. Once an interrupt is recognized at a particular level, interrupts at the same level or below are masked by the processor. As long as the processor's interrupt mask is unchanged, the M68000 interrupt structure will prohibit nesting the interrupts at the same interrupt level. However, additional interrupt requests from the MFP can be recognized before a previous channel's interrupt service routine is finished by lowering the processor's interrupt mask to the next lower interrupt level within the interrupt handler.

When nesting MFP interrupts, it may be desirable to permit interrupts on any MFP channel regardless of its priority, to preempt or delay interrupt processing of an earlier channel's interrupt service request. Or, it may be desirable to only allow subsequent higher priority channel interrupt requests to supercede previously recognized lower priority interrupt requests. The MFP interrupt structure provides the flexibility by offering two end-of-interrupt options for vectored interrupt schemes. Note that the end-of-interrupt modes are not active in a polled interrupt scheme.

3.4.1 Selecting The End-Of-Interrupt Mode

In a vectored interrupt scheme, the MFP may be programmed to operate in either the automatic end-of-interrupt mode or the software end-of-interrupt mode. The mode is selected by writing the S bit of the vector register. When the S bit is programmed to a one, the MFP is placed in the software end-of-interrupt mode, and when the S bit is a zero, all channels operate in the automatic end-of-interrupt mode.

3.4.2 Automatic End-Of-Interrupt Mode

When an interrupt vector is passed to the processor during an interrupt acknowledge cycle, the corresponding channel's interrupt pending bit is cleared. In the automatic end-of-interrupt mode, no further history of the interrupt remains in the MFP. The in-service bits of the interrupt in-service registers (ISRA and ISRB) are forced low. Subsequent interrupts, which are received on any MFP channel will generate an interrupt request to the processor even if the current interrupt's service routine has not been completed.

3.4.3 Software End-Of-Interrupt Mode

In the software end-of-interrupt mode, the channel's associated interrupt pending bit is cleared. In addition, the channel's in-service bit of in-service register A or B is set when its vector number is passed to the processor during the interrupt acknowledge cycle. A higher priority channel may subsequently request interrupt service and be acknowledged, but as long as the channel's in-service bit is set, no lower priority channel may request interrupt service nor pass its vector during an interrupt acknowledge sequence.

While only higher priority channels may request interrupt service, any channel can receive an interrupt and set its interrupt pending bit. Even the channel with its in-service bit is set can receive a second interrupt. However, no interrupt service request is made until its in-service bit is cleared.

The in-service bit for a particular channel can be cleared by writing a zero to its corresponding bit in ISRA or ISRB and ones to all other bit positions. Since bits in the in-service registers can only be cleared in software and not set, writing ones to the registers does not alter their contents. ISRA and ISRB may be read at any time.

SECTION 4
GENERAL PURPOSE INPUT/OUTPUT PORT

The general purpose input/output (I/O) port (GPIP) provides eight I/O lines (I0 through I7) that may be operated as either inputs or outputs under software control. In addition, these lines may optionally generate an interrupt on either a positive transition or a negative transition of the input signal. The flexibility of the GPIP allows it to be configured as an 8-bit I/O port or for bit I/O. Since interrupts are enabled on a bit-by-bit basis, a subset of the GPIP could be programmed as handshake lines or the port could be connected to as many as eight external interrupt sources, which would be prioritized by the MFP interrupt controller for the interrupt service.

4.1 M6800 INTERRUPT CONTROLLER

The MFP interrupt controller is particularly useful in a system which has many M6800-type devices. Typically, in a vectored M68000 system, M6800 peripherals use the autovector which corresponds to their assigned interrupt level since they can not provide a vector number in response to an interrupt acknowledge cycle. The autovector interrupt handler must then poll all M6800 devices at that interrupt level to determine which device is requesting service. However, by tying the $\overline{\text{IRQ}}$ output from an M6800 peripheral to the general purpose I/O port (GPIP) of an MFP, a unique vector number will be provided to the processor during an interrupt acknowledge cycle. This interrupt structure will significantly reduce interrupt latency for M6800 devices and other peripherals which do not support vectored interrupts.

4.2 GPIP CONTROL REGISTERS

The GPIP is programmed via three control registers. These registers control the data direction, provide user access to the port, and specify the active edge for each bit of the GPIP which will produce an interrupt. These registers are described in detail in the following paragraphs.

4.2.1 General Purpose I/O Data Register (GPDR)

The general purpose I/O data register is used to input data from or output data to the port. When data is written to the GPDR, those pins which are defined as inputs will remain in the high-impedance state. Pins which are defined as outputs will assume the state (high or low) of their corresponding bit in the data register. When the GPDR is read, data will be passed directly from the bits of the data register for pins which are defined as outputs. Data from pins defined as inputs will come from the input buffers.

