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	-55°C ~ 125°C (TC)
Security Features	-
Package / Case	132-BCQFP
Supplier Device Package	132-CQFP (24.13x24.13)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/ts68020mf25

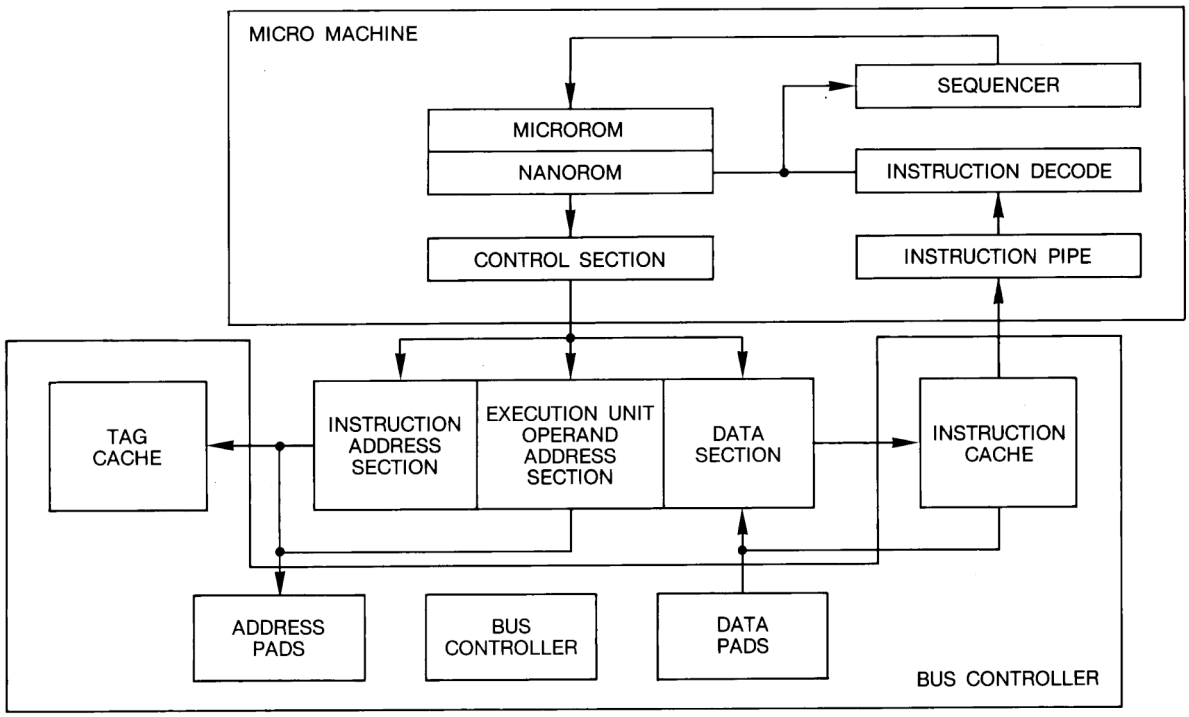
Introduction

The TS68020 is a high-performance 32-bit microprocessor. It is the first microprocessor to have evolved from a 16-bit machine to a full 32-bit machine that provides 32-bit address and data buses as well as 32-bit internal structures. Many techniques were utilized to improve performance and at the same time maintain compatibility with other processors of the TS68000 Family. Among the improvements are new addressing modes which better support high-level language structures, an expanded instruction set which provides 32-bit operations for the limited cases not supported by the TS68000 and several new instructions which support new data types. For special-purpose applications when a general-purpose processor alone is not adequate, a co-processor interface is provided.

The TS68020 is a high-performance microprocessor implemented in HCMOS, low power, small geometry process. This process allows CMOS and HMOS (high density NMOS) gates to be combined on the same device. CMOS structures are used where speed and low power is required, and HMOS structures are used where minimum silicon area is desired. This technology enables the TS68020 to be very fast while consuming less power (less than 1.5 watts) and still have a reasonably small die size. It utilizes about 190.000 transistors, 103.000 of which are actually implemented. The package is a pin-grid array (PGA) with 114 pins, arranged 13 pins on a side with a depopulated center and 132 pins ceramic quad flat pack.

