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	20MHz
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	132-BCQFP
Supplier Device Package	132-CQFP (24.13x24.13)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/ts68020vf20

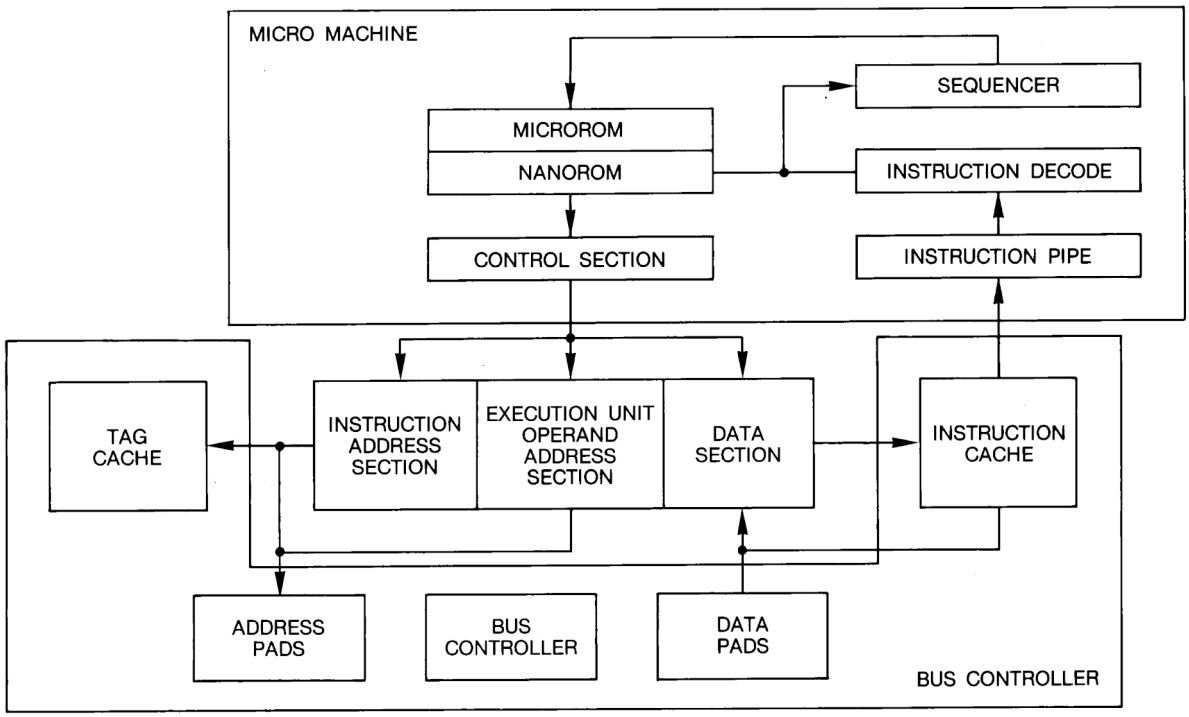
Introduction

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.

