Welcome to E-XFL.COM

Understanding Embedded - Microprocessors

Embedded microprocessors are specialized computing chips designed to perform specific tasks within an embedded system. Unlike general-purpose microprocessors found in personal computers, embedded microprocessors are tailored for dedicated functions within larger systems, offering optimized performance, efficiency, and reliability. These microprocessors are integral to the operation of countless electronic devices, providing the computational power necessary for controlling processes, handling data, and managing communications.

Applications of Embedded - Microprocessors

Embedded microprocessors are utilized across a broad spectrum of applications, making them indispensable in

Details	
Product Status	Obsolete
Core Processor	68000
Number of Cores/Bus Width	1 Core, 32-Bit
Speed	25MHz
Co-Processors/DSP	-
RAM Controllers	-
Graphics Acceleration	No
Display & Interface Controllers	-
Ethernet	-
SATA	-
USB	-
Voltage - I/O	5.0V
Operating Temperature	-40°C ~ 85°C (TC)
Security Features	-
Package / Case	114-BCPGA
Supplier Device Package	114-CPGA (34.54x34.54)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/ts68020vr25

The micromachine consists of an execution unit, nanorom and microrom storage, an instruction decoder, an instruction pipe, and associated control sections. The execution unit consists of an address section, an operand address section, and a data section. Microcode control is provided by a modified two-level store of microrom and nanorom. Programmed logical arrays (PLAs) are used to provide instruction decode and sequencing information. The instruction pipe and other individual control sections provide the secondary decode of instructions and generated the actual control signals that result in the decoding and interpretation of nanorom and micorom information.

Figure 2. PGA Terminal Designation

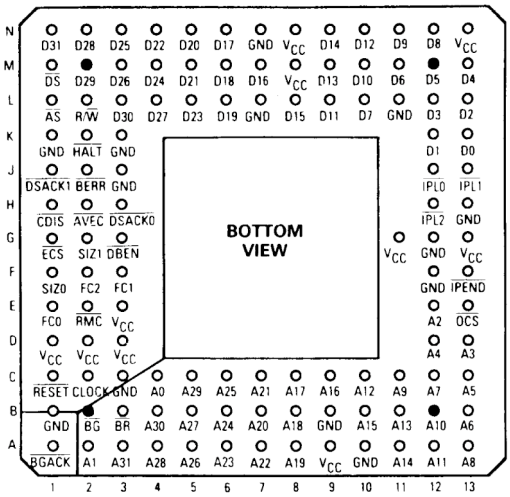


Figure 3. CQFP Terminal Designation

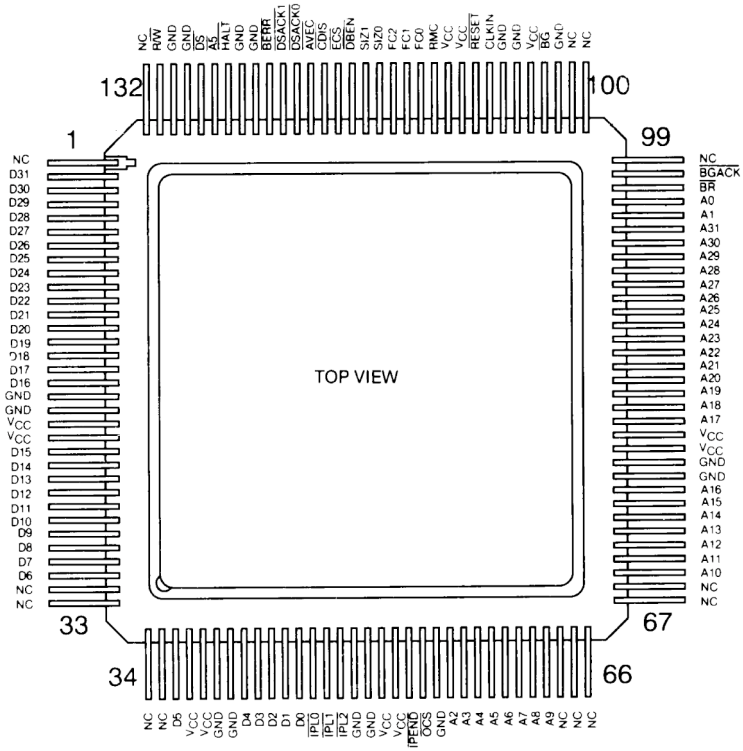
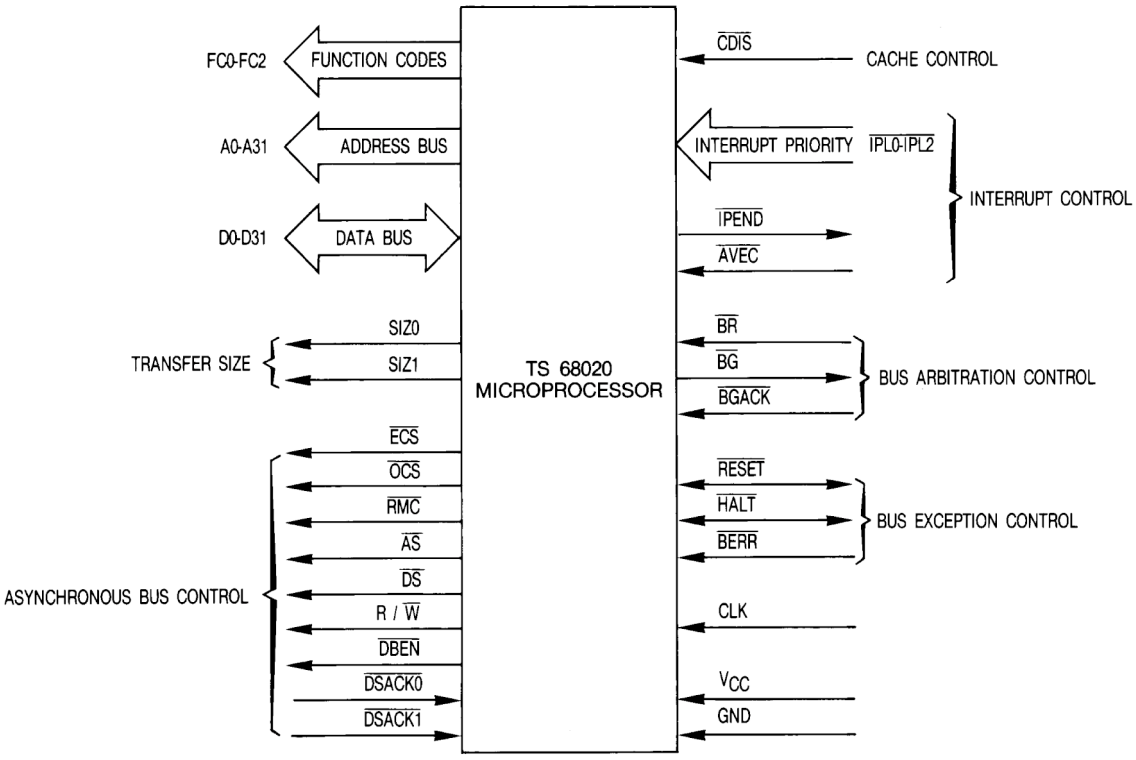


Figure 4. Functional Signal Groups



Signal Description

Figure 4 illustrates the functional signal groups and Table 1 lists the signals and their function.

The V_{CC} and GND pins are separated into four groups to provide individual power supply connections for the address bus buffers, data bus buffers, and all other output buffers and internal logic.