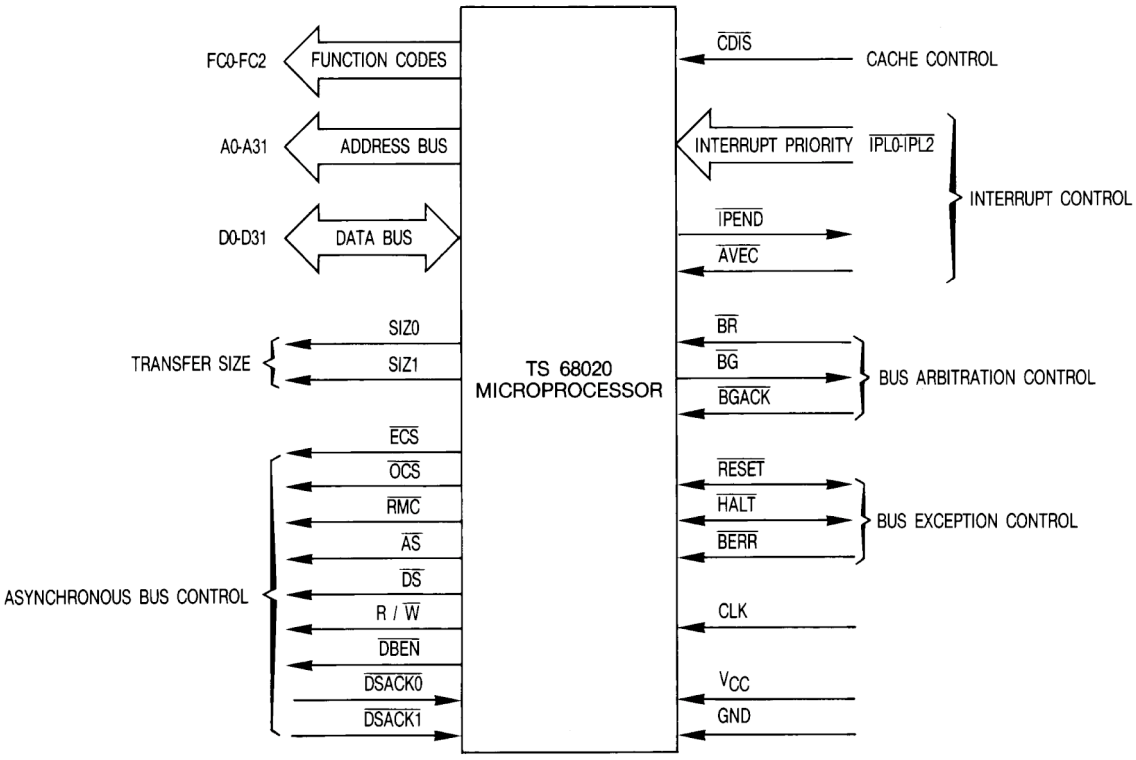
Figure 1 is a block diagram of the TS68020. The processor can be divided into two main sections: the bus controller and the micromachine. This division reflects the autonomy with which the sections operate.

Figure 1. TS68020 Block Diagram



The bus controller consists of the address and data pads and multiplexers required to support dynamic bus sizing, a macro bus controller which schedules the bus cycles on the basis of priority with two state machines (one to control the bus cycles for operated accesses and the other to control the bus cycles for instruction accesses), and the instruction cache with its associated control.

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

Detailed Specifications

Scope

This drawing describes the specific requirements for the microprocessor 68020, 16.67 MHz, 20 MHz and 25 MHz, in compliance with the MIL-STD-883 class B.

Applicable Documents

MIL-STD-883

- MIL-STD-883: Test Methods and Procedures for Electronics
- MIL-PRF-38535 appendix A: General Specifications for Microcircuits
- Desc Drawing 5962 - 860320xxx

Requirements

General

The microcircuits are in accordance with the applicable document and as specified herein.

Design and Construction

Terminal Connections

Depending on the package, the terminal connections shall be as shown in Figure 2 and Figure 3.

Lead Material and Finish

Lead material and finish shall be any option of MIL-STD-1835.

Package

The macrocircuits are packages in hermetically sealed ceramic packages which are conform to case outlines of MIL-STD-1835 (when defined):

- 114-pin SQ.PGA UP PAE Outline
- 132-pin Ceramic Quad Flat Pack CQFP

The precise case outlines are described on Figure 23 and Figure 24.

For dynamic characteristics (Table 6), test methods refer to IEC 748-2 method, where existing.

Indication of “min.” or “max.” in the column “test temperature” means minimum or maximum operating temperature.

Table 5. Static Characteristics. V_{CC} = 5.0V_{DC} ± 10%; GND = 0V_{DC}; T_c = -55/+125°C or -40/+85°C (Figure 4 to Figure 8)

Symbol	Parameter	Condition	Min	Max	Units
I_{CC}	Maximum Supply Current	V_{CC} = 5.5V T_{case} -55°C to +25°C		333	mA
I_{CC}	Maximum Supply Current	V_{CC} = 5.5V T_{case} = 125°C		207	mA
V_{IH}	High Level Input Voltage	V_O = 0.5V or 2.5 V_{CC} = 4.5V to 5.5V	2.0	V_{CC}	V
V_{IL}	Low Level Input Voltage	V_O = 0.5V or 2.4V V_{CC} = 4.5V to 5.5V	-0.5	0.8	V
V_{OH}	High Level Output Voltage All Outputs	I_{OH} = 400 µA	2.4		V
V_{OL}	Low Level Output Voltage Outputs A0-A31, FC0-FC2, D0-D31, SIZ0-SIZ1, $\overline{BG}$	I_{OL} = 3.2 mA Load Circuit as Figure 8 R = 1.22 kΩ		0.5	V
V_{OL}	Low Level Output Voltage Outputs $\overline{AS}$, $\overline{DS}$, $\overline{RMC}$, R/$\overline{W}$, $\overline{DBEN}$, $\overline{IPEND}$	I_{OL} = 5.3 mA Load Circuit as Figure 8 R = 740Ω		0.5	V
V_{OL}	Low Level Output Voltage Outputs $\overline{ECS}$, $\overline{OCS}$	I_{OL} = 2.0 mA Load Circuit as Figure 8 R = 2 kΩ		0.5	V
V_{OL}	Low Level Output Voltage Outputs $\overline{HALT}$, $\overline{RESET}$	I_{OL} = 10.7 mA Load Circuit as Figure 6 and Figure 7		0.5	V
$\lvert I_{IN} \rvert$	Input Leakage Current (High and Low State)	-0.5V ≤ V_{IN} ≤ V_{CC} (Max)		2.5	µA
$\lvert I_{OHZ} \rvert$	High level leakage current at three-state outputs Outputs A0-A31, $\overline{AS}$, $\overline{DBEN}$, $\overline{DS}$, D0-D31, R/$\overline{W}$, FC0-FC2, $\overline{RMC}$, SIZ0-SIZ1	V_{OH} = 2.4V		2.5	µA
$\lvert I_{OLZ} \rvert$	Low Level Leakage Current at Three-state Outputs Outputs A0-A31, $\overline{AS}$, $\overline{DBEN}$, $\overline{DS}$, D0-D31 R/$\overline{W}$, FC0-FC2, $\overline{RMC}$, SIZ0-SIZ1	V_{OL} = 0.5V		2.5	µA
I_{OS}	Output Short-circuit Current (Any Output)	V_{CC} = 5.5V V_O = 0V (Pulsed. Duration 1 ms Duty Cycle 10:1)		200	mA

