

**HIGH-PERFORMANCE 32-BIT INTEGRATED  
MICROCONTROLLER**

**DESCRIPTION**

The TS 68332 is a 32-bit microcontroller, combining high-performance data manipulation capabilities with powerful peripheral subsystems. The TS 68332 is the first member of the 68300 family of modular embedded controllers featuring fully static, high-speed complementary metal-oxide semiconductor technology. Based on the powerful TS 68020, the CPU32 instruction processing module provides enhanced system performance and utilizes the extensive software base of the 68000 family.

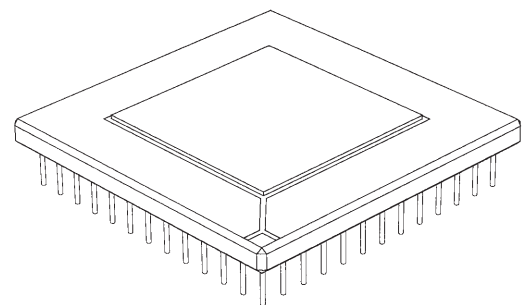
**MAIN FEATURES**

- Low-power operation including special STOP mode.
- Frequency : 16.78 MHz at 5 V  $\pm$  10 % supply and 20.97 MHz at 5 V  $\pm$  5 %, software programmable.
- Technology : 1  $\mu$  high-density complementary metal-oxide semiconductor (HCMOS), static design.
- Package : 132-pin Ceramic Leaded Chip Carrier (CERQUAD) and 132-pin Ceramic Pin Grid Array (PGA).
- Modular architecture in a single chip.
- CPU : 32-bit 68000 family (upward object-code compatible with the 68010).
- New instructions for controller applications.
- Intelligent 16-bit timer :
  - 16 independent, programmable channels,
  - Any channel can perform any time function (for example input capture, output compare, pulse width modulation, etc.),
  - Two timer count registers with 2-bit programmable prescalers,
  - Selectable channel priority levels,
  - Reduced CPU intervention,
  - RISC like CPU within the TPU.
- Two serial I/O subsystems :
  - Enhanced 68HC11-type serial communications interface (SCI) universal asynchronous receiver transmitter (UART) with parity,
  - Enhanced 68HC11-type serial peripheral interface with I/O RAM Queue (QSPI).
- On-chip memory : 2 K bytes standby RAM.
- On-chip, programmable, chip-select logic
  - Up to 12 signals for memory and peripheral interface with I/O select.
- System failure protection :
  - 68HC11-type Computer Operating Properly (COP) watchdog timer,
  - 68HC11-type periodic interrupt timer,
  - 68000 family spurious interrupt, HALT, and bus time-out monitors.
- Up to 48 discrete I/O pins.

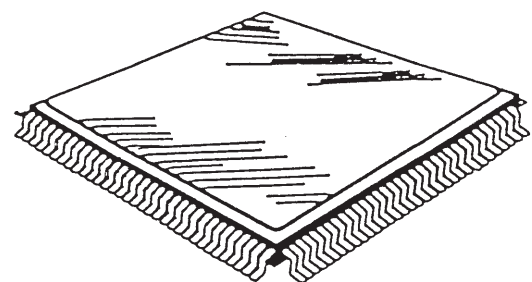
**SCREENING / QUALITY**

This product is manufactured in full compliance with :

- MIL-STD-883 (class B).
- DSCC 5962-91501.
- or according to TCS standard.



**R Suffix  
PGA 132**  
Ceramic Pin Grid Array



**A Suffix  
CERQUAD 132**  
Ceramic Leaded Chip Carrier

See the ordering information page 36.

Pin connection : see page 4.