	7	6	5	4	3	2	1	0	
$01	GPIP7	GPIP6	GPIP5	GPIP4	GPIP3	GPIP2	GPIP1	GPIP0	GPDR
Reset:	0	0	0	0	0	0	0	0	

4.2.2 Active Edge Register (AER)

The active edge register (AER) allows each of the GPIP lines to produce an interrupt on either a one-to-zero or a zero-to-one transition. Writing a zero to the appropriate edge bit of the active edge register will cause the associated input to generate an interrupt on the one-to-zero transition. Writing a one to the edge bit will produce an interrupt on the zero-to-one transition of the corresponding line. When the processor sets a bit, interrupts will be generated on the rising edge of the associated input signal. When the processor clears a bit, interrupts will be generated on the falling edge of the associated input signal.

	7	6	5	4	3	2	1	0	
$03	GPIP7	GPIP6	GPIP5	GPIP4	GPIP3	GPIP2	GPIP1	GPIP0	AER
Reset:	0	0	0	0	0	0	0	0	

NOTE

> The inputs to the exclusive-OR of the transition detector are the edge bit and the input buffer. As a result, writing the AER may cause an interrupt-producing transition, depending upon the state of the input. So, the AER should be configured before enabling interrupts via the interrupt enable registers (IERA and IERB). Also, changing the edge bit while interrupts are enabled may cause an interrupt on the corresponding channel.

4.2.3 Data Direction Register (DDR)

The data direction register (DDR) allows the programmer to define I0 through I7 as inputs or outputs by writing the corresponding bit. When a bit of the data direction register is written as a zero, the corresponding interrupt I/O pin will be a high-impedance input. Writing a one to any bit of the data direction register will cause the corresponding pin to be configured as a push-pull output.

	7	6	5	4	3	2	1	0	
$05	GPIP7	GPIP6	GPIP5	GPIP4	GPIP3	GPIP2	GPIP1	GPIP0	DDR
Reset:	0	0	0	0	0	0	0	0	

SECTION 5
TIMERS

The MFP contains four 8-bit timers which provide many functions typically required in microprocessor systems. The timers can supply the baud rate clocks for the on-chip serial I/O channel, generate periodic interrupts, measure elapsed time, and count signal transitions. In addition, two timers have waveform generation capability.

All timers are prescaler/counter timers with a common independent clock input (XTAL1 and XTAL2) and are not required to be operated from the system clock. Each timer's output signal toggles when the timer's main counter times out. Additionally, timers A and B have auxiliary control signals which are used in two of the operation modes. An interrupt channel is assigned to each timer, and when the auxiliary control signals are used in the pulse width measurement mode, a separate interrupt channel will respond to transitions on these inputs.

5.1 OPERATION MODES

Timers A and B are full function timers which, in addition to the delay mode, operate in the pulse width measurements mode and the event count mode. Timers C and D are delay timers only. A brief discussion of each of the timer modes follows.

5.1.1 Delay Mode Operation

All timers may operate in the delay mode. In this mode, the prescaler is always active. The prescaler specifies the number of timer clock cycles which must elapse before a count pulse is applied to the main counter. A count pulse causes the main counter to decrement by one. When the timer has decremented down to 01 (hexadecimal), the next count pulse will cause the main counter to be reloaded from the timer data register and a time out pulse will be produced. This time out pulse is coupled to the timer's interrupt channel and, if the channel is enabled, an interrupt will occur. The time out pulse also causes the timer output pin to toggle. The output will remain in this new state until the next time out pulse occurs.

For example, if delay mode with a divide-by-10 prescaler is selected and the timer data register is loaded with 100 (decimal), the main counter will decrement once every 10 timer clock cycles. After 1000 timer clocks, a time out pulse will be produced. This time out pulse will generate an interrupt if the channel is enabled (IERA, IERB), and in addition, the timer's output line will toggle. The output line will complete one full period every 2000 cycles of the timer clock.

If the prescaler value is changed while the timer is enabled, the first time out pulse will occur at an indeterminate time no less than one nor more than 200 timer clock cycles. Subsequent time out pulses will then occur at the correct interval.

If the main counter is loaded with 01 (hexadecimal), a time out pulse will occur every time the prescaler presents a count pulse to the main counter. If the main counter is loaded with 00, a time out pulse will occur every 256 count pulses.