Table 6. Dynamic Electrical Characteristics (Continued)

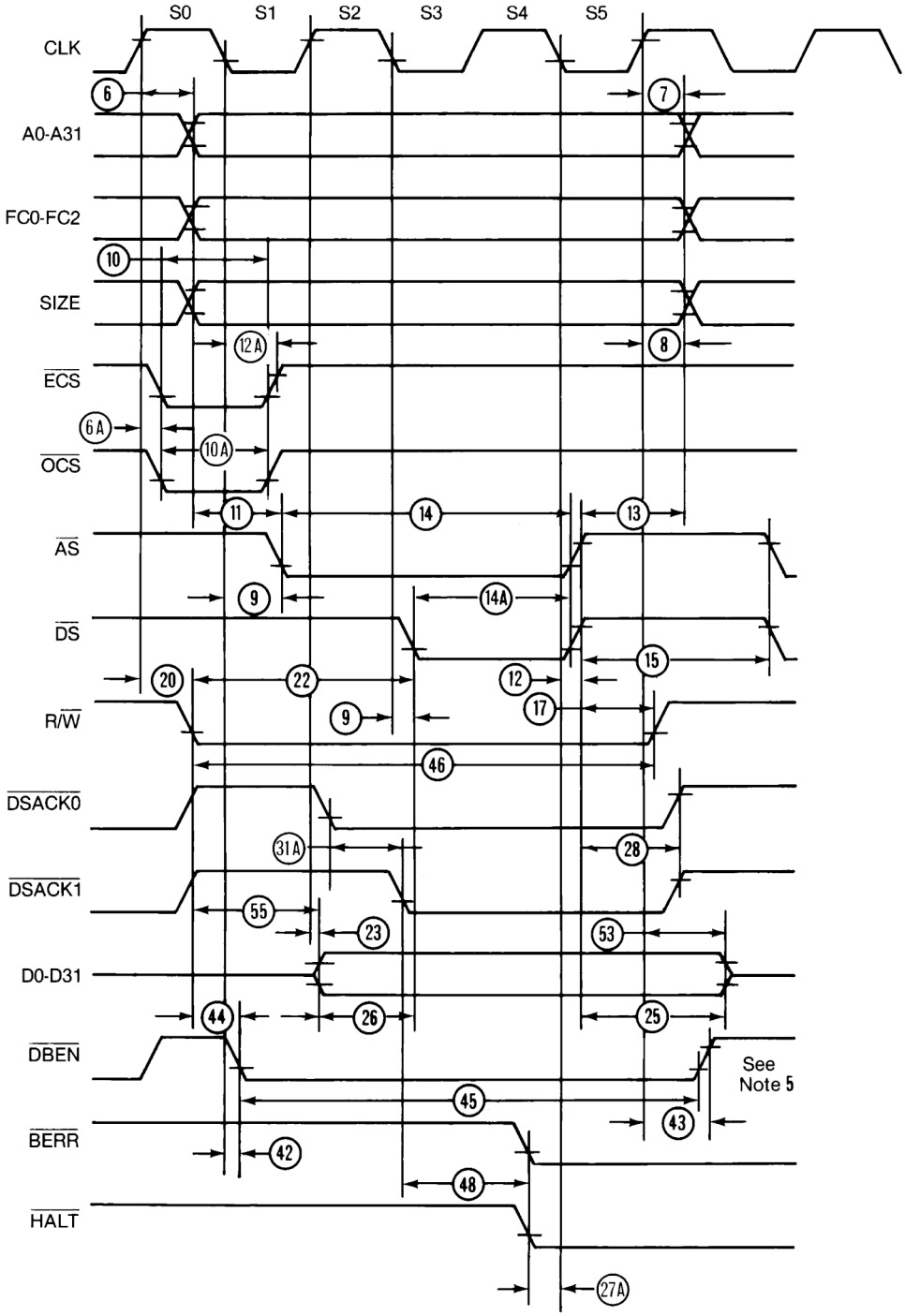
Symbol	Parameter	Interval Number	68020-16		68020-20		68020-25		Unit	Notes
			Min	Max	Min	Max	Min	Max		
t_{DVSA}	Data Out Valid to $\overline{DS}$ Asserted (Write) 26	26	15		10		5		ns	(6)
t_{DICL}	Data in Valid to Clock Low (Data Setup)	27	5		5		5		ns	
t_{BELCL}	Late $\overline{BERR}/\overline{HALT}$ 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)