## **SUMMARY**

### **A - GENERAL DESCRIPTION**

- 1 - INTRODUCTION
- 2 - SIGNAL DESCRIPTION

### **B - DETAILED SPECIFICATION**

- 1 - SCOPE
- 2 - APPLICABLE DOCUMENTS
  - 2.1 - MIL-STD-883
- 3 - REQUIREMENTS
  - 3.1 - General
  - 3.2 - Design and construction
  - 3.3 - Electrical characteristics
  - 3.4 - Thermal characteristics (at 25°C)
  - 3.5 - Mechanical and environment
  - 3.6 - Marking
- 4 - Quality conformance inspection
  - 4.1 - DESC / MIL-STD-883
- 5 - Electrical characteristics
  - 5.1 - General requirements
  - 5.2 - Static characteristics
  - 5.3 - Dynamic characteristics
  - 5.4 - Test conditions specific to the device
- 6 - FUNCTIONAL DESCRIPTION
  - 6.1 - Module memory map
  - 6.2 - CPU32 overview
  - 6.3 - Bus operation
  - 6.4 - System integration module
  - 6.5 - QSM Queued Serial Module
  - 6.6 - Standby RAM (with TPU emulation)
  - 6.7 - TPU overview
- 7 - PREPARATION FOR DELIVERY
  - 7.1 - Packaging
  - 7.2 - Certificate of compliance
- 8 - HANDLING
- 9 - PACKAGE MECHANICAL DATA
  - 9.1 - 132 pins - Ceramic Pin Grid Array
  - 9.2 - 132 pins - Ceramic Leaded Chip Carrier
- 10 - TERMINAL CONNECTIONS
  - 10.1 - 132 pins - Ceramic Pin Grid Array
  - 10.2 - 132 pins - Ceramic Leaded Chip Carrier (CERQUAD)
- 11 - ORDERING INFORMATION
  - 11.1 - Hi-REL product
  - 11.2 - Standard product