Group	V_{CC}	GND
Address Bus	A9, D3	A10, B9,C3, F12
Data Bus	M8, N8, N13	L7, L11, N7, K3
Logic	D1, D2, E3, G11, G13	G12, H13, J3, K1
Clock	—	B1

Table 6. Dynamic Electrical Characteristics (Continued)

Symbol	Parameter	Interval Number	68020-16		68020-20		68020-25		Unit	Notes
			Min	Max	Min	Max	Min	Max		
t_{DVSA}	Data Out Valid to $\overline{DS}$ Asserted (Write) 26	26	15		10		5		ns	(6)
t_{DICL}	Data in Valid to Clock Low (Data Setup)	27	5		5		5		ns	
t_{BELCL}	Late $\overline{BERR}/\overline{HALT}$ Asserted to Clock Low Setup Time	27A	20		15		10		ns	
t_{SNDN}	$\overline{AS}$, $\overline{DS}$ Negated to $\overline{DSACKx}/\overline{BERR}/\overline{HALT}/\overline{AVEC}$ Negated	28	0	80	0	65	0	50	ns	
t_{SNDI}	$\overline{DS}$ Negated to Data On Invalid (Data in Hold Time)	29	0		0		0		ns	(6)
t_{SNDIZ}	$\overline{DS}$ Negated to Data in High Impedance	29A		60		50		40	ns	
t_{DADI}	$\overline{DSACKx}$ Asserted to Data In Valid	31		50		43		32		(2)(11)
t_{DADV}	$\overline{DSACK}$ Asserted to $\overline{DSACKx}$ Valid ($\overline{DSACK}$ Asserted Skew)	31A		15		10		10	ns	(3)(11)
t_{HRrf}	$\overline{RESET}$ Input Transition Time	32		1.5		1.5		1.5	Clks	
t_{CLBA}	Clock Low to $\overline{BG}$ Asserted	33	0	30	0	25	0	20	ns	
t_{CLBN}	Clock Low to $\overline{BG}$ Negated	34	0	30	0	25	0	20	ns	
t_{BRAGA}	$\overline{BR}$ Asserted to $\overline{BG}$ Asserted (RMC Not Asserted)	35	1.5	3.5	1.5	3.5	1.5	3.5	Clks	(11)
t_{GAGN}	$\overline{BGACK}$ Asserted to $\overline{BG}$ Negated	37	1.5	3.5	1.5	3.5	1.5	3.5	Clks	(11)
t_{GABRN}	$\overline{BGACK}$ Asserted to $\overline{BR}$ Negated	37A	0	1.5	0	1.5	0	1.5	Clks	(11)
t_{GN}	$\overline{BG}$ Width Negated	39	90		75		60		ns	(11)
t_{GA}	$\overline{BG}$ Width Asserted	39A	90		75		60		ns	
t_{CHDAR}	Clock High to $\overline{DBEN}$ Asserted (Read)	40	0	30	0	25	0	20	ns	
t_{CLDNR}	Clock Low to $\overline{DBEN}$ Negated (Read)	41	0	30	0	25	0	20	ns	
t_{CLDAW}	Clock Low to $\overline{DBEN}$ Negated (Read)	42	0	30	0	25	0	20	ns	
t_{CHDNW}	Clock High to $\overline{DBEN}$ Asserted (Read)	43	0	30	0	25	0	20	ns	
t_{RADA}	R/W Low to $\overline{DBEN}$ Asserted (Write)	44	15		10		10		ns	(6)
t_{DA}	$\overline{DBEN}$ Width Asserted	45								
	READ		60		50		40		ns	(5)
	WRITE		120		100		80		ns	(5)
t_{RWA}	R/$\overline{W}$ Width Asserted (Write or Read)	46	150		125		100		ns	
t_{AIST}	Asynchronous Input Setup Time	47A	5		5		5		ns	(11)
t_{AIHT}	Asynchronous Input Hold Time	47B	15		15		10		ns	(11)
t_{DABA}	$\overline{DSACKx}$ Asserted to $\overline{BERR}/\overline{HALT}$ Asserted	48		30		20		18	ns	(4)(11)
t_{DOCH}	Data Out Hold from Clock High	53	0		0		0		ns	
t_{BNHN}	$\overline{BERR}$ Negated to $\overline{HALT}$ Negated (Rerun)		0		0		0		ns	

Table 6. Dynamic Electrical Characteristics (Continued)

Symbol	Parameter	Interval Number	68020-16		68020-20		68020-25		Unit	Notes
			Min	Max	Min	Max	Min	Max		
f	Frequency of Operation		8.0	16.67	12.5	20.0	12.5	25	MHz	
t_{RADC}	R/W Asserted to Data Bus Impedance Change	55	30		25		20			(11)
t_{HRPW}	$\overline{RESET}$ Pulse Width (Reset Instruction)	56	512		512		512		Clks	(11)
t_{BNHN}	$\overline{BERR}$ Negated to $\overline{HALT}$ Negated (Rerun)	57	0		0		0		ns	(11)
t_{GANBD}	$\overline{BGACK}$ Negated to Bus Driven	58	1		1		1		Clks	(10)(11)
t_{GNBD}	$\overline{BG}$ Negated to Bus Driven	59	1		1		1		Clks	(10)(11)

Notes:
1. This number can be reduced to 5 nanoseconds if the strobes have equal loads.
2. If the asynchronous setup time (= 47) requirements are satisfied, the DSACKx low to data setup time (= 31) and $\overline{DSACKx}$ low to $\overline{BERR}$ low setup time (= 48) can be ignored. The data must only satisfy the data in to clock low setup time (= 27) for the following clock cycle, $\overline{BERR}$ must only satisfy the late $\overline{BERR}$ low to clock setup time (= 27) for the following clock cycle.
3. This parameter specifies the maximum allowable skew between $\overline{DSACK0}$ to $\overline{DSACK1}$ asserted or $\overline{DSACK1}$ to $\overline{DSACK0}$ asserted pattern = 47 must be met by $\overline{DSACK0}$ and $\overline{DSACK1}$.
4. In the absence of $\overline{DSACKx}$, $\overline{BERR}$ is an asynchronous input using the asynchronous input setup time (= 47).
5. $\overline{DBEN}$ may stay asserted on consecutive write cycles.
6. Actual value depends on the clock input waveform.
7. This pattern indicates the minimum high time for $\overline{ECS}$ and $\overline{OCS}$ in the event of an internal cache hit followed immediately by a cache miss or operand cycle.
8. This specification guarantees operations with the 68881 co-processor, and defines a minimum time for DS negated to AS asserted (= 13A). Without this parameter, incorrect interpretation of = 9A and = 15 would indicate that the 68020 does not meet 68881 requirements.
9. This pattern allows the systems designer to guarantee data hold times on the output side of data buffers that have output enable signals generated with $\overline{DBEN}$.
10. Guarantees that an alternate bus master has stopped driving the bus when the 68020 regains control of the bus after an arbitration sequence.
11. Cannot be tested. Provided for system design purposes only.
12. T_{case} = -55°C and +130°C in a Power off condition under Thermal soak for 4 minutes or until thermal equilibrium. Electrical parameters are tested “instant on” 100 m sec. after power is applied.
13. All outputs unload except for load capacitance. Clock = fmax,
    LOW: $\overline{HALT}$, $\overline{RESET}$
    HIGH: $\overline{DSACK0}$, $\overline{DSACK1}$, $\overline{CDIS}$, $\overline{IPL0}$-$\overline{IPL2}$, $\overline{DBEN}$, $\overline{AVEC}$, $\overline{BERR}$.