5.1.2 Pulse Width Measurement Operation

Besides the delay mode, timers A and B may be programmed to operate in the pulse width measurement mode. In this mode, an auxiliary control input is required; timers A and B auxiliary input lines are TAI and TBI. Also, in the pulse width measurement mode, interrupt channels normally associated with I4 and I3 will instead respond to transitions on TAI and TBI, respectively. General purpose lines I3 and I4 may still be used for I/O, but may not be used as interrupt generating inputs. A conceptual circuit of the selection of the interrupt source is shown in Figure 5-1.

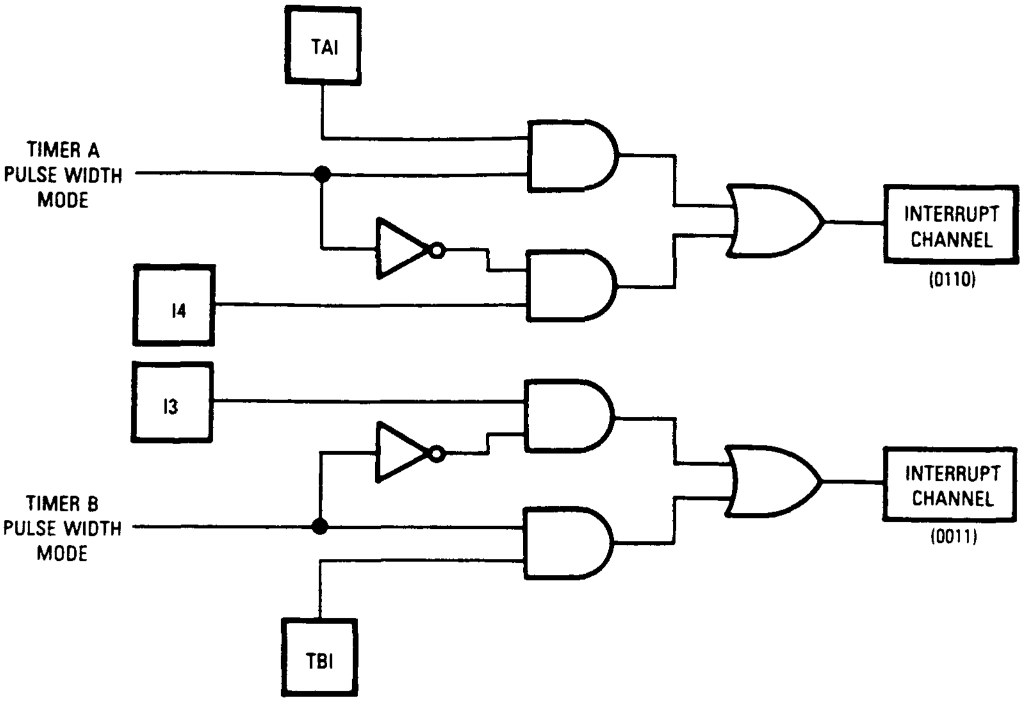

Figure 5-1. Conceptual Circuit of Interrupt Source Selection

The pulse width measurement mode functions similarly to the delay mode, with the auxiliary control signal acting as an enable to the timer. When the control signal is active, the prescaler and main counter are allowed to operate. When the control signal is negated, the timer is stopped. So, the width of the active pulse on TAI or TBI is measured by the number of timer counts which occur while the timer is allowed to operate.

The active state of the auxiliary input line is defined by the associated interrupt channel's edge bit in the active edge register (AER). GPIP4 of the AER is the edge bit associated with TAI, and GPIP3 is associated with TBI. When the edge bit is a one, the auxiliary input will be active high, enabling the timer while the input signal is at a high level. If the edge bit is zero, the auxiliary input will be active low and the timer will operate while the input signal is at a low level.

The state of the active edge bit also specifies whether a zero-to-one transition or a one-to-zero transition of the auxiliary input pin will produce an interrupt when the interrupt channel is enabled. In normal operation, programming the active edge bit to a one will produce an interrupt on the zero-to-one transition of the associated input signal. Alternately, programming the edge bit to a zero will produce an interrupt on the one-to-zero transition of the input signal. However, in the pulse width measurement mode, the interrupt generated by a transition on TAI or TBI will occur on the opposite transition as that normally defined by the edge bit.

For example, in the pulse width measurement mode, if the edge bit is a one, the timer will be allowed to run while the auxiliary input is high. When the input transitions from high to low, the timer will stop and, if the interrupt channel is enabled, an interrupt will occur. By having the interrupt occur on the one-to-zero transition instead of the zero-to-one transition, the processor will be interrupted when the pulse being measured has been terminated and the width of the pulse from the timer is available.