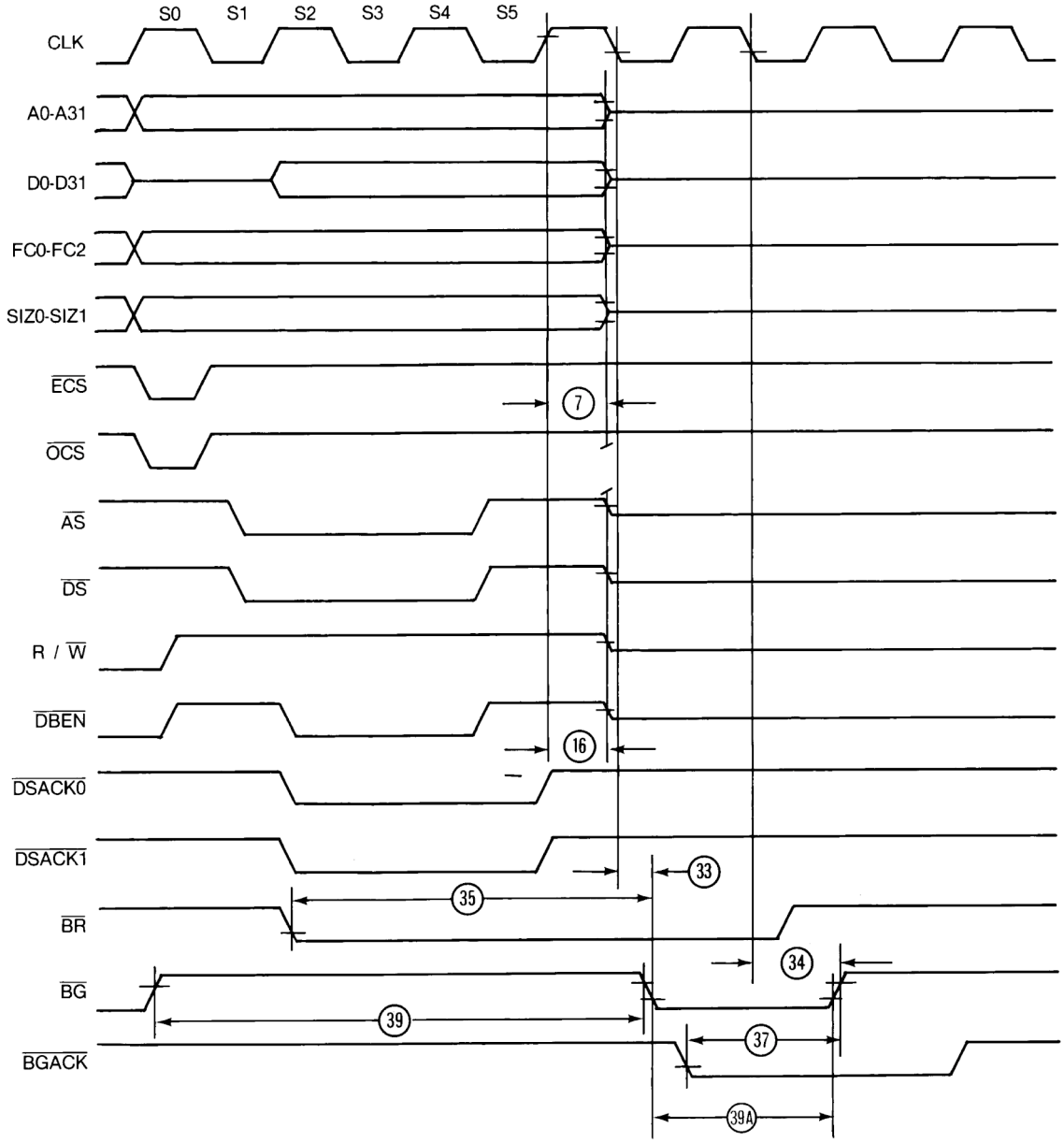
Notes:
1. This number can be reduced to 5 nanoseconds if the strobes have equal loads.
2. If the asynchronous setup time (= 47) requirements are satisfied, the DSACKx low to data setup time (= 31) and $\overline{DSACKx}$ low to $\overline{BERR}$ low setup time (= 48) can be ignored. The data must only 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: $\overline{DSACK0}$, $\overline{DSACK1}$, $\overline{CDIS}$, $\overline{IPL0}$-$\overline{IPL2}$, $\overline{DBEN}$, $\overline{AVEC}$, $\overline{BERR}$.

Figure 10. Write Cycle Timing Diagram (Continued)