Test Conditions Specific to the Device

Loading Network

The applicable loading network shall be defined in column “Test conditions” of Table 6, referring to the loading network number as shown in Figure 6, Figure 7, Figure 8 below.

Figure 6. RESET Test Loads

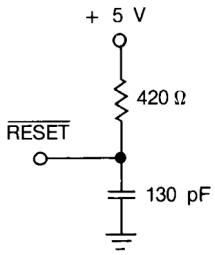


Figure 7. HALT Test Load

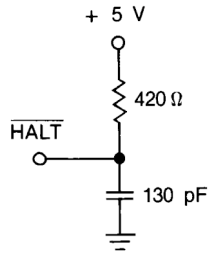


Figure 8. Test Load

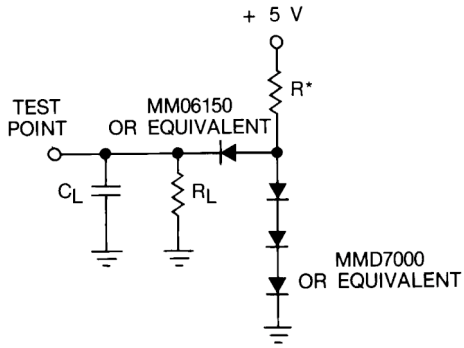


Table 7. Load Network

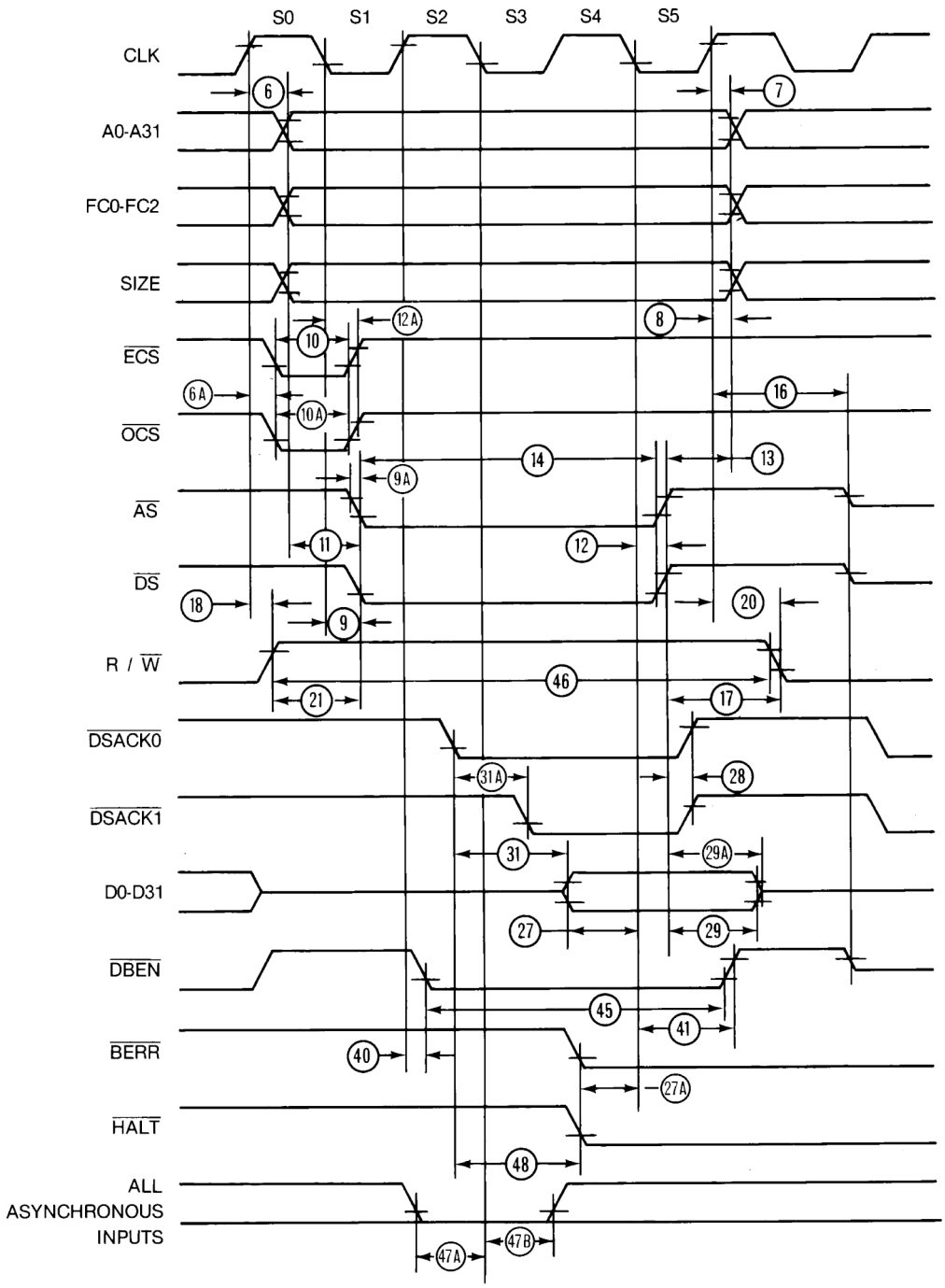
Load NBR	Figure	R	R_L	C_L	Output Application
1	7	2 k	6.0 k	50 pF	$\overline{OCS}$, $\overline{ECS}$
2	7	1.22 k	6.0 k	130 pF	A0-A31, D0-D31, $\overline{BG}$, FC0-FC2, SIZ0-SIZ1
3	7	0.74 k	6.0 k	130 pF	$\overline{AS}$, $\overline{DS}$, R/$\overline{W}$, $\overline{RMC}$, $\overline{DBEN}$, $\overline{IPEND}$

Note: 1. Equivalent loading may be simulated by the tester.