After reading the contents of the timer, the processor must reinitialize the main counter by writing to the timer data register to allow consecutive pulses to be measured. If the data register is written after the auxiliary input signal becomes active, the timer will count from the previous contents of the timer data register until it counts through 01 (hexadecimal). At that time, the main counter is loaded with the new value from the timer data register, a time out pulse is generated which will toggle the timer output, and an interrupt may be optionally generated on the timer interrupt channel. Note that the pulse width measured includes counts from before the main counter was reloaded. If the timer data register is written while the pulse is transitioning to the active state, an indeterminate value may be written into the main counter .

Once the timer is reprogrammed for another mode, interrupts will again occur as normally defined by the edge bit. Note that an interrupt may be generated as the result of placing the timer into the pulse width measurement mode or by reprogramming the timer for another mode. Also, an interrupt may be generated by changing the state of the edge bit while in the pulse width measurement mode.

5.1.3 Event Count Mode Operation

In addition to the delay mode and the pulse width measurement mode, timers A and B may be programmed to operate in the event count mode. Like the pulse width measurement mode, the event count mode requires an auxiliary input signal, TAI or TBI. General purpose lines I3 and I4 can be used for I/O or as interrupt producing inputs.

In the event count mode, the prescaler is disabled allowing each active transition on TAI and TBI to produce a count pulse. The count pulse causes the main counter to decrement by one. When the timer counts through 01 (hexadecimal), a time out pulse is generated which will cause the output signal to toggle and may optionally produce an interrupt via the associated timer interrupt channel. The timer's main counter is also reloaded from the timer data register. To count transitions reliably, the input signal may only transition once every four timer clock periods. For this reason, the input signal must have a maximum frequency of one-fourth that of the timer clock.

The active edge of the auxiliary input signal is defined by the associated channel's edge bit. GPIP4 of the AER specifies the active edge for TAI, and GPIP3 defines the active edge for TBI. When the edge bit is programmed to a one, a count pulse will be generated on the zero-to-one transition of the auxiliary input signal. When the edge bit is programmed to a zero, a count pulse will be generated on the one-to-zero transition. Also, note that changing the state of the edge bit while the timer is in the event count mode may produce a count pulse.

5.2 TIMER REGISTERS

The four timers are programmed via three control registers and four data registers. The following paragraphs describe the different registers.

5.2.1 Timer Data Registers (TxDR)

The four timer data registers (TDRs) are designed as Timer A data register (TADR), Timer B (TBDR), Timer C (TCDR), and Timer D (TDDR). Each timer's main counter is an 8-bit binary down counter. The timer data registers contain the value of their respective main counter. This value was captured on the last low-to-high transition of the data strobe pin.

The main counter is initialized by writing to the TDR. If the timer is stopped, data is loaded simultaneously into both the TDR and main counter. If the TDR is written to while the timer is enabled, the value is not loaded into the timer until the timer counts through 01 (hexadecimal). If a write is performed while the timer is counting through 01, then an indeterminate value will be loaded into the timer's main counter.

Typical Timer Data Register

	7	6	5	4	3	2	1	0	
$1F, $21, $23, $25	D7	D6	D5	D4	D3	D2	D1	D0	TADR, TBDR, TCDR, TDDR

5.2.2 Timer Control Registers (TxCR)

Timer control register A (TACR) and timer control register B (TBCR) are associated with timers A and B, respectivey. Timers C and D are programmed using one control register—the timer C and D control register (TCDCR). The bits in the control register select the operation mode, prescaler value, and disable the timers. Both control registers have bits which allow the programmer to reset output lines TA0 and TB0.

	7	6	5	4	3	2	1	0	
$19	•	•	•	Reset TAO	AC3	AC2	AC1	AC0	TACR

	7	6	5	4	3	2	1	0	
$1B	•	•	•	Reset TB0	BC3	BC2	BC1	BC0	TBCR
Reset:	0	0	0	0	0	0	0	0	

•Unused bits read as zero.

Reset TA0/TB0 — Reset Timer A and B Output Lines.

TA0 and TB0 may be forced low at any time by writing a one to the reset location in TACR and TBCR. Output is held low during the write operation, and at the end of the bus cycle the output is allowed to toggle in response to a time-out pulse. When resetting TA0 and TB0, the other bits in the TCR must be written with their previous value to avoid altering the operating mode.

AC3-AC0, BC3-BC0 — Select Timer A and B Operation Mode.

When the timer is stopped, counting is inhibited. The contents of the timer's main counter is not affected, although any residual count in the prescaler.