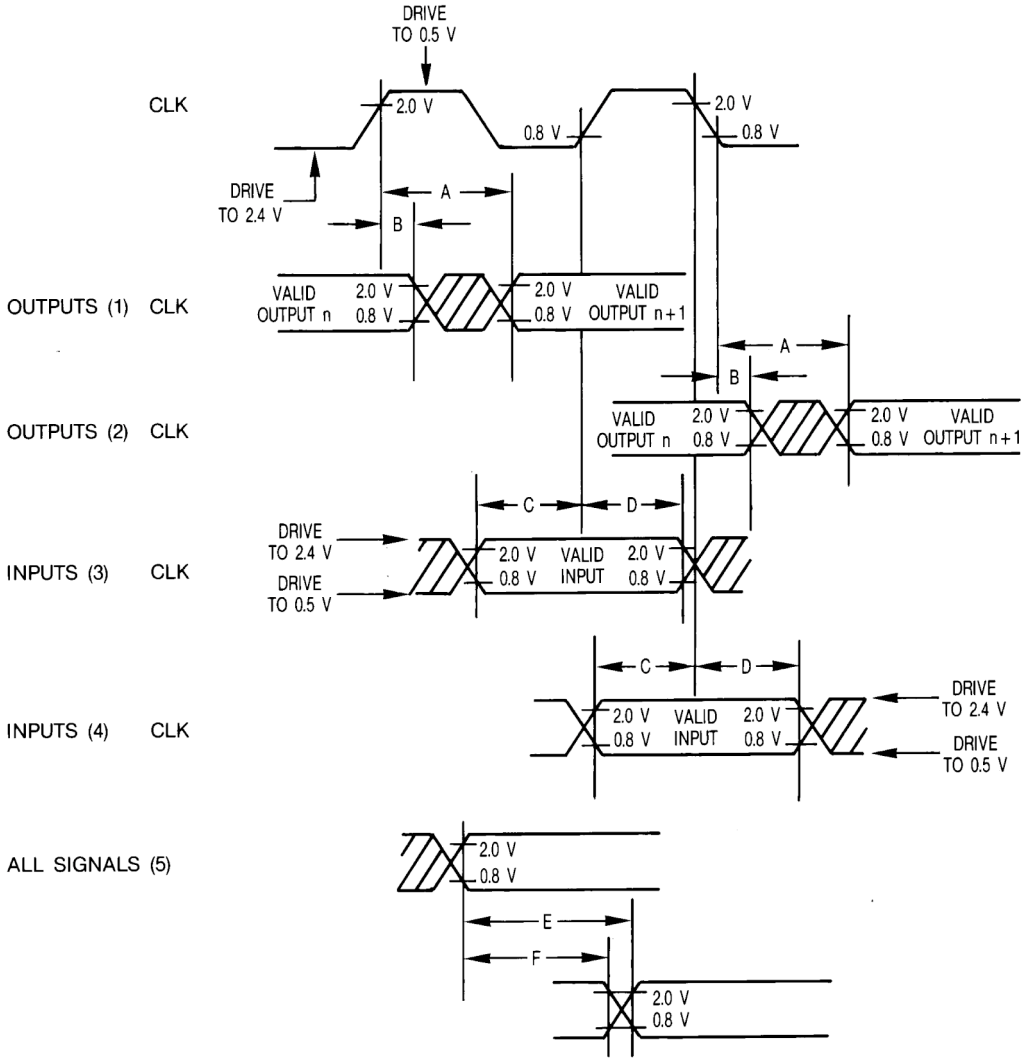
Note: Timing measurements are referenced to and from a low voltage of 0.8V and a high voltage of 2.0V, unless otherwise noted. The voltage swing thorough this range should start outside and pass through the range such that the rise or fall will be linear between 0.8V and 2.0V.

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.

Figure 12. Drive Levels and Test Points for AC Specification



Legend:

A) Maximum Output Delay Specification

B) Minimum Output Hold Time

C) Minimum Input Setup Time Specification

D) Minimum Input Hold Time Specification

E) Signal Valid to Signal Valid Specification (Maximum or Minimum)

F) Signal Valid to Signal Invalid Specification (Maximum or Minimum)

Notes:
1. This output timing is applicable to all parameters specified relative to the rising edge of the clock.
2. This out put timing is applicable to all parameters specified relative to the falling edge of the clock.
3. This input timing is applicable to all parameters specified relative to the falling edge of the clock.
4. This input timing is applicable to all parameters specified relative to the falling edge of the clock.
5. This timing is applicable to all parameters specified relative to the assertion/negation of another signal.