Time Definitions

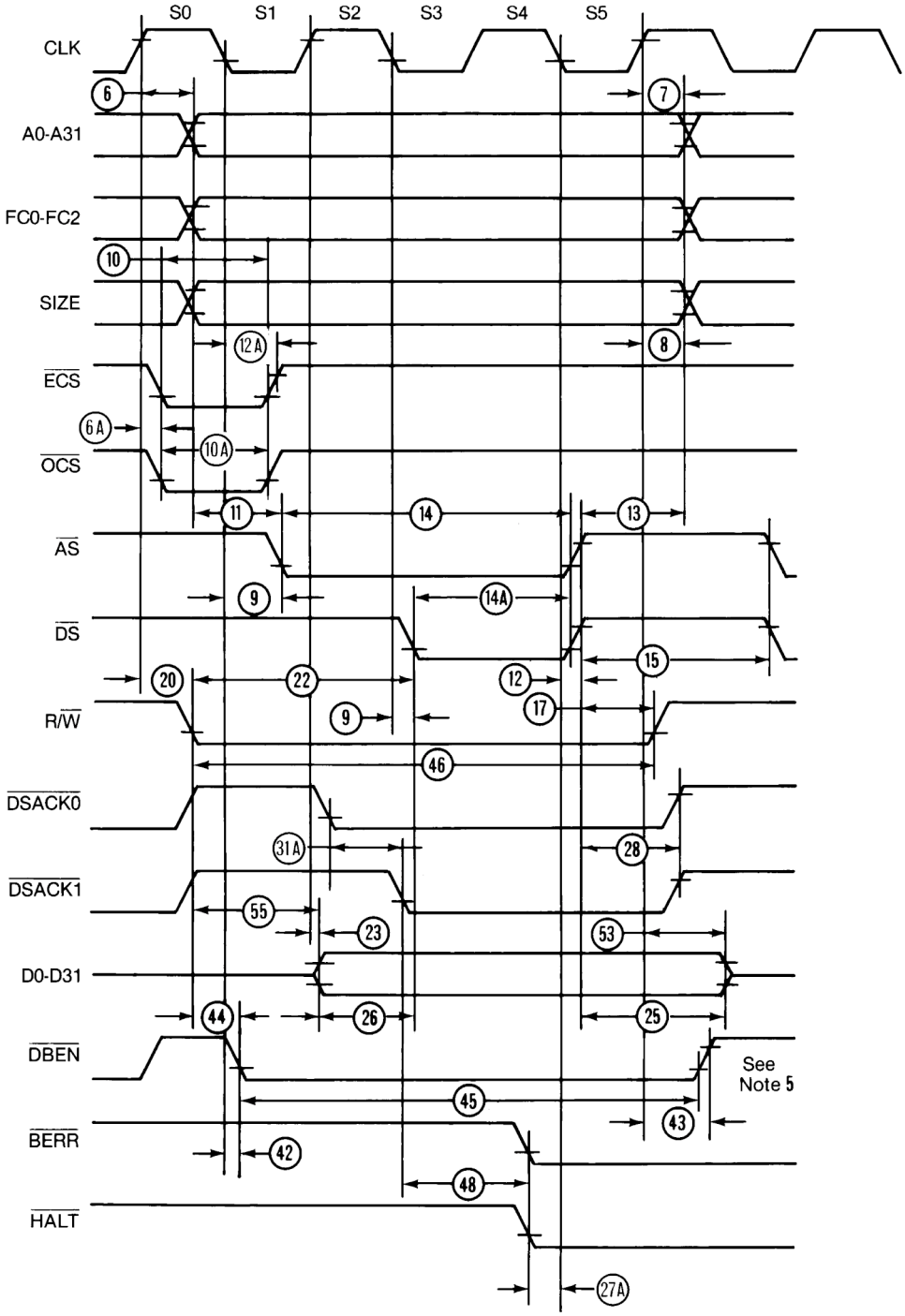
The times specified in Table 6 as dynamic characteristics are defined in Figure 9 below, by a reference number given the column “interval N°” of the tables together with the relevant figure number.

Figure 9. Read Cycle Timing Diagram



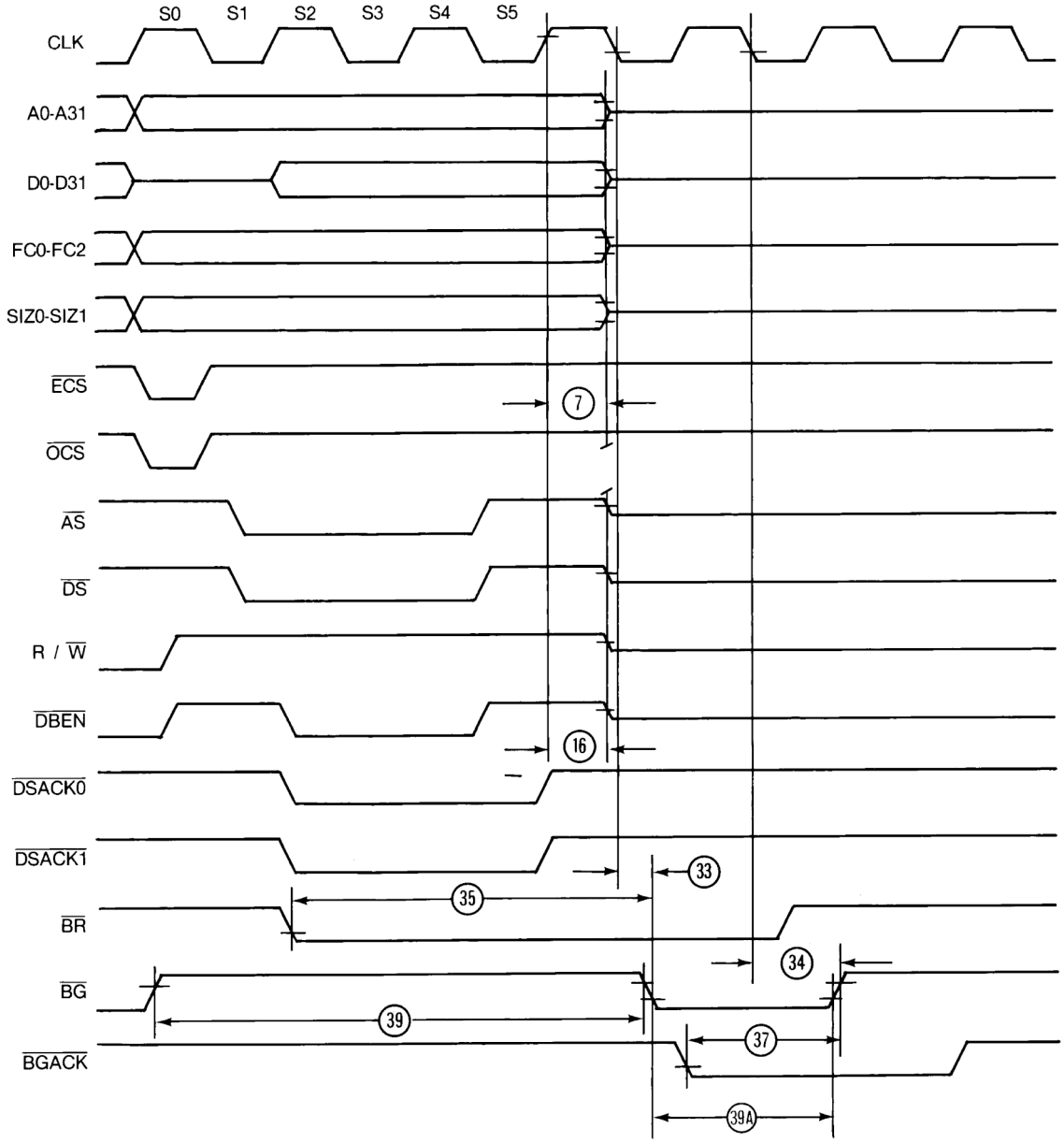
Note: Timing measurements are referenced to and from a low voltage of 0.8V and a high voltage of 2.0V, unless otherwise noted. The voltage swing through this range should start outside and pass through the range such that the rise or fall will be linear between 0.8V and 2.0V.

Figure 10. Write Cycle Timing Diagram (Continued)



Note: Timing measurements are referenced to and from a low voltage of 0.8V and a high voltage of 2.0V, unless otherwise noted. The voltage swing thorough this range should start outside and pass through the range such that the rise or fall will be linear between 0.8V and 2.0V.

Figure 11. Bus Arbitration Timing Diagram



Note: Timing measurements are referenced to and from a low voltage of 0.8V and a high voltage of 2.0V, unless otherwise noted. The voltage swing thorough this range should start outside and pass through the range such that the rise or fall will be linear between 0.8V and 2.0V.