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

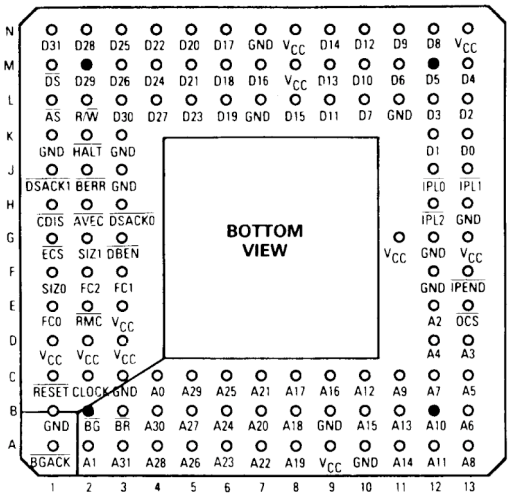


Figure 3. CQFP Terminal Designation

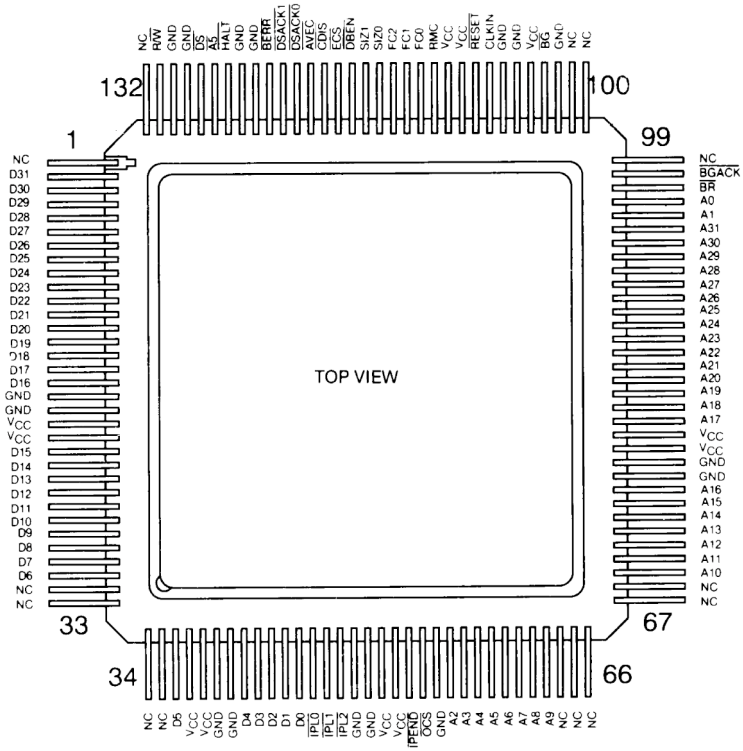
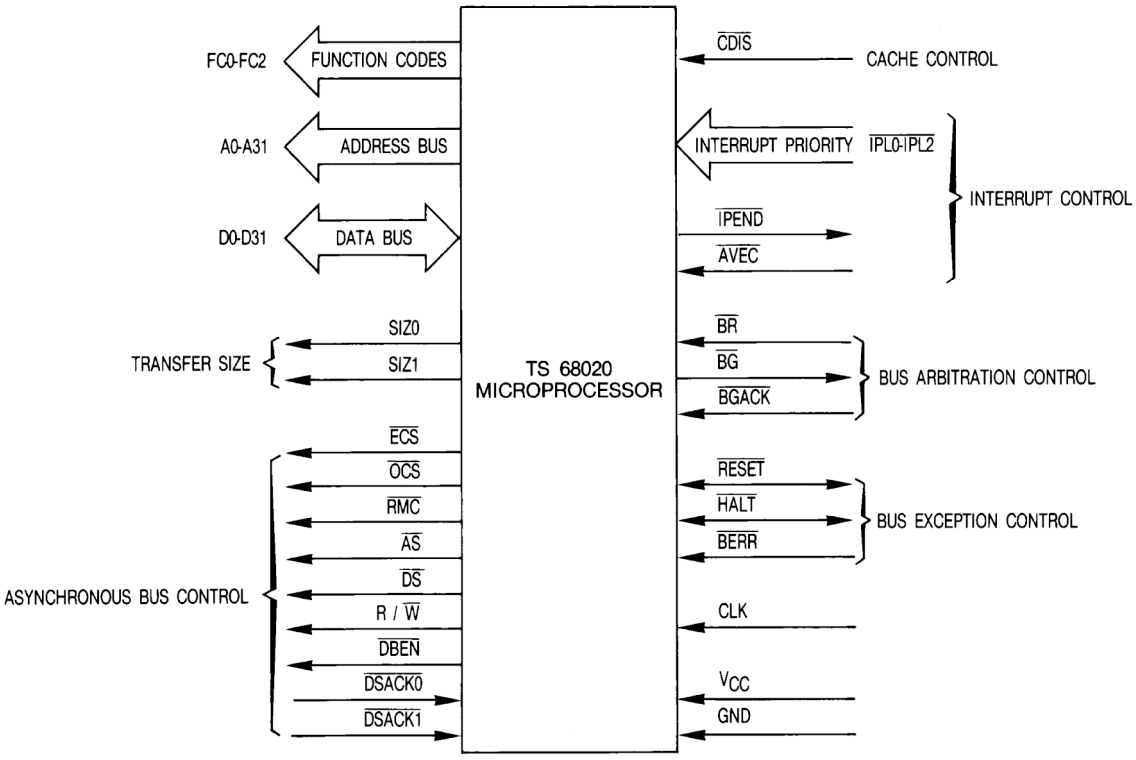


Figure 4. Functional Signal Groups



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

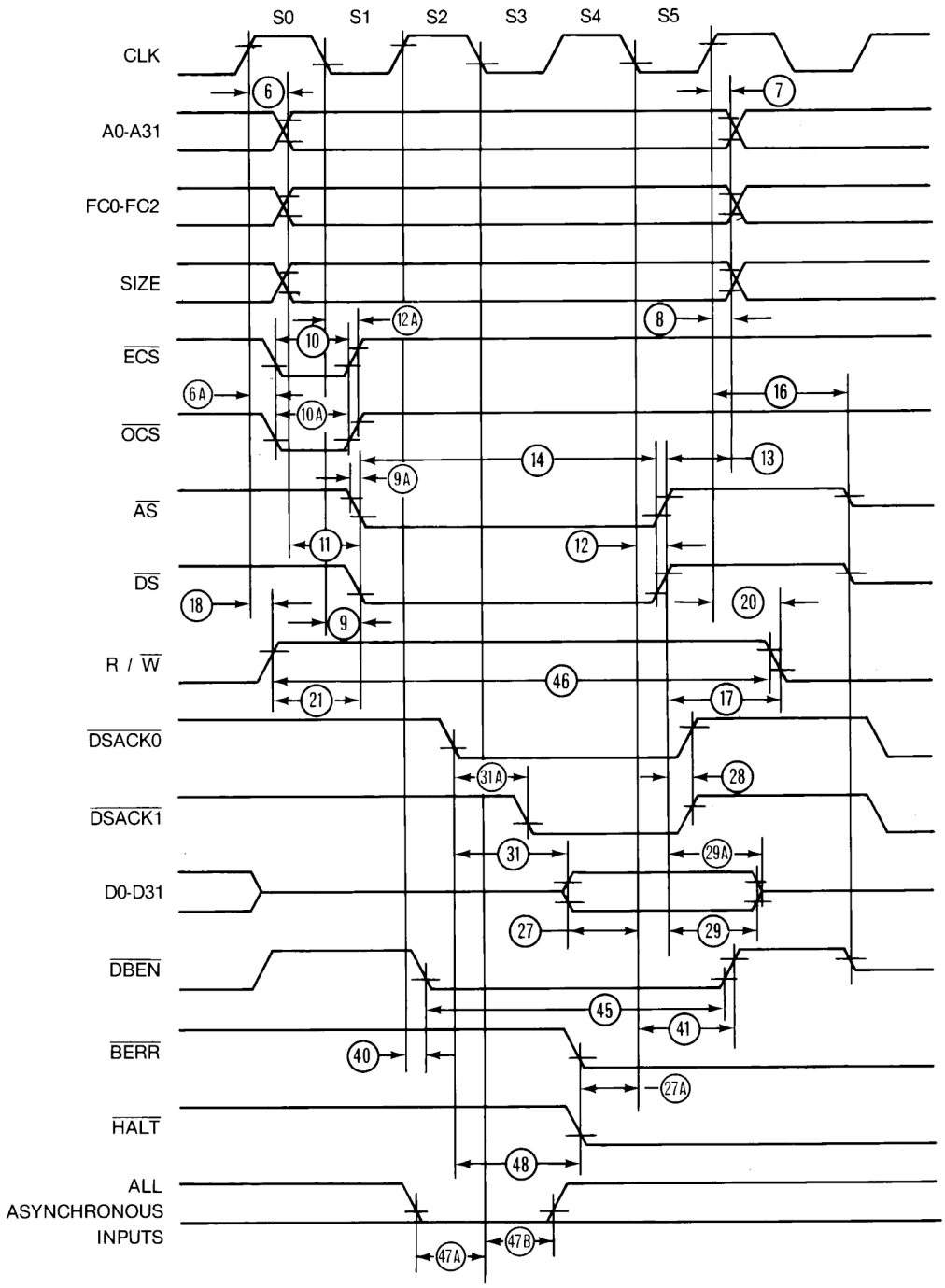
Table 6. Dynamic Electrical Characteristics (Continued)