Figure 14. ECS and OCS Capacitance Derating Curve

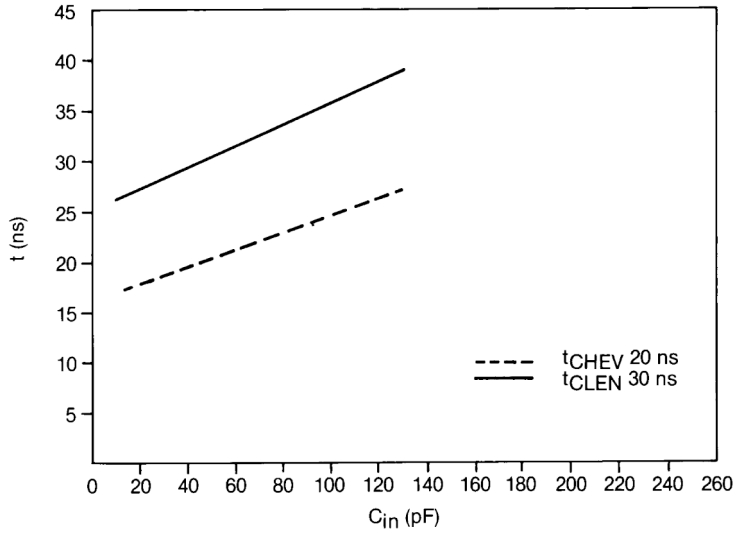


Figure 15. R/W, FC, SIZ0-SIZ1, and RMC Capacitance Derating Curve

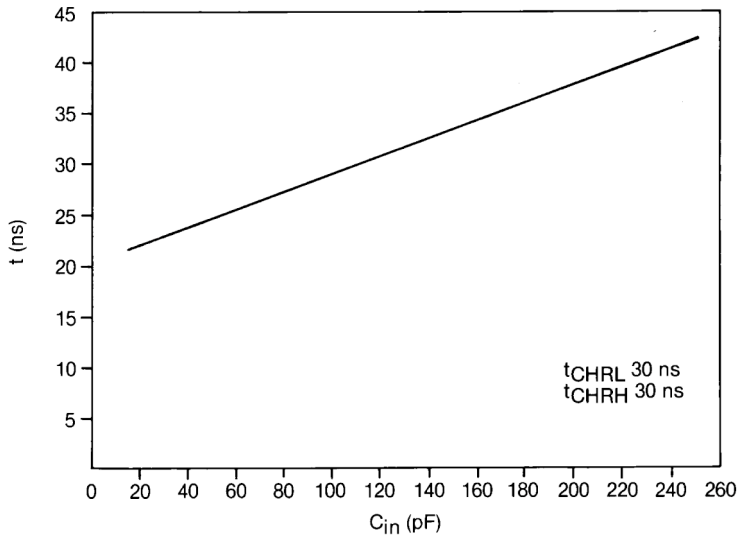
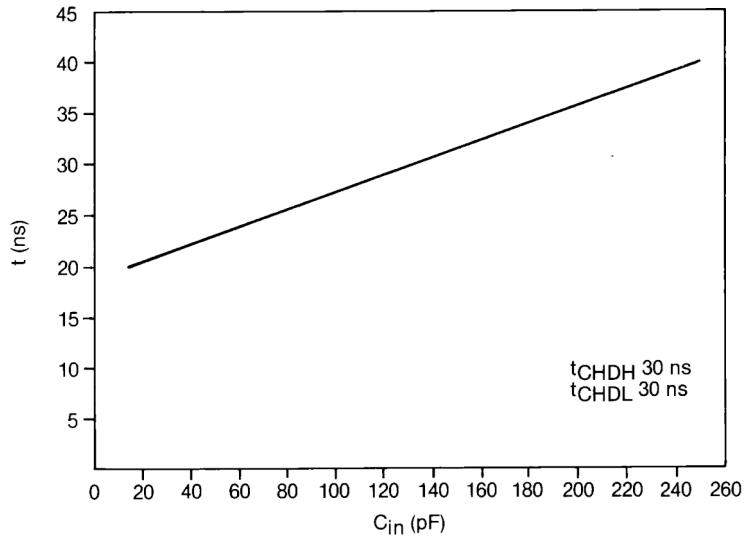


Figure 18. Data Capacitance Derating Curve



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).

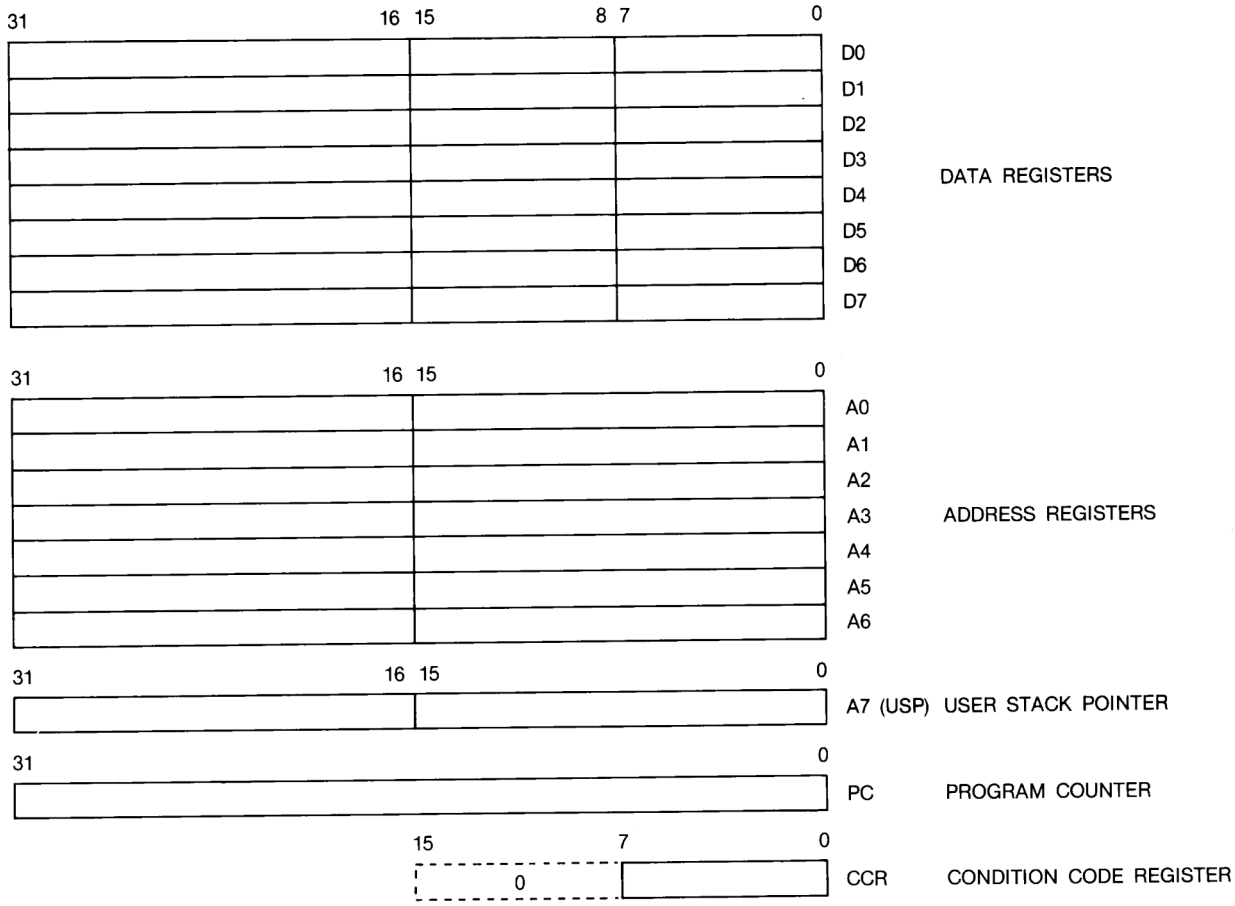
All microprocessors of the TS68000 Family support instruction tracing (via the T0 status bit in the TS68020) where each instruction executed is followed by a trap to a user-defined trace routine. The TS68020 adds the capability to trace only the change of flow instructions (branch, jump, subroutine call 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.