Input and Output Signals for Dynamic Measurements

AC Electrical Specifications Definitions

The AC specifications presented consist of output delays, input setup and hold times, and signal skew times. All signals are specified relative to an appropriate edge of the TS68020 clock input and, possibly, relative to one or more other signals.

The measurement of the AC specifications is defined by the waveforms in Figure 12. In order to test the parameters guaranteed by Atmel, inputs must be driven to the voltage levels specified in Figure 12. Outputs of the TS68020 are specified with minimum and/or maximum limits, as appropriate, and are measured as shown. Inputs to the TS68020 are specified with minimum and, as appropriate, maximum setup and hold times, and are measurement as shown. Finally, the measurements for signal-to-signal specification are also shown.

Note that the testing levels used to verify conformance of the TS68020 to the AC specifications does not affect the guaranteed DC operation of the device as specified in the DC electrical characteristics.

Figure 16. DS, AS, IPEND, and BG Capacitance Derating Curve

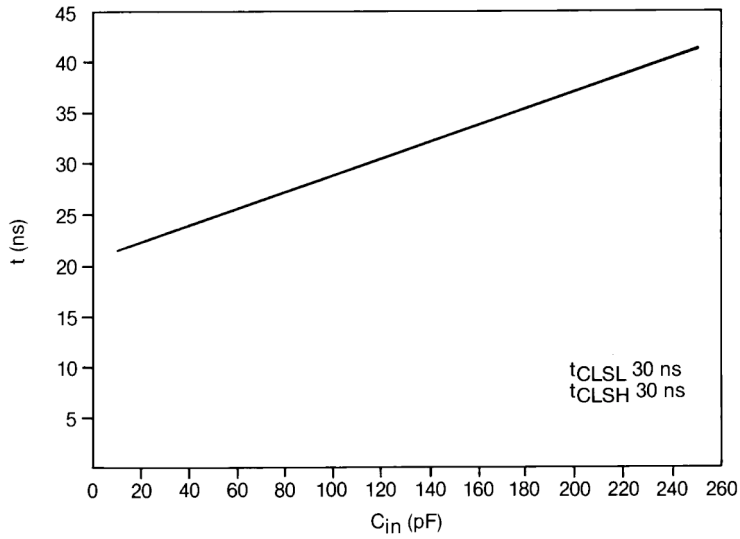


Figure 17. DBEN Capacitance Derating Curve

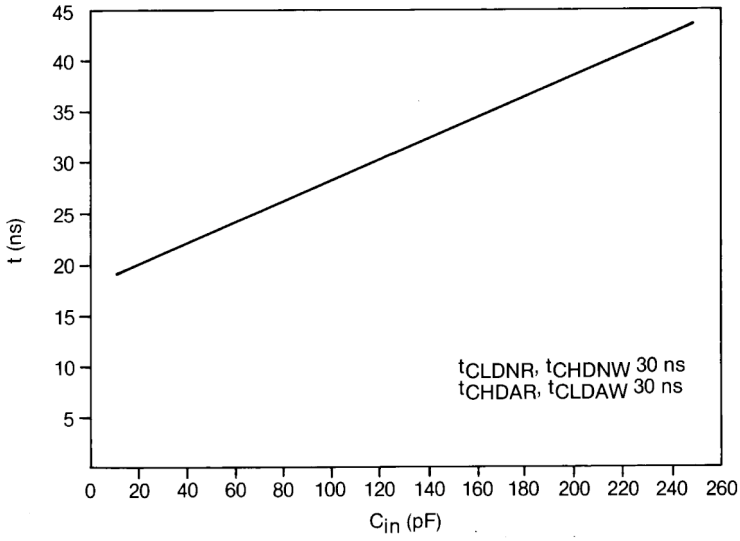
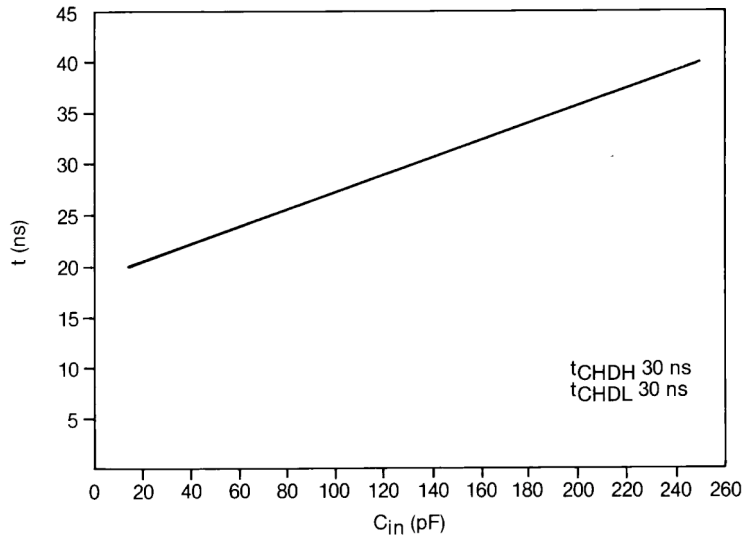


Figure 18. Data Capacitance Derating Curve



Functional Description

Description of Registers

As shown in the programming models (Figure 19 and Figure 20) the TS68020 has sixteen 32-bit general-purpose registers, a 32-bit program counter, two 32-bit supervisor stack pointers, a 16-bit status register, a 32-bit vector base register, two 3-bit alternate function code registers, and two 32-bit cache handling (address and control) registers. Registers D0-D7 are used as data registers for bit and bit field (1- to 32-bit), byte (8-bit), long word (32-bit), and quad word (64-bit) operations. Registers A0-A6 and the user, interrupt, and master stack pointers are address registers that may be used as software stack pointers or base address registers. In addition, the address registers may be used for word and long word operations. All of the 16 (D0-D7, A0-A7) registers may be used as index registers.

The status register (Figure 21) contains the interrupt priority mask (three bits) as well as the condition codes: extend (X), negated (N), zero (Z), overflow (V), and carry (C). Additional control bits indicate that the processor is in the trace mode (T1 or T0), supervisor/user state (S), and master/interrupt state (M).

All microprocessors of the TS68000 Family support instruction tracing (via the T0 status bit in the TS68020) where each instruction executed is followed by a trap to a user-defined trace routine. The TS68020 adds the capability to trace only the change of flow instructions (branch, jump, subroutine call and return, etc.) using the T1 status bit. These features are important for software program development and debug.

The vector base register is used to determine the runtime location of the exception vector table in memory, hence it supports multiple vector tables so each process or task can properly manage exceptions independent of each other.