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	

Time Definitions

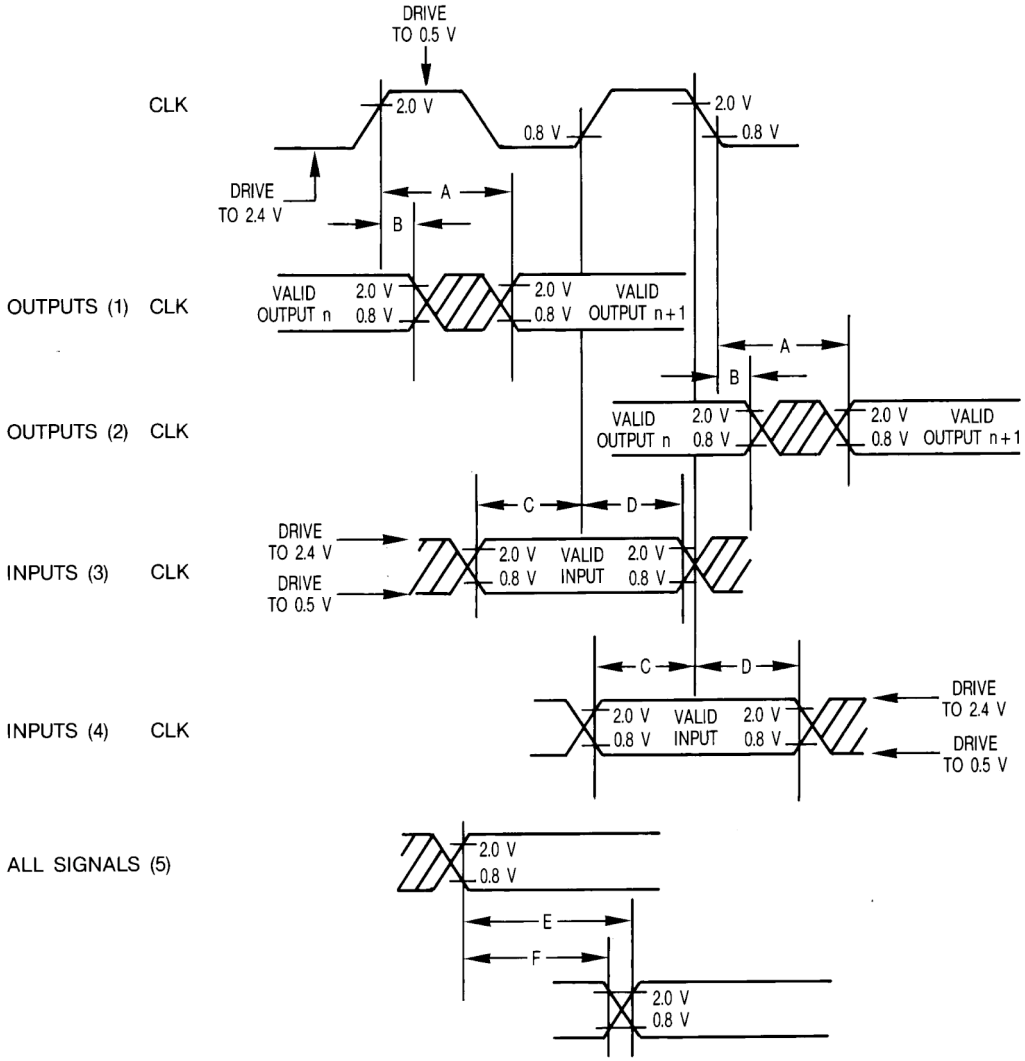
The times specified in Table 6 as dynamic characteristics are defined in Figure 9 below, by a reference number given the column “interval N°” of the tables together with the relevant figure number.