AC3 BC3	AC2 BC2	AC1 BC1	AC0 BC0	Operation Mode
0	0	0	0	Timer Stopped
0	0	0	1	Delay Mode, /4 Prescaler
0	0	1	0	Delay Mode, /10 Prescaler
0	0	1	1	Delay Mode, /16 Prescaler
0	1	0	0	Delay Mode, /50 Prescaler
0	1	0	1	Delay Mode, /64 Prescaler
0	1	1	0	Delay Mode, /100 Prescaler
0	1	1	1	Delay Mode, /200 Prescaler
1	0	0	0	Event Count Mode
1	0	0	1	Pulse Width Mode, /4 Prescaler
1	0	1	0	Pulse Width Mode, /10 Prescaler
1	0	1	1	Pulse Width Mode, /16 Prescaler
1	1	0	0	Pulse Width Mode, /50 Prescaler
1	1	0	1	Pulse Width Mode, /64 Prescaler
1	1	1	0	Pulse Width Mode, /100 Prescaler
1	1	1	1	Pulse Width Mode, /200 Prescaler

	7	6	5	4	3	2	1	0	
$1D	*	CC2	CC1	CC0	**	DC2	DC1	DC0	TCDCR
Reset:	0	0	0	0	0	0	0	0	

*Unused bits read as zero.

CC2-CC0, DC3-DC0 — Select Timer C and D Operation Mode.

When the timer is stopped, counting is inhibited. The contents of the timer's main counter is not affected, although any residual count in the prescaler is lost.

CC2 DC2	CC1 DC1	CC0 DC0	Operation Mode
0	0	0	Timer Stopped
0	0	1	Delay Mode, /4 Prescaler
0	1	0	Delay Mode, /10 Prescaler
0	1	1	Delay Mode, /16 Prescaler
1	0	0	Delay Mode, /50 Prescaler
1	0	1	Delay Mode, /64 Prescaler
1	1	0	Delay Mode, /100 Prescaler
1	1	1	Delay Mode, /200 Prescaler

SECTION 6
UNIVERSAL SYNCHRONOUS/ASYNCHRONOUS RECEIVER-TRANSMITTER

The universal synchronous asynchronous receiver-transmitter (USART) is a single, full-duplex serial channel with a double-buffered receiver and transmitter. There are separate receive and transmit clocks and, also, separate receive and transmit status and data bytes. The receive and transmit sections are also assigned separate interrupt channels. Each section has two interrupt channels: one for normal conditions and the other for error conditions. All interrupt channels are edge-triggered. Generally, it is the output of a flag bit or bits which is coupled to the interrupt channel. Thus, if an interrupt-producing event occurs while the associated interrupt channel is disabled, no interrupt would be produced, even if the channel was subsequently enabled because a transition did not occur while the channel was enabled. That particular event would have to occur again, generating another edge, before an interrupt would be generated. The interrupt channels may be disabled and instead, a DMA device can be used to transfer the data via the control signals receiver ready ($\overline{\text{RR}}$) and transmitter ready ($\overline{\text{TR}}$). See **6.4 DMA OPERATION** for more information.

6.1 CHARACTER PROTOCOLS

The MFP USART supports asynchronous and, with the help of a polynomial generator checker, byte synchronous character formats. These formats are selected independently of the divide-by-one and divide-by-18 clock modes. It is possible to clock data synchronously into the MC68901 but still use start and stop bits. After a start bit is detected, data will be shifted in and a stop bit will be checked to determine proper framing. In this mode, all normal asynchronous format features apply.

When the divide-by-one clock mode is selected, synchronization must be accomplished externally. The receiver will sample serial data on the rising edge of the receiver clock. In the divide-by-18 clock mode, the data is sampled at mid-bit time to increase transient noise rejection.

Also, when the divide-by-18 clock mode is selected, the USART resynchronization logic is enabled. This logic increases the channels clock skew tolerance. Refer to **6.1.1 Asynchronous Format** for more information on the resynchronization logic.

6.1.1 Asynchronous Format

Variable character length and start/stop bit configurations are available under software control for asynchronous operation. The user can choose a character length from five to eight bits and a stop bit length of one, one and one-half, or two bits. The user can also select odd, even, or no parity.

In the asynchronous format, start bit detection is always enabled. New data is not shifted into the receive shift register until a zero bit is received. When the divide-by-18 clock mode is selected, the false start bit logic is also active. Any transition must be stable for three positive receive clock edges to be considered valid. For a start bit to be good, a valid zero-to-one transition must not occur for eight positive receiver clock transitions after the initial one-to-zero transition. After a valid start bit has been detected, the data is checked continuously for valid transitions. When a valid transition is detected, an internal counter is forced to state zero, and no further transition checking is initiated until state four. At state eight, the "previous state" of the transition checking logic is clocked into the receiver. As a result of this resynchronization logic, it is possible to run with asynchronous clocks without start and stop bits if there are sufficient valid transitions in the data streams.