The TS68000 Family processors distinguish address spaces as supervisor / used and program/data. These four combinations are specified by the function code pins (FC0/FC1/FC2) during bus cycles, indication the particular address space. Using the function codes, the memory sub-system can distinguish between authorized access (supervisor mode is privileged access) and unauthorized access (user mode may not have access to supervisor program or data areas). To support the full privileges of the supervisor, the alternate function code registers allow the supervisor to specify an access to user program or data areas by preloading the SFC/DFC registers appropriately.

The cache registers (control — CACR, address — CAAR) allow software manipulation of the on-chip instruction cache. Control and status accesses to the instruction cache are provided by the cache control register (CACR), while the cache address register (CAAR) holds the address for those cache control functions that require an address.

Figure 19. User Programming Model



The 18 addressing modes, shown in Table 8, include nine basic types:

- Register Direct
- Register Indirect
- Register Indirect with Index
- Memory Indirect
- Program Counter Indirect with Displacement
- Program Counter Indirect with Index
- Program Counter Memory Indirect
- Absolute
- Immediate

The register indirect addressing modes support postincrement, predecrement, offset, and indexing. Programmers find these capabilities particularly useful for handling advanced data structures common to sophisticated applications and high level languages. The program counter relative mode also has index and offset capabilities; programmers find that this addressing mode is required to support position-independent software. In addition to these addressing modes, the TS68020 provides data operand sizing and scaling; these features provide performance enhancements to the programmer.

Table 8. TS68020 Addressing Modes

Addressing Modes	Syntax
Register Direct	
Data Register Direct	Dn
Address Register Direct	An
Register Indirect	
Address Register Indirect	(An)
Address Register Indirect with Post Increment	(An) +
Address Register Indirect with Predecrement	– (An)
Address Register Indirect with Displacement	(d_{16}An)
Register Indirect with Index	
Address Register Indirect with Index (8-bit Displacement)	(d_8, An, Xn)
Address Register Indirect with Index (Base Displacement)	(bd, An, Xn)
Memory Indirect	
Memory Indirect Post-Indexed	([bd, An], Xn, od)
Memory Indirect Pre-Indexed	([bd, An, Xn], od)
Program Counter Indirect with Displacement	(d_{16}, PC)
Program Counter Indirect with Index	
PC Indirect with Index (8-bit Displacement)	(d_8, PC, Xn)
PC Indirect with Index (Base Displacement)	(bd, PC, Xn)
Program Counter Memory Indirect	
PC Memory Indirect Post-Indexed	([bd, PC], Xn, od)
PC Memory Indirect Pre-Indexed	([bd, PC, Xn]), od)

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.

Other microprocessors in the TS68000 Family can operate any TS68000 co-processor even though they may not have the hardware implementation of the co-processor interface as does the TS68020. Since the co-processor is operated through the co-processor interface registers which are accessed via normal asynchronous bus cycles, the co-processor may be used as a peripheral device. Software easily emulates the communication protocol by addressing the co-processor interface registers appropriately and passing the necessary commands and operands required by the co-processor.

The co-processor interface registers are implemented by the co-processor in addition to those registers implemented as extensions to the TS68020 programmer's model. For example, the TS68881 implements the co-processor interface registers shown in Table 10 and the registers in the programming model, including eight 80-bit floating-point data registers and three 32-bit control/status registers used by the TS68881 programmer.

Table 10. Co-processor Interface Registers

Register	Function	R/W
Response	Requests Action from CPU	R
Control	CPU	W
Save	Initiate Save of Internal State	R
Restore	Initiate Restore of Internal State	R/W
Operation Word	Current Co-processor Instruction	W
Command Word	Co-processor Specific Command	W
Condition Word	Condition to be Evaluated	W
Operand	32-bit Operand	R/W
Register Select	Specifies CPU Register or Mask	R
Instruction Address	Pointer to Co-processor Instruction	R/W
Operand Address	Pointer to Co-processor Operand	R/W

Table 11. Co-processor Primitives

Processor Synchronization Busy with Current Instruction Proceed with Next Instruction, If No Trace Service Interrupts and Re-query, If Trace Enable Proceed with Execution, Condition True/False
Instruction Manipulation Transfer Operation Word Transfer Words from Instruction Stream
Exception Handling Take Privilege Violation if S Bit Not Set Take Pre-Instruction Exception Take Mid-Instruction Exception Take Post-Instruction Exception

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.