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 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.

Additional Information

Additional information shall not be for any inspection purposes.

Power Consideration

See Table 4.

Capacitance (Not for Inspection Purposes

Symbol	Parameter	Test Conditions	Min	Unit
C_{in}	Input Capacitance	V_{in} = 0V T_{amb} = 25°C f = 1 MHz	20	pF

Capacitance Derating Curves

Figure 13 to Figure 18 inclusive show the typical derating conditions which apply. The capacitance includes any stray capacitance. The graphs may not be linear outside the range shown.

Figure 13. Address Capacitance Derating Curve

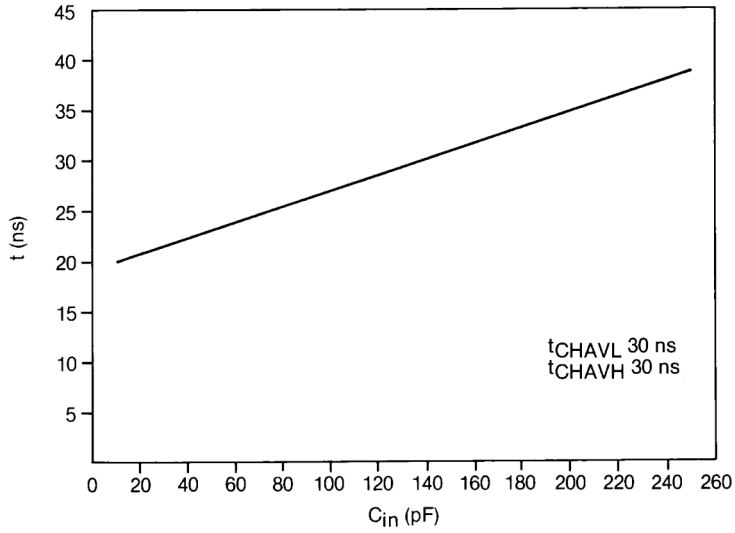
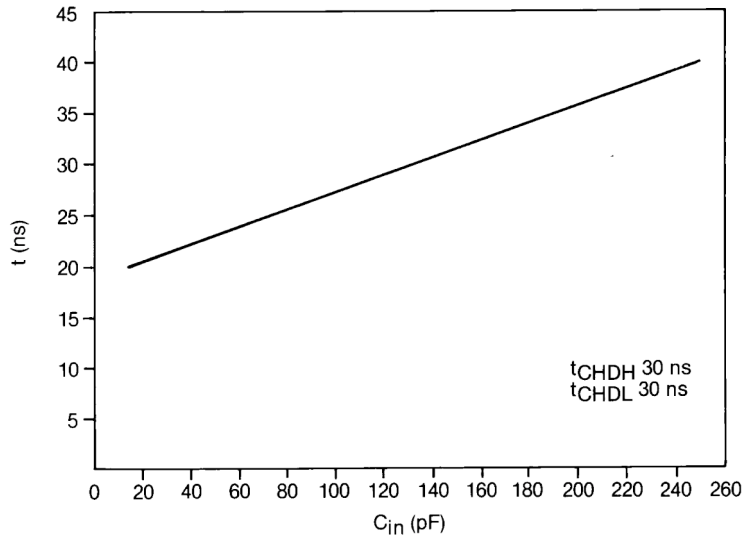


Figure 18. Data Capacitance Derating Curve



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.