6.1.2 Synchronous Format

When the synchronous character format is selected, the 8-bit synchronous character loaded into the synchronous character register (SCR) is compared to received serial data until a match is found. Once synchronization is established, incoming data is clocked into the receiver. The synchronous word will be continuously transmitted during an underrun condition. All synchronous characters can be optionally stripped from the receive buffer (i.e. taken out of the data stream and thrown away) by clearing the appropriate bit in the receive status register (RSR).

	7	6	5	4	3	2	1	0	
$27	D7	D6	D5	D4	D3	D2	D1	D0	SCR
Reset:	0	0	0	0	0	0	0	0	

The synchronous character should be written after the character length is selected, since unused bits in the synchronous character register are zeroed out. When parity is enabled, synchronous word length is the character length plus one. The MFP will compute and append the parity bit for the synchronous character when a character length of eight is selected. However, if the character length is less than eight, the user must determine the synchronous word parity and write it into the synchronous character register along with the synchronous character. The parity bit must be the most-significant bit. The MFP will then transmit the extra bit in the synchronous word as a parity bit.

6.1.3 USART Control Register (UCR)

This register selects the clock mode and the character format for the receive and transmit sections.

	7	6	5	4	3	2	1	0	
$29	CLK	CL1	CL0	ST1	ST0	PE	E/O	*	UCR
Reset:	0	0	0	0	0	0	0	U	

*Unused bits read as zero.

CLK — Clock Mode
1 = Data clocked into and out of receiver and transmitter at one sixteenth the frequency of their respective clocks.
0 = Data clocked into and out of receiver and transmitter at the frequency of their respective clocks.

CL1,CL0 — Character Length
These bits specify the length of the character exclusive of start bits, and parity.

CL1	CL0	Character Length
0	0	8 Bits
0	1	7 Bits
1	0	6 Bits
1	1	5 Bits

ST0-ST1 — Start/Stop Bit and Format Control

These bits select the number of start and stop bits and specify the character format.

ST1	ST0	Start Bits	Stop Bits	Format
0	0	0	0	Synchronous
0	1	1	1	Asynchronous
1	0	1	1-1/2	Asynchronous*
1	1	1	2	Asynchronous

*Used with Divide-by-16 mode only.

PE — Parity Enable

Parity is not automatically appended to the synchronous character for character lengths of less than eight bits. Therefore, parity should be written into the SCR along with the synchronous character.

1 = Parity checked by receiver and parity calculated and inserted during data transmission.
0 = No parity check and no parity bit computed for transmission.

E/O — Even/Odd Parity

1 = Even parity is selected.
0 = Odd parity is selected.

Bit 0 — Not used

6.2 RECEIVER

As data is received on the serial input line (SI), it is clocked into an internal 8-bit shift register until the specified number of data bits have been assembled. The character will then be transferred to the receive buffer, assuming that the last word in the receive buffer has been read. This transfer will set the buffer full bit in the Receiver Status Register (RSR) and produce a buffer full interrupt to the processor, assuming this interrupt has been enabled.

Reading the receive buffer satisfies the buffer full condition and allows a new data word to be transferred to the receive buffer when it is assembled. The receive buffer is accessed by reading the USART data register (UDR). The UDR is simply an 8-bit data register used when transferring data between the MFP and the CPU.

Each time a word is transferred to the receive buffer, its status information is latched into the receiver status register (RSR). The RSR is not updated again until the data word in the receive buffer has been read. When a buffer full condition exists, the RSR should always be read before the receive buffer (UDR) to maintain the correct correspondence between data and flags. Otherwise, it is possible that after reading the UDR and prior to reading the RSR, a new word could be received and transferred to the receive buffer. Its associated flags would be latched into the RSR, writing over the flags for the previous data word. Thus, when the RSR was read to access the status information for the first data word, the flags for the new word would be retrieved.

6.2.1 Receiver Interrupt Channels

The USART receiver section is assigned two interrupt channels. One indicates the buffer full condition while the other channel indicates an error condition. Error conditions include overrun, parity error, frame error, synchronous found, and break. These interrupting conditions correspond to the OE, PE, FE, and F/S or B bits of the receiver status register. These flags will function as described in **6.2.2 Receiver Status Register** whether the receiver interrupt channels are enabled or disabled.