Table 9. Instruction Set (Continued)

Mnemonic	Description
CALLM	Call Module
CAS	Compare and Swap Operands
CAS2	Compare and Swap Dual Operands
CHK	Check Register Against Bound
CHK2	Check Register Against Upper and Lower Bounds
CLR	Clear
CMP	Compare
CMPA	Compare Address
CMPI	Compare Immediate
CMPM	Compare Memory to Memory
CMP2	Compare Register Against Upper and Lower Bounds
DBcc	Test Condition, Decrement and Branch
DIVS, DIVSL	Signed Divide
DIVU, DIVUL	Unsigned Divide
EOR	Logical Exclusive OR
EORI	Logical Exclusive OR Immediate
EXG	Exchange Registers
EXT, EXTB	Sign Extend
ILLEGAL	Take Illegal Instruction Tape
JMP	Jump
JSR	Jump to Subroutine
LEA	Load Effective Address
LINK	Link and Allocate
LSL, LSR	Logical Shift Left and Right
MOVE	Move
MOVEA	Move Address
MOVE CCR	Move Condition Code Register
MOVE SR	Move Status Register
MOVE USP	Move User Stack Pointer
MOVEC	Move Control Register
MOVEM	Move Multiple Registers
MOVEP	Move Peripheral
MOVEQ	Move Quick
MOVES	Move Alternate Address Space
MULS	Signed Multiply
MULU	Unsigned Multiply
NBCD	Negate Decimal with Extend
NEG	Negate
NEGX	Negate with Extend
NOP	No Operation
NOT	Logical Complement

Table 9. Instruction Set (Continued)

Mnemonic	Description
OR	Logical Inclusive OR
ORI	Logical Inclusive OR Immediate
PACK	Pack BCD
PEA	Push Effective Address
RESET	Reset External Devices
ROL, ROR	Rotate Left and Right
ROXL, ROXR	Rotate with Extend Left and Right
RTD	Return and Deallocate
RTE	Return and Exception
RTM	Return from Module
RTR	Return and Restore Codes
RTS	Return from Subroutine
SBCD	Subtract Decimal with Extend
Scc	Set Conditionally
STOP	Stop
SUB	Subtract
SUBA	Subtract Address
SUBI	Subtract Immediate
SUBQ	Subtract Quick
SUBX	Subtract with Extend
SWAP	Swap Register Words
TAS	Test Operand and Set
TRAP	Trap
TRAPcc	Trap Conditionally
TRAPV	Trap on Overflow
TST	Test Operand
UNLK	Unlink
UNPK	Unpack BCD
Co-processor Instructions	
cpBCC	Branch Conditionally
cpDBcc	Test Co-processor Condition, Decrement and Branch
cpGEN	Co-processor General Instruction
cpRESTORE	Restore Internal State of Co-processor
cpSAVE	Save Internal State of Co-processor
cpScc	Set Conditionally
cpTRAPcc	Trap Conditionally

Bit Field Operation

The TS68020 supports variable length bit field operations up to 32-bit. A bit field may start in any bit position and span any address boundary for the full length of the bit field, up to the 32-bit maximum. The bit field insert (BFINS) inserts a value into a field. Bit field extract unsigned (BFEXTU) and bit field extract signed (BFEXTS) extract an unsigned or signed value from the field. BFFFO finds the first bit in a bit field that is set. To complement the TS68000 bit manipulation instruction, there are bit field change, clear, set and test instructions (BFCHG, BFCLR, BFSET, BFTST). Using the on-chip barrel shifter, the bit and bit field instructions are very fast and particularly useful in applications using packed bits and bit fields, such as graphics and communications.

Binary Coded Decimal (BCD) Support

The TS68000 Family supports BCD operations including add, subtract, and negation. The TS68020 adds the PACK and UNPACK operations for BCD conversions to and from binary form as well as other conversions, e.g., ASCII and EBCDIC. The PACK instruction reduces two bytes of data into a single byte while UNPACK reverses the operation.

Bounds Checking

Previous 68000 Family members offer variable bounds checking only on the upper limit of the bound. The underlying assumption is that the lower bound is zero. This is expanded on the TS68020 by providing two new instructions, CHK2 and CMP2. These instructions allow checking and comparing of both the upper and lower bounds. These instructions may be either signed or unsigned. The CMP2 instructions sets the condition codes upon completion while the CHK2 instruction, in addition to setting the condition codes, will take a system trap if either boundary condition is exceeded.

System Traps

Three additions have been made to the system trap capabilities of the TS68020. The current TRAPV (trap on overflow) instruction has been expanded to a TRAPcc format where any condition code is allowed to be the trapping condition. And, the TRAPcc instruction is expanded to optionally provide one or two additional words following the trap instruction so user-specified information may be presented to the trap handler. These additional words can be used when needed to provide simple error codes or debug information for interactive runtime debugging or post-mortem program dumps. Compilers may provide direction to run-time execution routines towards handling of specific conditions.

The breakpoint instruction, BKPT, is used to support the program breakpoint function for debug monitors and real-time in-circuit or hardware emulators, and the operation will be dependent on the actual system implementation. Execution of this instruction causes the TS68020 to run a breakpoint acknowledge bus cycle, with a 3-bit breakpoint identifier placed on address lines A2, A3, and A4. This 3-bit identifier permits up to eight breakpoints to be easily differentiated. The normal response to the TS68020 is an operation word (typically an instruction, originally replaced by the debugger with the breakpoint instruction) placed on the data lines by external debugger hardware and the breakpoint acknowledge cycle properly terminated. The TS68020 then executes this operation word in place of the breakpoint instruction. The debugger hardware can count the number of executions of each breakpoint and halt execution after a pre-determined number of cycles.

The TS68020 uses instruction continuation to support virtual memory. In order for the TS68020 to use instruction continuation, it stores its internal state on the supervisor stack when a bus cycle is terminated with a bus error signal. It then loads the program counter with the address of the virtual memory bus error handler from the exception vector table (entry number two) and resumes program execution to that new address. When the bus error exception handler routine has completed execution, an RTE instruction is executed which reloads the TS68020 with the internal state stored on the stack, reruns the faulted bus cycle (when required), and continues the suspended instruction.

Instruction continuation is crucial to the support of virtual I/O devices in memory-mapped input/output systems. Since the registers of a virtual device may be simulated in the memory map, an access to such a register will cause a fault and the function of the register can be emulated by software.

Virtual Machine

A typical use for a virtual machine system is the development of software, such as an operating system, for a new machine also under development and not yet available for programming use. In such a system, a governing operating system emulates the hardware of the prototype system and allows the new operating system to be executed and debugged as though it were running on the new hardware. Since the new operating system is controlled by the governing operating system, it is executed at a lower privilege level than the governing operating system. Thus, any attempts by the new operating system to use virtual resources that are not physically present (and should be emulated) are trapped to the governing system and handled by its software. In the TS68020, a virtual machine is fully supported by running the new operating system in the user mode. The governing operating system executes in the supervisor mode and any attempt by the new operating system to access supervisor resources or execute privileged instructions will cause a trap to the governing operating system.

Operand Transfer Mechanism

Though the TS68020 has a full 32-bit data bus, it offers the ability to automatically and dynamically downsize its bus to 8- or 16-bit if peripheral devices are unable to accommodate the entire 32-bit. This feature allows the programmer the ability to write code that is not bus-width specific. For example, long word (32-bit) accesses to peripherals may be used in the code, yet the TS68020 will transfer only the amount of data that the peripheral can manage. This feature allows the peripheral to define its port size as 8-, 16-, or 32-bit wide and the TS68020 will dynamically size the data transfer accordingly, using multiple bus cycles when necessary. Hence, programmers are not required to program for each device port size or know the specific port size before coding; hardware designers have flexibility to choose implementations independent of software prejudices.

This is accomplished through the use of the $\overline{\text{DSACK}}$ pins and occurs on a cycle-by-cycle basis. For example, if the processor is executing an instruction that requires the reading of a long word operand, it will attempt to read 32-bit during the first bus cycle to a long word address boundary. If the port responds that it is 32-bit wide, the TS68020 latches all 32-bit of data and continues. If the port responds that it is 16-bit wide, the TS68020 latches 16 valid bits of data and runs another cycle to obtain the other 16-bit of data. An 8-bit port is handled similarly by with four bus read cycles. Each port is fixed in assignment to particular sections of the data bus.