Figure 20. Supervisor Programming Model Supplement

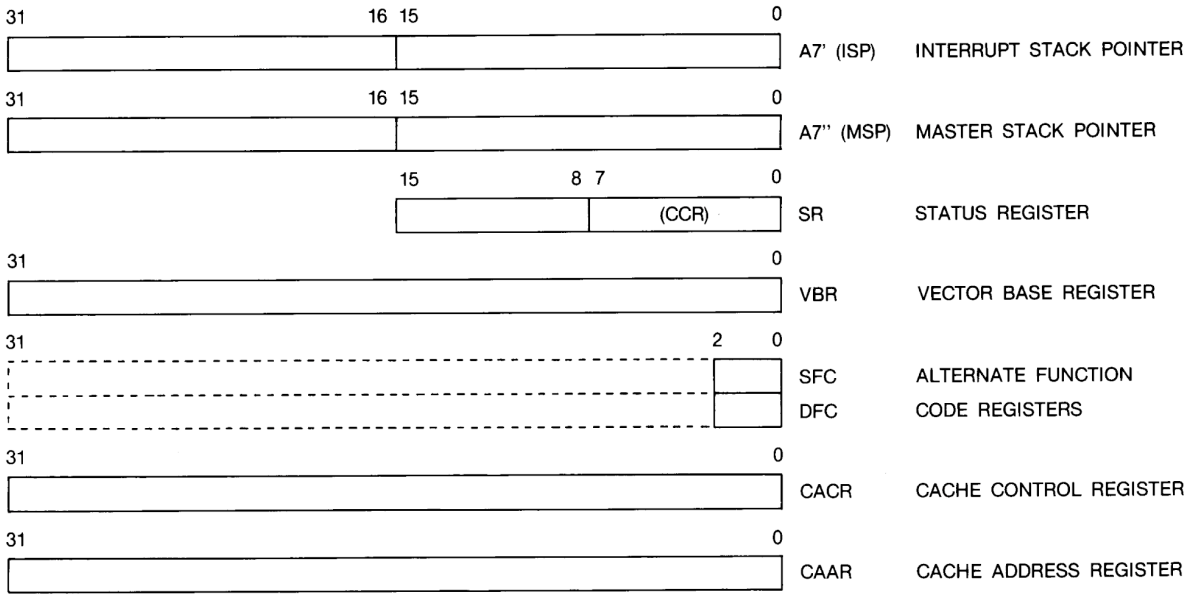
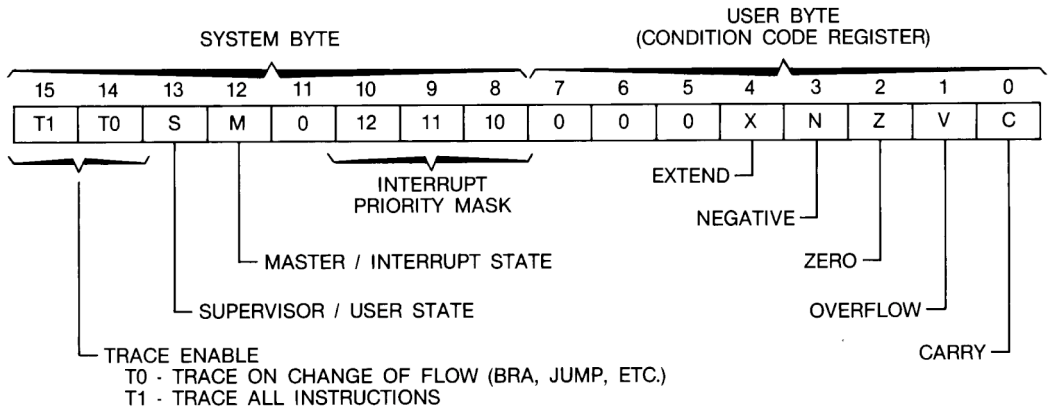


Figure 21. Status Register



Data Types and Addressing Modes

Seven basic types are supported. These data types are:

- Bits
- Bits Flieds (String of consecutive bits, 1-32 bits long)
- BCD Digits (Packed: 2 digits/byte, Unpacked: 1 digit/byte)
- Byte Integers (8-bit)
- Word Integers (16-bit)
- Long Word Integers (32-bit)
- Quad Word Integers (64-bit)

In addition, operations on other data types, such as memory addresses, status word data, etc...., are provided in the instruction set. The co-processor mechanism allows direct support of floating-point data type with the TS68881 and TS68882 floating-point co-processors, as well as specialized user-defined data types and functions.

Table 8. TS68020 Addressing Modes (Continued)

Addressing Modes	Syntax
Absolute	
Absolute Short	xxx.W
Absolute Long	xxx.L
Immediate	=data

Notes:
1. Dn = Data Register, D0-D7.
2. An = Address Register, A0-A7.
3. d_8, d_{16} = A twos-complement, or sign—extended displacement; added as part of the effective calculation; size is 8 (d_8) or 16 (d_{16}) bits; when omitted assemblers use a value of zero.
4. Xn = Address or data register used as an index register; form is Xn, SIZE*SCALE, where SIZE is.W or.L (indicates index register size) and SCALE is 1, 2, 4, or 8 (index register is multiplied by SCALE); use of SIZE and/or SCALE is optional.
5. bd = A two-complement base displacement; when present, size can be 16- or 32-bit.
6. od = Outer displacement, added as part of effective address calculation after any memory indirection; use is optional with a size of 16- or 32-bit.
7. PC = Program Counter.
8. (data) = Immediate value of 8, 16 or 32 bits.
9. () = Effective Address.
10. [ ] = Use as indirect address to long word address.

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 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.

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

Table 11. Co-processor Primitives (Continued)

General Operand Transfer
Evaluate and Pass (Ea.)
Evaluate (Ea.) and Transfer Data
Write to Previously Evaluated (Ea.)
Take Address and Transfer Data
Transfer to/from Top of Stack
Register Transfer
Transfer CPU Register
Transfer CPU Control Register
Transfer Multiple CPU Registers
Transfer Multiple Co-processor Registers
Transfer CPU SR and/or ScanPC

Up to eight processors are supported in a single system with a system-unique co-processor identifier encoded in the co-processor instruction. When accessing a co-processor, the TS68020 executes standard read and write bus cycle in CPU address space, as encoded by the function codes, and places the co-processor identifier on the address bus to be used by chip-select logic to select the particular co-processor. Since standard bus cycle are used to access the co-processor, the co-processor may be located according to system design requirements, whether it be located on the micro-processor local bus, on another board on the system bus, or any other place where the chip-select and co-processor protocol using standard TS68000 bus cycles can be supported.

Co-processor Protocol

Interprocessor transfers are all initiated by the main processor during co-processor instruction execution. During the processing of a co-processor instruction, the main processor transfers instruction information and data to the associated co-processor, and receives data, requests, and status information from the co-processor. These transfers are all based on the TS68000 bus cycles.

The typical co-processor protocol which the main processor follows is:

a) The main processor initiates the communications by writing command information to a location in the co-processor interface.

b) The main processor reads the co-processor response to that information.

1) The response may indicate that the co-processor is busy, and the main processor should again query the co-processor. This allows the main processor and co-processor to synchronize their concurrent operations.

2) The response may indicate some exception condition; the main processor acknowledges the exception and begins exception processing.