Package Mechanical Data

Figure 23. 114-lead - Ceramic Pin Grid Array

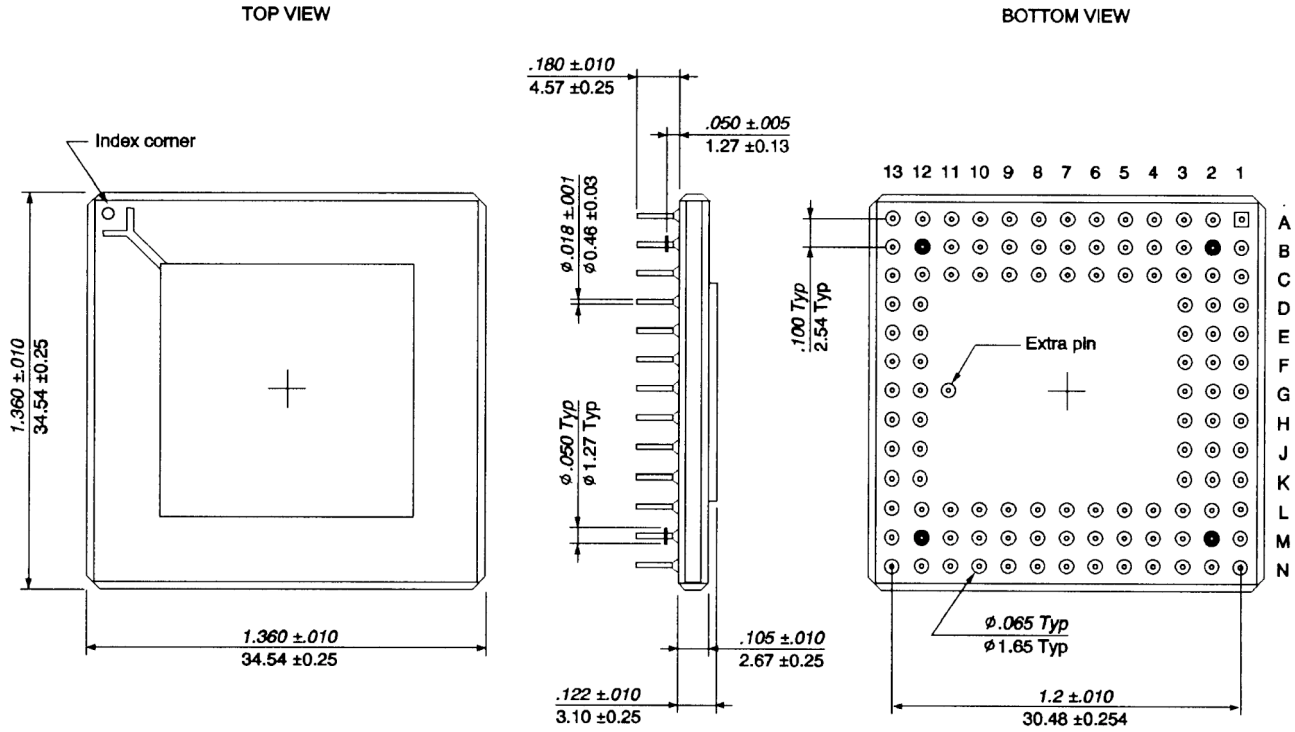
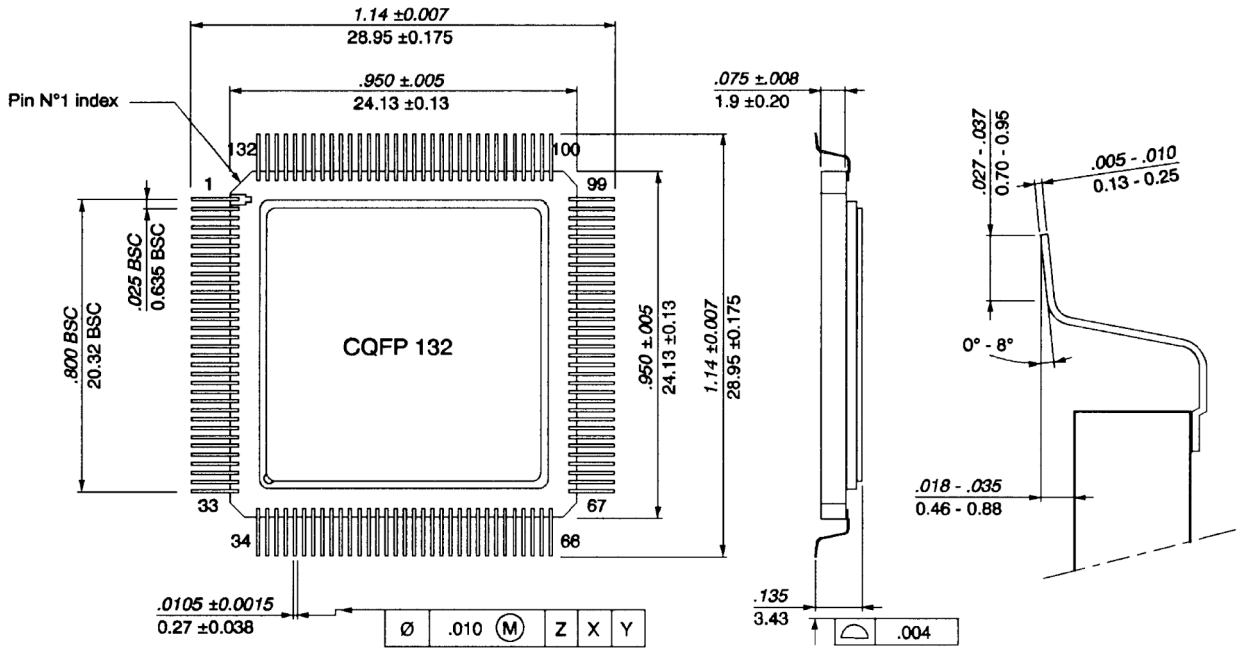


Figure 24. 132 Pins - Ceramic Quad Flat Pack



Mass

PGA 114 - 6 grams typically
CQFP 132 - 14 grams typically

Terminal Connections

114-lead - Ceramic Pin Grid Array

See Figure 2.

132-lead - Ceramic Quad Flat Pack

See Figure 3.

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

© Atmel Corporation 2002.
Atmel Corporation makes no warranty for the use of its products, other than those expressly contained in the Company's standard warranty which is detailed in Atmel's Terms and Conditions located on the Company's web site. The Company assumes no responsibility for any errors which may appear in this document, reserves the right to change devices or specifications detailed herein at any time without notice, and does not make any commitment to update the information contained herein. No licenses to patents or other intellectual property of Atmel are granted by the Company in connection with the sale of Atmel products, expressly or by implication. Atmel's products are not authorized for use as critical components in life support devices or systems.

ATMEL® is the registered trademark of Atmel.

Other terms and product names may be the trademarks of others.

Printed on recycled paper.

2115A–HIREL–07/02 0M