Justification of data on the bus is handled automatically by dynamic bus sizing. When reading 16-bit data from a 32-bit port, the data may appear on the top or bottom half of the bus, depending on the address of the data. The TS68020 determines which portion of the bus is needed to support the transfer and dynamically adjusts to read or write the data on those data lines.

The TS68020 will always transfer the maximum amount of data on all bus cycles; i.e., it always assumes the port is 32-bit wide when beginning the bus cycle. In addition, the TS68020 has no restrictions concerning alignment of operands in memory; long word operands need not be aligned on long word address boundaries. When misaligned data requires multiple bus cycles, the TS68020 aligned data requires multiple bus cycles, the TS68020 automatically runs the minimum number of bus cycles.

The Co-processor Concept

The co-processor interface is a mechanism for extending the instruction set of the TS68000 Family. Examples of these extensions are the addition of specialized data operands for the existing data types or, for the case of the floating point, the inclusion of new data types and operations for them as implemented by the TS68881 and TS68882 floating point co-processors.

The programmer's model for the TS68000 Family of microprocessors is based on sequential, non-concurrent instruction execution. This means each instruction is completely executed prior to the beginning of the next instruction. Hence, instructions do not operate concurrently in the programmer's model. Most microprocessors implement the sequential model which greatly simplifies the programmer responsibilities since sequencing control is automatic and discrete.

The TS68000 co-processor interface is designed to extend the programmer's model and it provides full support for the sequential, non-concurrent instruction execution model. Hence, instruction execution by the co-processor is assumed to not overlap with instruction execution with the main microprocessor. Yet, the TS68000 co-processor interface does allow concurrent operation when concurrency can be properly accommodated. For example, the TS68881 or TS68882 floating-point co-processor will allow the TS68020 to proceed executing instruction while the co-processor continues a floating-point operation, up to the point that the TS68020 sends another request to the co-processor. Adhering to the sequential execution model, the request to the co-processor continues a floating-point operation, up to the co-processor completes each TS68881 and TS68882 instruction before it starts the next, and the TS68020 is allowed to proceed as it can in a concurrent fashion.

co-processors are divided into two types by their bus utilization characteristics. A co-processor is a DMA co-processor if it can control the bus independent of the main processor. A co-processor is a non-DMA co-processor if it does not have the capability of controlling the bus. Both co-processor types utilize the same protocol and main processor resources. Implementation of a co-processor as a DMA or non-DMA type is based primarily on bus bandwidth of the co-processor, performance, and cost issues.

The communication protocol between the main processor and the co-processor necessary to execute a co-processor instruction is based on a group of co-processor interface registers (Table 10) which are defined for the TS68000 Family co-processor interface. The TS68020 hardware uses standard TS68000 asynchronous bus cycles to access the registers. Thus, the co-processor doesn't require a special bus hardware; the bus interface implemented by a co-processor for its interface register set must only satisfy the TS68020 address, data, and control signal timing to guarantee proper communication with the main processor. The TS68020 implements the communication protocol with all co-processors in hardware (and microcode) and handles all operations automatically so the programmer is only concerned with the instructions and data types provided by the co-processor as extensions to the TS68020 instruction set and data types.

When the main processor encounters the next co-processor instruction, the main processor queries the co-processor until the co-processor is ready; meanwhile, the main processor can go on to service interrupts and do a context switch to execute other tasks, for example.

Each co-processor instruction type has specific requirements based on this simplified protocol. The co-processor interface may use as many extension words as requires to implement a co-processor instruction.

Primitives/Response

The response register is the means by which the co-processor communicates service requests to the main processor. The content of the co-processor response register is a primitive instruction to the main processor which is read during co-processor communication by the main processor. The main processor “executes” this primitive, thereby providing the services requires by the co-processor. Table 11 summarizes the co-processor primitives that the TS68020 accepts.

Exceptions

Kinds of Exceptions

Exception can be generated by either internal or external causes. The externally generated exceptions are the interrupts, the bus error, and reset requests. The interrupts are requests from peripheral devices for processor action while the bus error and reset pins are used for access control and processor restart. The internally generated exceptions come from instructions, address errors, tracing, or breakpoints. The TRAP, TRAPcc, TRAPV, cpTRAPcc, CHK, CHK2, and DIV instructions can all generate exceptions as part of their execution. Tracing behaves like a very high priority, internally generated interrupt whenever it is processed. The other internally generated exceptions are caused by illegal instructions, instruction fetches from odd addresses, and privilege violations.

Exception Processing Sequence

Exception processing occurs in four steps. During the first step, an internal copy is made of the status register. After the copy is made, the special processor state bits in the status register are changed. The S bit is set, putting the processor into supervisor privilege state. Also, the T1 and T0 bits are negated, allowing the exception handler to execute unhindered by tracing. For the reset and interrupt exceptions, the interrupt priority mask is also updated.

In the second step, the vector number of the exception is determined. For interrupts, the vector number is obtained by a processor read that is classified as an interrupt acknowledge cycle. For co-processor detected exceptions, the victor number is included in the co-processor exception primitive response. For all other exceptions, internal logic provides the vector number. This vector number is then used to generate the address of the exception vector.

The third step is to save the current processor status. The exception stack frame is created and filled on the supervisor stack. In order to minimize the amount of machine state that is saved, various stack frame sizes are used to contain the processor state depending on the type of exception and where it occurred during instruction execution. If the exception is an interrupt and the M bit is on, the M bit is forced off, and a short four word exception stack frame is saved on the master stack which indicates that the exception is saved on the interrupt stack. If the exception is a reset, the M bit is simply forced off, and the reset vector is accessed.

Preparation for Delivery

Certificate of Compliance

Atmel offers a certificate of compliance with each shipment of parts, affirming the products are in compliance with MIL-STD-883 and guaranteeing the parameters are tested at extreme temperatures for the entire temperature range.

Handling

MOS devices must be handled with certain precautions to avoid damage due to accumulation of static charge. Input protection devices have been designed in the chip to minimize the effect of this static buildup. However, the following handling practices are recommended:

a) Device should be handled on benches with conductive and grounded surface.

b) Ground test equipment, tools and operator

c) Do not handle devices by the leads.

d) Store devices in conductive foam or carriers.

e) Avoid use of plastic, rubber, or silk in MOS areas.

f) Maintain relative humidity above 50%, if practical.