3) The response may indicate that the co-processor needs the main processor to perform some service such as transferring data to or from the co-processor. The co-processor may also request that the main processor query the co-processor again after the service is complete.

4) The response may indicate that the main processor is not needed for further processing of the instruction. The communication is terminated, and the main processor is free to begin execution of the next instruction. At this point in the co-processor protocol, as the main processor continues to execute the instruction stream, the main processor may operate concurrently with the co-processor.

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.

The TS68020 provides an extension to the exception stacking process. If the M bit in the status register is set, the master stack pointer (MSP) is used for all task related exceptions. When a non-task exception occurs (i.e., an interrupt), the M bit is cleared and the interrupt stack pointer (ISP) is used. This feature allows all the task's stack area to be carried within a single processor control block and new tasks may be initiated by simply reloading the master stack pointer and setting the M bit.

The fourth and last step of the exception processing is the same for all exceptions. The exception vector offset is determined by multiplying the vector number by four. This offset is then added to the contents of the vector base register (VBR) to determine the memory address of the exception vector. The new program counter value is fetched from the exception vector. The instruction at the address given in the exception vector is fetched, and the normal instruction decoding and execution is started.

On-chip Instruction Cache

Studies have shown that typical programs spend most of their execution time in a few main routines or tight loops. This phenomenon is known as locality of reference, and has an impact on performance of the program. The TS68020 takes limited advantage of this phenomenon in the form of its loop mode operation which allows certain instructions, when coupled with the DBcc instruction, to execute without the overhead of instruction fetches. In effect, this is a three word cache. Although the cache hardware has been supplied in a full range of computer systems for many years, technology now allows this feature to be integrated into the microprocessor.

TS68020 Cache Goals

There were two primary goals for the TS68020 microprocessor cache. The first design goal was to reduce the processor external bus activity. In a given TS68000 system, the TS68000 processor will use approximately 80 to 90 percent (for greater) of the available bus bandwidth. This is due to its extremely efficient perfecting algorithm and the overall speed of its internal architecture design. Thus, in an TS68000 system with more than one bus master (such as a processor and DMA device) or in a multiprocessor system, performance degradation can occur due to lack of available bus bandwidth. Therefore, an important goal for an TS68020 on-chip cache was to provide a substantial increase in the total available bus bandwidth.

The second primary design goal was to increase effective CPU throughput as larger memory sizes or slower memories increased average access time. By placing a high speed cache between the processor and the rest of the memory system, the effective access time now becomes:

$$t_{ACC} = h^{**}t_{CACHE} = (1 - h)^{*}t_{ext}$$

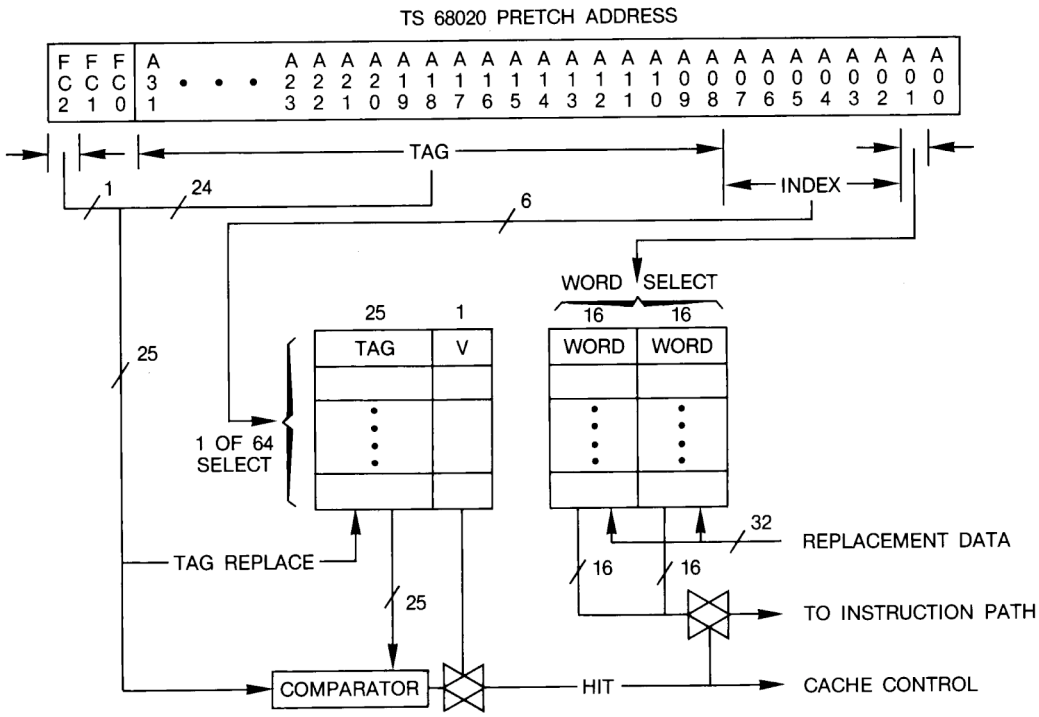
where t_{ACC} is the effective system access time, t_{CACHE} is the cache access time, t_{ext} is the access time of the rest of the system, and h is the hit ratio or the percentage of time that the data is found in the cache. Thus, for a given system design, an TS68020 on-chip cache provides a substantial CPU performance increase, or allows much slower and less expensive memories to be used for the same processor performance.