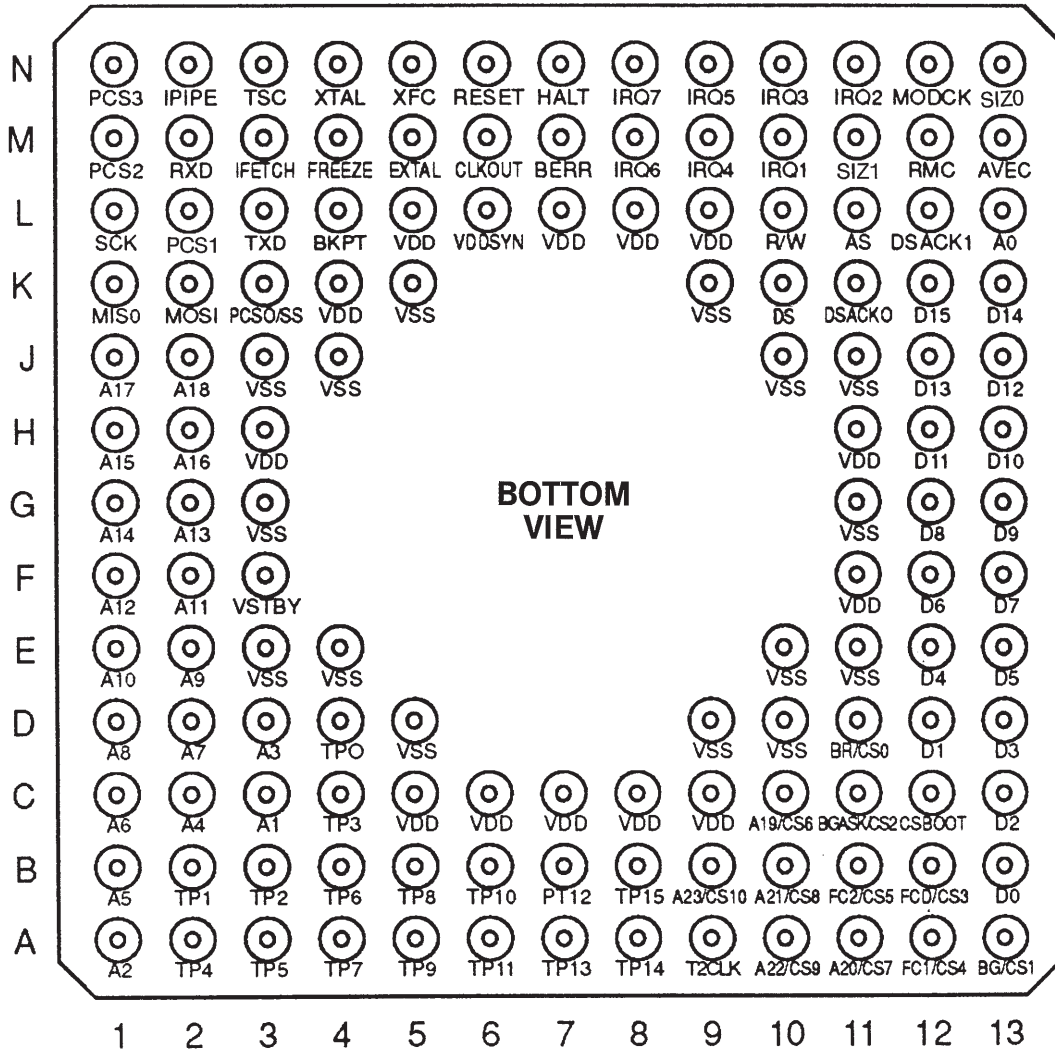


Figure 2.1 : PGA terminal designation.

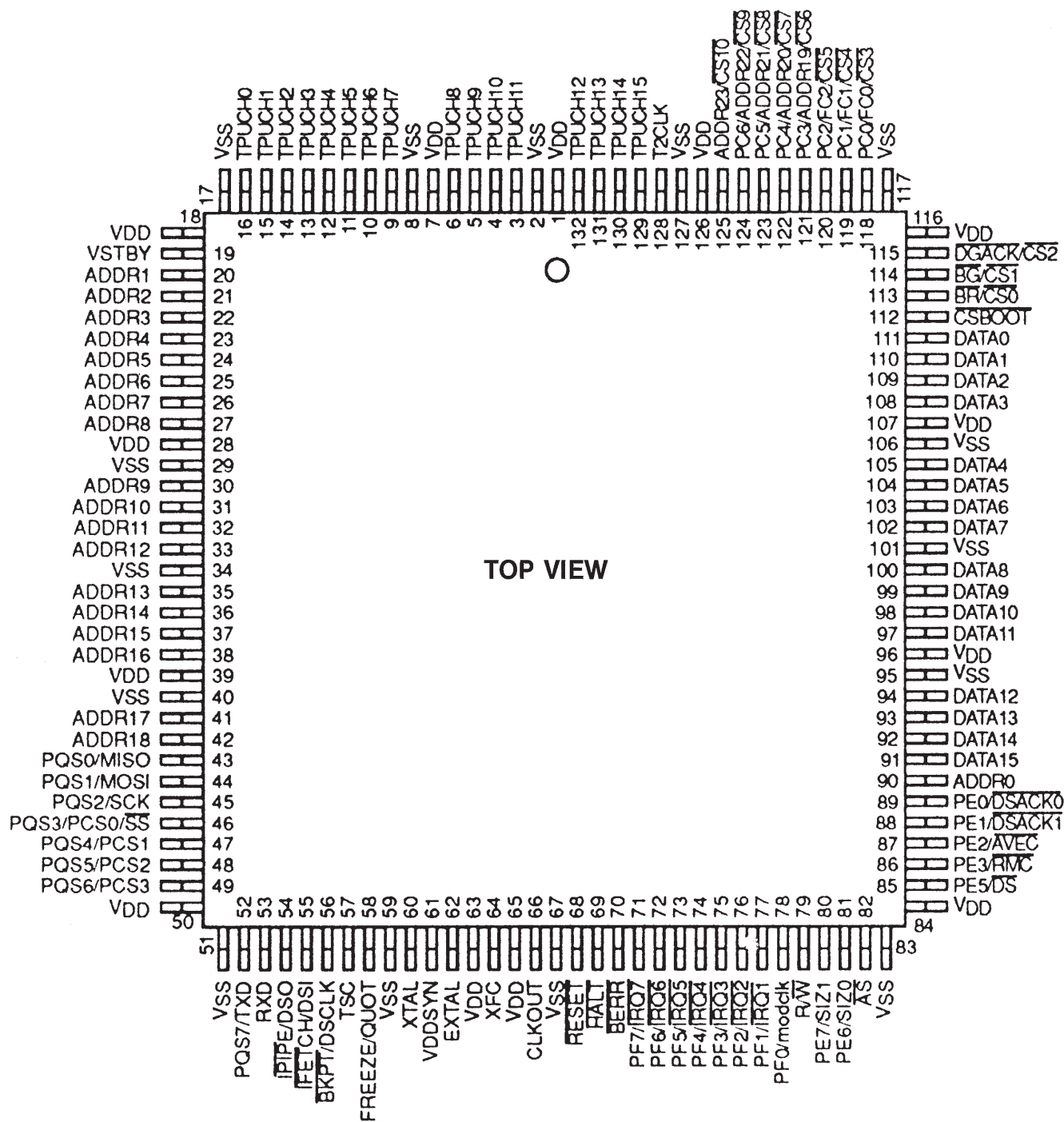


Figure 2.2 : CERQUAD terminal designation.