Package Mechanical Data

Figure 23. 114-lead - Ceramic Pin Grid Array

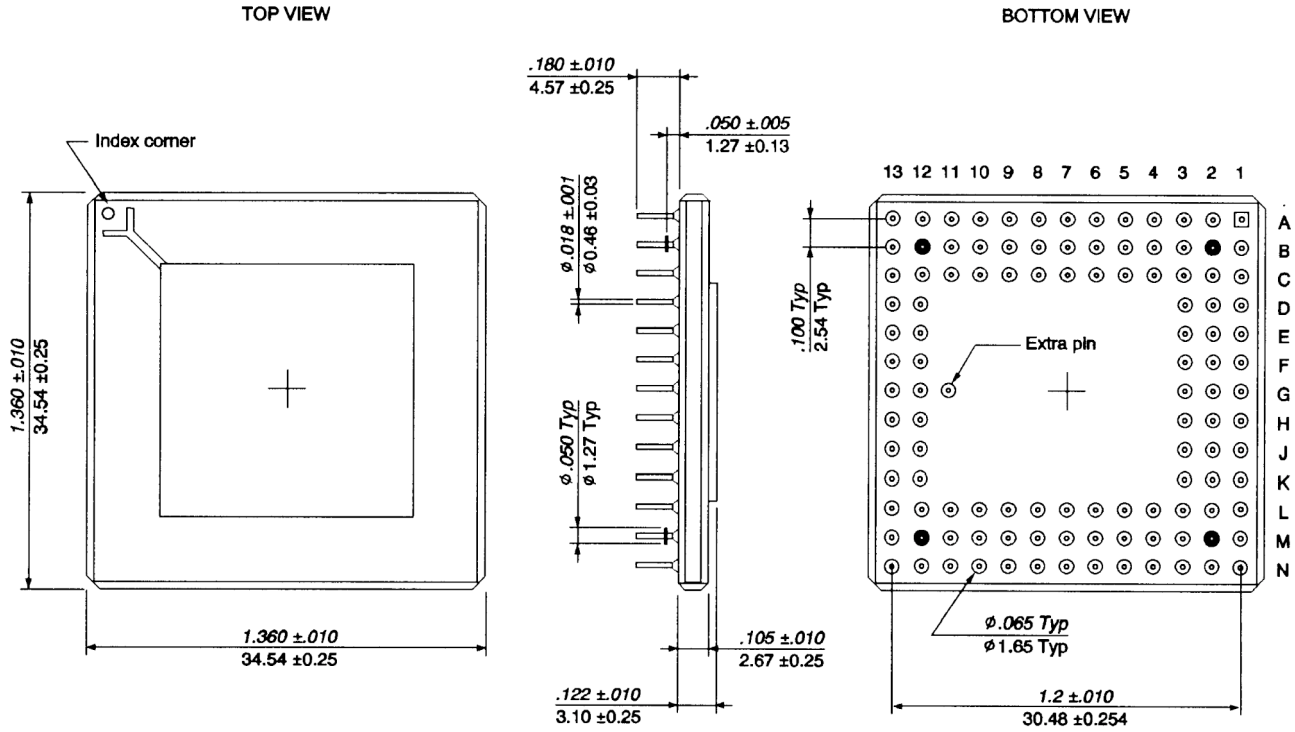
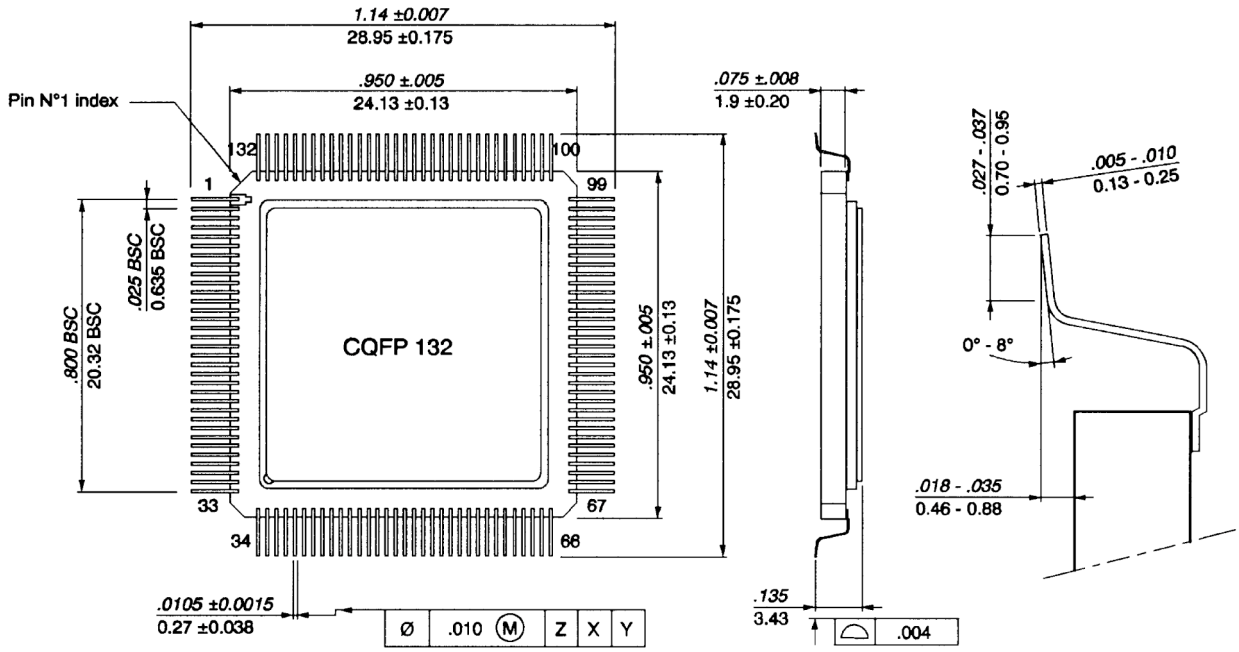


Figure 24. 132 Pins - Ceramic Quad Flat Pack



Atmel Headquarters

Corporate Headquarters
2325 Orchard Parkway
San Jose, CA 95131
TEL 1(408) 441-0311
FAX 1(408) 487-2600

Europe
Atmel Sarl
Route des Arsenaux 41
Case Postale 80
CH-1705 Fribourg
Switzerland
TEL (41) 26-426-5555
FAX (41) 26-426-5500

Asia
Room 1219
Chinachem Golden Plaza
77 Mody Road Tsimhatsui
East Kowloon
Hong Kong
TEL (852) 2721-9778
FAX (852) 2722-1369

Japan
9F, Tonetsu Shinkawa Bldg.
1-24-8 Shinkawa
Chuo-ku, Tokyo 104-0033
Japan
TEL (81) 3-3523-3551
FAX (81) 3-3523-7581

Atmel Operations

Memory
2325 Orchard Parkway
San Jose, CA 95131
TEL 1(408) 441-0311
FAX 1(408) 436-4314

Microcontrollers
2325 Orchard Parkway
San Jose, CA 95131
TEL 1(408) 441-0311
FAX 1(408) 436-4314

La Chantrerie
BP 70602
44306 Nantes Cedex 3, France
TEL (33) 2-40-18-18-18
FAX (33) 2-40-18-19-60

ASIC/ASSP/Smart Cards
Zone Industrielle
13106 Rousset Cedex, France
TEL (33) 4-42-53-60-00
FAX (33) 4-42-53-60-01

1150 East Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906
TEL 1(719) 576-3300
FAX 1(719) 540-1759

Scottish Enterprise Technology Park
Maxwell Building
East Kilbride G75 0QR, Scotland
TEL (44) 1355-803-000
FAX (44) 1355-242-743

RF/Automotive
Theresienstrasse 2
Postfach 3535
74025 Heilbronn, Germany
TEL (49) 71-31-67-0
FAX (49) 71-31-67-2340

1150 East Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906
TEL 1(719) 576-3300
FAX 1(719) 540-1759

Biometrics/Imaging/Hi-Rel MPU/
High Speed Converters/RF Datacom
Avenue de Rochepleine
BP 123
38521 Saint-Egreve Cedex, France
TEL (33) 4-76-58-30-00
FAX (33) 4-76-58-34-80

e-mail
literature@atmel.com

Web Site
http://www.atmel.com

2115A–HIREL–07/02 0M