The throughput increase in the TS68020 is gained in two ways. First, the TS68020 cache is accessed in two clock cycles versus the three cycles (minimum) required for an external access. Any instruction fetch that is currently resident in the cache will provide a 33% improvement over the corresponding external access.

Second, and probably the most important benefit of the cache, is that it allows instruction stream fetches and operand accesses to proceed in parallel. For example, if the TS68020 requires both an instruction stream access and an operand access, and the instruction is resident in the cache, the operand access will proceed unimpeded rather than being queued behind the instruction fetch. Similarly, the TS68020 is fully capable of executing several internal instructions (instructions that do not require the bus) while completing an operand access for another instruction.

The TS68020 instruction cache is a 256-byte direct mapped cache organized as 64 long word entries. Each cache entry consists of a tag field made up of the upper 24 address bits, the FC2 (user/supervisor) value, one valid bit, and 32-bit of instruction data (Figure 22).

Figure 22. TS68020 On-chip Cache Organization



The TS68020 employs a 32-bit data bus and fetches instructions on long word address boundaries. Hence, each 32-bit instruction fetch brings in two 16-bit instruction words which are then written into the on-chip cache. When the cache is enabled, the subsequent prefetch will find the next 16-bit instruction word is already present in the cache and the related bus cycle is saved. If the cache were not enabled, the subsequent prefetch will find the bus controller still holds the full 32-bit and can satisfy the prefetch and again save the related bus cycle. So, even when the on-chip instruction cache is not enabled, the bus controller provides an instruction “cache hit” rate up to 50%.

Package Mechanical Data

Figure 23. 114-lead - Ceramic Pin Grid Array

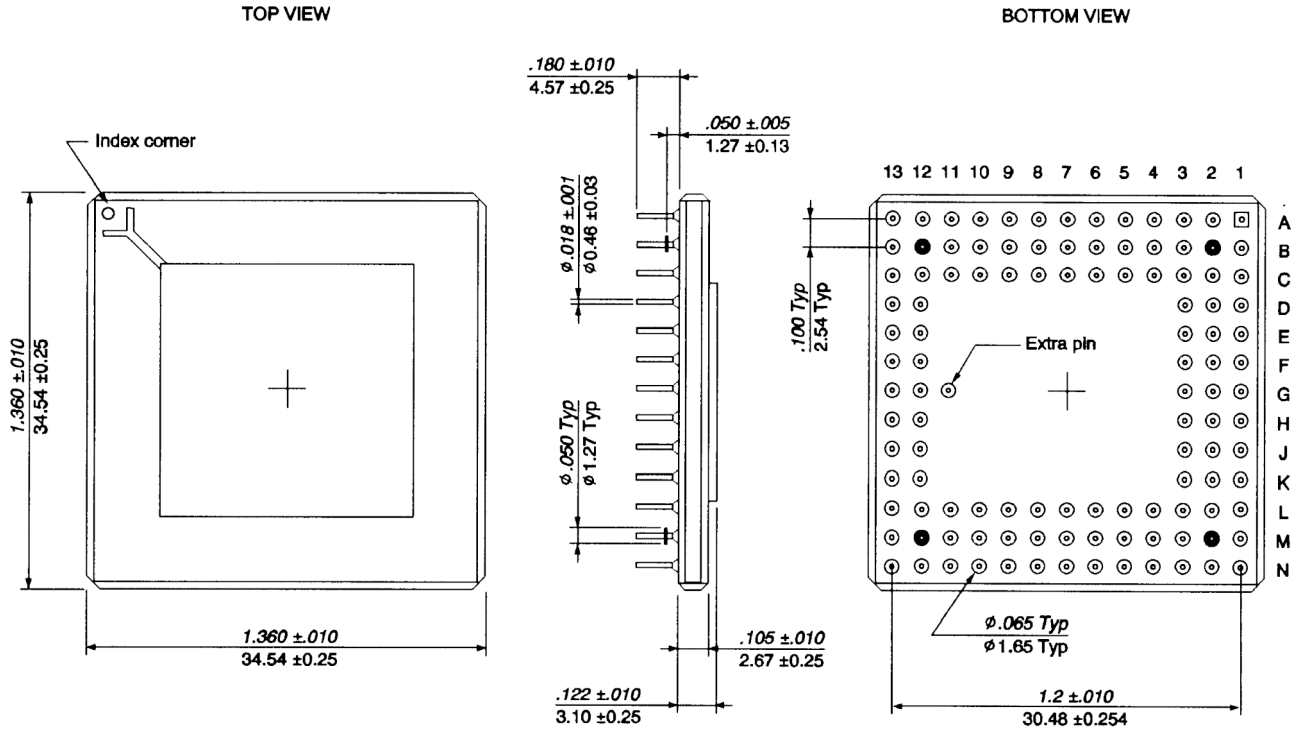
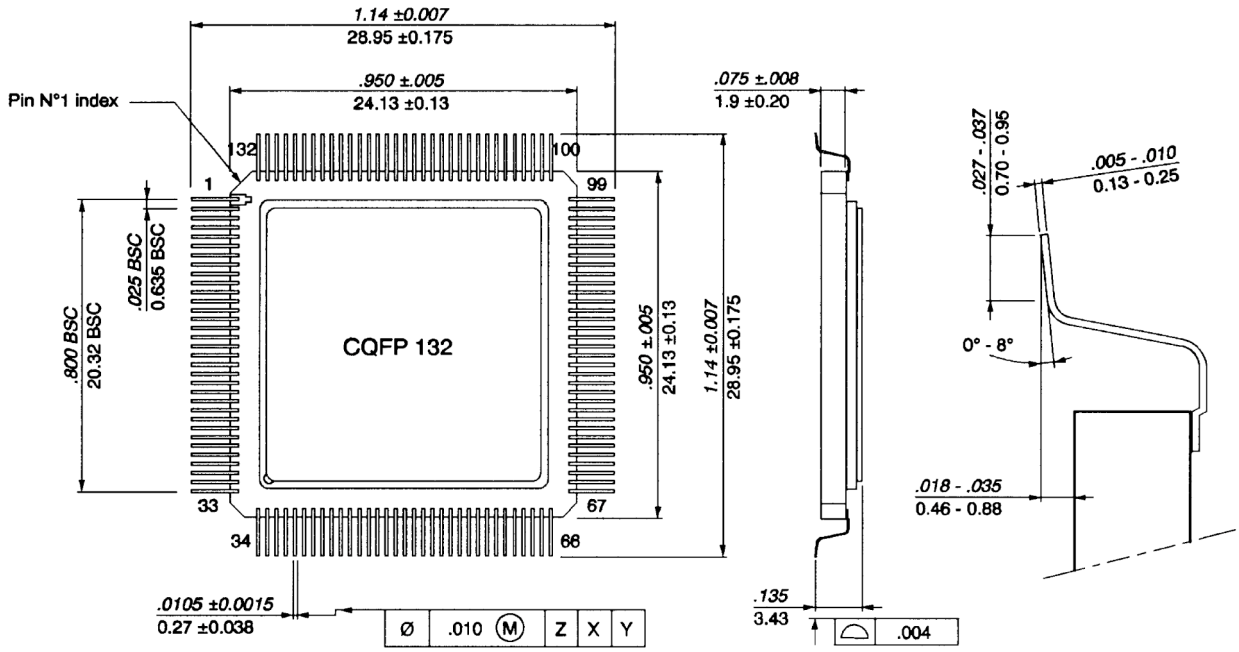