## 2 - SIGNAL DESCRIPTION

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

**Table 1 - Signal index**

Signal name	Mnemonic	Function
Address Bus	A23-A0	24-bit address bus
Data Bus	D15-D0	16-bit data bus used to transfer byte or word data per bus cycle
Data Bus Function Codes	FC2-FC0	Identify the processor state and the address space of the current bus cycle
Boot Chip Select	$\overline{\text{CSBOOT}}$	Chip-select boot startup ROM containing user's reset vector and initialisation program
Chip selects	$\overline{\text{CS10-CS0}}$	Enables peripherals at programmed addresses
Bus Request	$\overline{\text{BR}}$	Indicates that an external device requires bus mastership
Bus Grant	$\overline{\text{BG}}$	Indicates that current bus cycle is complete and the TS 68332 has relinquished the bus
Bus Grant Acknowledge	$\overline{\text{BGACK}}$	Indicates that an external device has assumed bus mastership
Data and Size Acknowledgement	$\overline{\text{DSACK1}}, \overline{\text{DSACK0}}$	Provides asynchronous data transfers and dynamic bus sizing
Autovector	$\overline{\text{AVEC}}$	Requests an automatic vector during an interrupt acknowledge cycle
Read-Modify-Write Cycle	$\overline{\text{RMC}}$	Identifies the bus cycle as part of an indivisible read-modify-write cycle
Address Strobe	$\overline{\text{AS}}$	Indicates that a valid address is on the address bus
Data Strobe	$\overline{\text{DS}}$	During a read cycle, DS indicates that an external device should place valid data on the data bus. During a write cycle, DS indicates that valid data is on the data bus
Size	SIZ1 - SIZ0	Indicates the number of bytes remaining to be transferred for this cycle
Read/Write	$\overline{\text{R/W}}$	Indicates the direction of data transfer on the bus
Interrupt Request Level	$\overline{\text{IRQ7}} - \overline{\text{IRQ0}}$	Provides an interrupt priority level to the CPU
Reset	$\overline{\text{RESET}}$	System reset
Halt	$\overline{\text{HALT}}$	Suspend external bus activity
Bus Error	$\overline{\text{BERR}}$	Indicates that an erroneous bus operation is being attempted
System Clockout	CLKOUT	Internal system clock
Crystal Oscillator	EXTAL, XTAL	Connection for an external crystal to the internal oscillator circuit
External Filter Capacitor	XFC	Connection pin for an external capacitor to filter the circuit of the phase-locked loop
Clock Mode Select	MODCK	Selects the source of the internal system clock
Instruction Fetch	$\overline{\text{IFETCH}}$	Indicates when the CPU is performing an instruction word pre-fetch and when the instruction pipeline has been flushed
Instruction Pipe	$\overline{\text{IPIPE}}$	Used to track movement of words through the instruction pipeline
Breakpoint	$\overline{\text{BKPT}}$	Signals a hardware breakpoint to the CPU

Table 1 - Signal index (continued)