While only one interrupt is generated per character received, two dedicated interrupt channels allow separate vector numbers to be assigned for normal and abnormal receiver conditions. When a received word has an error associated with it and the error interrupt channel is enabled, an interrupt will be generated on the error channel only. However, if the error channel is disabled, an interrupt for an error condition will be generated on the buffer full interrupt channel along with interrupts produced by the buffer full condition. The receiver status register must always be read to determine which error condition produced the interrupt.

6.2.2 Receiver Status Register (RSR)

The RSR contains the receiver buffer full flag, the synchronous strip enable, the various status information associated with the data word in the receive buffer. The RSR is latched each time a data word is transferred to the receive buffer. RSR flags cannot change again until the new data word has been read. However, the M/CIP bit is allowed to change.

	7	6	5	4	3	2	1	0	
$2B	BF	OE	PE	FE	F/S or B	M/CIP	SS	RE	RSR
Reset:	0	0	0	0	0	0	0	0	

BF — Buffer Full

Receiver word is transferred to the receive buffer.
Receiver buffer is read by accessing the USART data register.

OE — Overrun Error

Overrun error occurs when a received word is to be transferred to the receive buffer, but the buffer is full. Neither the receiver buffer nor the RSR is overwritten.

1 = Receiver buffer full.
0 = Read by RSR or MFP reset.

PE — Parity Error

1 = Parity error detected on character transfer to receiver buffer.
0 = No parity error detected on character transfer to receiver buffer.

FE — Frame Error

A frame error exists when a non-zero data character is not followed by a stop bit in the asynchronous character format.

1 = Frame error detected on character transfer to receiver buffer.
0 = No frame error detected on character transfer to receiver buffer.

F/S or B — Found/Search or Break Detect

The F/S bit is used in the synchronous character format. When set to zero, the USART receiver is placed in the search mode. F/S is cleared when the incoming character does not match the synchronous character.

1 = Match found. Character length counter enabled.

0 = Incoming data compared to SCR. Character length counter disabled.

The B bit is used in the asynchronous character format. This flag indicates a break condition which continues until a non-zero data bit is received.

1 = Character transferred to the receive buffer is a break condition.

0 = Non-zero data bit received and break condition was acknowledged by reading the RSR at least once.

M or CIP — Match/Character in Progress

The M bit is used in the synchronous character format and indicates a synchronous character has been received.

1 = Character transferred to the receive buffer matches the synchronous character.

0 = Character transferred to the receive buffer does not match the synchronous character.

The CIP bit is used in the asynchronous character format and indicates that a character is being assembled.

1 = Start bit is detected.

0 = Final stop bit has been received.

SS — Synchronous Strip Enable

1 = Characters that match the synchronous character will not be loaded into the receiver buffer and no buffer full condition will be produced.

0 = Characters that match the synchronous character will be transferred to the receive buffer and a buffer full condition will be produced.

RE — Receiver Enable

This bit should not be set until the receiver clock is active. When the transmitter is disabled in auto-turnaround mode this bit is set.

1 = Receiver operation is enabled.

0 = Receiver is disabled.

6.2.3 Special Receive Conditions

Certain receive conditions relating to the overrun error flag and the break detect flag require further explanation. Consider the following examples:

1) A break is received while the receive buffer is full. This does not produce an overrun condition. Only the B flag will be set after the receiver buffer is read.
2) A new word is received, and the receive buffer is full. A break is received before the receive buffer is read.

Both the B and OE flags will be set when the buffer full condition is satisfied.

6.3 TRANSMITTER

The transmit buffer is loaded by writing to the USART data register (UDR). The data character will be transferred to an internal 8-bit shift register when the last character in the shift register has been transmitted. This transfer will produce a buffer empty condition. If the transmitter completes the transmission of the character in the shift register before a new character is written to the transmit buffer, an underrun error will occur. In the asynchronous character format, the transmitter will send a mark until the transmit buffer is written. In the synchronous character format, the transmitter will continuously send the synchronous character until the transmit buffer is written.

The transmit buffer can be loaded prior to enabling the transmitter. After the transmitter is enabled, there is a delay before the first bit is output. The serial output line (SO) should be programmed to be high, low, or high impedance (by setting the appropriate bits in the Transmitter Status Register (TSR)) before the transmitter is enabled forcing the output line to the desired state until the first bit of the first character is shifted out. The state of the H and L bits in the TSR determine the state of the first transmitted bit after the transmitter is enabled. If the high impedance mode is selected prior to the transmitter being enabled, the first bit transmitted is indeterminate. Note that the SO line will always be driven high for one bit time prior to the character in the transmit shift register being transmitted when the transmitter is first enabled.