Figure 24. 132 Pins - Ceramic Quad Flat Pack



Ordering Information

Hi-REL Product

Commercial Atmel Part-Number	Norms	Package	Temperature Range T_c (°C)	Frequency (MHz)	Drawing Number
TS68020MRB/C16	MIL-STD-883	PGA 114	-55/+125	16.67	-
TS68020MR1B/C16	MIL-STD-883	PGA 114/tin	-55/+125	16.67	-
TS68020MRB/C20	MIL-STD-883	PGA 114	-55/+125	20	-
TS68020MR1B/C20	MIL-STD-883	PGA 114/tin	-55/+125	20	-
TS68020MRB/C25	MIL-STD-883	PGA 114	-55/+125	25	-
TS68020MR1B/C25	MIL-STD-883	PGA 114/tin	-55/+125	25	-
TS68020MFB/C16	MIL-STD-883	CQFP 132	-55/+125	16.67	-
TS68020MF1B/C16	MIL-STD-883	CQFP 132/tin	-55/+125	16.67	-
TS68020MFB/C20	MIL-STD-883	CQFP 132	-55/+125	20	-
TS68020MF1B/C20	MIL-STD-883	CQFP 132/tin	-55/+125	20	-
TS68020MFB/C25	MIL-STD-883	CQFP 132	-55/+125	25	-
TS68020MF1B/C25	MIL-STD-883	CQFP 132/tin	-55/+125	25	-
TS68020DESC02XA	DESC	PGA 114/tin	-55/+125	16.67	5962-8603202XA
TS68020DESC03XA	DESC	PGA 114/tin	-55/+125	20	5962-8603203XA
TS68020DESC04XA	DESC	PGA 114/tin	-55/+125	25	5962-8603204XA
TS68020DESC02XC	DESC	PGA 114	-55/+125	16.67	5962-8603202XC
TS68020DESC03XC	DESC	PGA 114	-55/+125	20	5962-8603203XC
TS68020DESC04XC	DESC	PGA 114	-55/+125	25	5962-8603204XC
TS68020DESC02YA	DESC	CQFP 132/tin	-55/+125	16.67	5962-8603202YA
TS68020DESC03YA	DESC	CQFP 132/tin	-55/+125	20	5962-8603203YA
TS68020DESC04YA	DESC	CQFP 132/tin	-55/+125	25	5962-8603204YA
TS68020DESC02YC	DESC	CQFP 132	-55/+125	16.67	5962-8603202YC
TS68020DESC03YC	DESC	CQFP 132	-55/+125	20	5962-8603203YC
TS68020DESC04YC	DESC	CQFP 132	-55/+125	25	5962-8603204YC

Standard Product

Commercial Atmel Part-Number	Norms	Package	Temperature Range T_c (°C)	Frequency (MHz)	Drawing Number
TS68020VR16	Internal Standard	PGA 114	-40/+85	16.67	Internal
TS68020VR20	Internal Standard	PGA 114	-40/+85	20	Internal
TS68020VR25	Internal Standard	PGA 114	-40/+85	25	Internal
TS68020MR16	Internal Standard	PGA 114	-55/+125	16.67	Internal
TS68020MR20	Internal Standard	PGA 114	-55/+125	20	Internal
TS68020MR25	Internal Standard	PGA 114	-55/+125	25	Internal

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