When the transmitter is disabled, any character currently being transmitted will continue to completion. However, any character in the transmit buffer will not be transmitted and will remain in the buffer. Thus, no buffer empty condition will occur. If the buffer is empty when the transmitter is disabled, the buffer empty condition will remain, but no underrun condition will be generated when the character in transmission is completed. If no character is being transmitted when the transmitter is disabled, the transmitter will stop at the next rising edge of the internal shift clock.

In the asynchronous character format, the transmitter can be programmed to send a break. The break will be transmitted once the word currently in the shift register has been sent. If the shift register is empty, the break command will be effective immediately. A transmit error interrupt will be generated at every normal character boundary to aid in timing the break transmission. The contents of the TSR are not affected, however. The break will continue until the break bit is cleared. The underrun error (UE) must be cleared from the TSR. Also, the interrupt pending register must be cleared of pending transmitter errors at the beginning of the break transmission, or no interrupts will be generated at the character boundary time. The break (B) bit cannot be set until the transmitter has been enabled and has had sufficient time (one transmitter clock cycle) to perform internal reset and initialization functions.

Any character in the transmit buffer at the start of a break will be transmitted when the break is terminated, assuming the transmitter is still enabled. If the transmit buffer is empty at the start of a break, it may be written at any time during the break. If the buffer is still empty at the end of the break, an underrun condition will exist.

Disabling the transmitter during a break condition causes the transmitter to cease transmission of the break character at the end of the current character. No end-of-break stop bit will be transmitted. Even if the transmit buffer is empty, no buffer empty condition will occur nor will an underrun condition occur. Also, any word in the transmit buffer will remain.

6.3.1 Transmitter Interrupt Channels

The USART transmit section is assigned two interrupt channels. The normal channel indicates a buffer empty condition, and the error channel indicates an underrun or end condition. These interrupting conditions correspond to the BE, UE, and END flags in the TSR. The flag bits will function as described in **6.3.2 Transmitter Status Register** whether their associated interrupt channel is enabled or disabled.

6.3.2 Transmitter Status Register

The TSR contains various transmitter error flags and transmitter control bits for selecting auto-turnaround and loopback mode.

	7	6	5	4	3	2	1	0	
$2D	BE	UE	AT	END	B	H	L	TE	TSR
Reset:	0	0	0	0	0	0	0	0	

BE — Buffer Empty

1 = Character in the transmit buffer transferred to TSR.
0 = Transmit buffer reloaded by writing to the USR.

U — Underrun Error

One full transmitter clock cycle is required after UE bit is set before it can be cleared. This bit does not require clearing before writing to the UDR.
1 = Character in the TSR was transmitted before a new word was loaded into the transmit buffer.
0 = Transmitter disabled or read performed on TSR.

AT — Auto-Turnaround

When set, the receiver will be enabled automatically after the transmitter has been disabled and the last character being transmitted is complete.

END — End of Transmission

If the transmitter is disabled while a character is being transmitted, this bit is set after transmission is complete. If no character was being transmitted, then this bit is set immediately. Reenabling the transmitter clears this bit.

B — Break

This bit only functions in the asynchronous format. When B is set, BE cannot be set. A break consists of all zeros with no stop bit. This bit cannot be set until transmitter is enabled and internal reset and initialization is complete.
1 = Break transmitted and transmission stops.
0 = Break ceases and normal transmission resumes.

H, L — High and Low

These bits configure the transmitter output (SO) when the transmitter is disabled. Changing these bits after the transmitter is enabled will alter the output state until END is cleared.

H	L	Output State
0	0	High Impedance
0	1	Low
1	0	High
1	1	Loopback Mode

TE — Transmitter Enable

The serial output will be driven according to H and L bits until transmission begins. A one bit is transmitted before character transmission in the TSR begins.
1 = Transmitter enabled.
0 = Transmitter disabled. UE bit cleared and END bit set.

6.4 DMA OPERATION

USART error conditions are valid only for each character boundary. When the USART performs block data transfers by using the DMA handshake lines receiver ready ($\overline{\mathrm{RR}}$) and transmitter ready ($\overline{\mathrm{TR}}$), errors must be saved and checked at the end of a block. This is accomplished by enabling the error channel for the receiver or transmitter and by masking interrupts for this channel. Once the transfer is complete, interrupt pending register A is read. Any pending receiver or transmitter error indicates an error in the data transfer.

$\overline{\mathrm{RR}}$ is asserted when the buffer full bit is set in the RSR unless a parity error or frame error is detected by the receiver. $\overline{\mathrm{TR}}$ is asserted when the buffer empty bit is set in the TSR unless a break is currently being transmitted.