Signal name	Mnemonic	Function
Freeze	FREEZE	Indicates that the CPU has acknowledged a breakpoint
Quotient Out	QUOT	Serial I/O and clock for background debug mode
Test Mode Enable	$\overline{\text{TSTME}}$	Hardware enable for test mode
Three-State Control	TSC	Places all output drivers in a high-impedance state
Development Serial In, Out, Clock	DSI, DSO, DSCLK	Serial I/O and clock for background debug mode
TPU Channels	TP15-TP0	TPU channel input/output Serial I/O and clock for background debug mode
TPU Clock In	T2CLK	External clock source to the TPU
SCI Receive Data	RXD	Serial input to the SCI
SCI Transmit Data	TXD	Serial output from the SCI
Peripheral Chip Select	$\overline{\text{PCS3}} - \overline{\text{PCS0}}$	QSPI peripheral chip selects
Slave Select	$\overline{\text{SS}}$	Places the QSPI in slave mode
QSPI Serial Clock	SCK	Furnishes the clock from the QSPI in master mode or to the QSPI in slave mode
Master-In Slave-Out	MISO	Furnishes serial input to the QSPI in master mode, and serial output from the QSPI in slave mode
Master-Out Slave-In	MOSI	Furnishes serial output from the QSPI in master mode, and serial input to the QSPI in slave mode
Standby RAM	VSTBY	Power supply for RAM
Synchronizer Power	VDDSYN	Power supply to VCO
System Power Supply and Return	VDD, VSS	Power supply and return to the MCU

## B - DETAILED SPECIFICATIONS

### 1 - SCOPE

This drawing describes the specific requirements for the microcontroller 68332 at 16.78 MHz and 20.97 MHz, in compliance either with MIL-STD-883 class B or TCS standard.

### 2 - APPLICABLE DOCUMENTS

#### 2.1 - MIL-STD-883

- 1) MIL-STD-883 : test methods and procedures for electronics.
- 2) MIL-I-38535 : general specifications for microcircuits.
- 3) Desc Drawing : 5962-91501.

### 3 - REQUIREMENTS

#### 3.1 - General

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

#### 3.2 - Design and construction

##### 3.2.1 - Terminal connections

Depending on the package, the terminal connections shall be is shown in Figures 2.1 and 2.2.

##### 3.2.2 - Lead material and finish

Lead material and finish shall be any option of MIL-STD-853.

### 3.2.3 - Package

The macrocircuits are packaged in hermetically sealed ceramic packages which conform to case outlines of MIL-M-38510 appendix C (when defined) :

- 132-PIN SQ.PGA UP PAE outline
- 132-PIN Ceramic CERQUAD

The precise case outlines are described in § 9.1 and § 9.2.

### 3.3 - Electrical characteristics

#### Absolute maximum ratings (see Table 2)

The following ratings define the conditions under which the device operates without damage. Sections of the device may not operate normally while being exposed to the electrical extremes and contains circuitry to protect against damage from high static voltages or electrical fields. It is advised, however, that normal precautions be taken to avoid application of any voltages higher than maximum-rated voltages to this high-impedance circuit. Reliability of operation is enhanced if unused inputs are tied to an appropriate logic voltage (e.g., either  $V_{SS}$  or  $V_{DD}$ ).

Table 2

Symbol	Parameter	Test conditions	Min	Max	Unit
$V_{DD}$	Supply voltage		-0.3	+7.0	V
$V_I$	Input voltage		-0.3	+7.0	V
$P_{dmax}$	Max Power dissipation	Low power operation		600	mW
		Stand by mode		500	mW
$T_{case}$	Operating temperature	M suffix	-55	+125	°C
		V suffix	-40	+85	°C
$T_{stg}$	Storage temperature		-55	+150	°C
$T_j$	Junction temperature			+160	°C
$T_{leads}$	Lead temperature	Max 5 sec. soldering		+270	°C

### 3.4 - Thermal characteristics (at 25°C)

Table 3

Package	Symbol	Parameter	Value	Unit
PGA 132	$\theta_{J-A}$	Thermal resistance ceramic Junction-to-Ambient	TBD	°C/W
	$\theta_{J-C}$	Thermal resistance ceramic Junction-to-Case	10	°C/W
CERQUAD 132	$\theta_{J-A}$	Thermal resistance ceramic Junction-to-Ambient	TBD	°C/W
	$\theta_{J-C}$	Thermal resistance ceramic Junction-to-Case	10	°C/W

#### Power considerations

The average chip-junction temperature,  $T_J$ , in °C can be obtained from :

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

$T_A$  = Ambient Temperature, °C

$\theta_{JA}$  = Package Thermal Resistance, Junction-to-Ambient, °C/W

$P_D$  =  $P_{INT} + P_{I/O}$

$P_{INT}$  =  $I_{CC} \times V_{CC}$ , Watts — Chip Internal Power

$P_{I/O}$  = Power Dissipation on Input and Output Pins — User Determined

For most applications  $P_{I/O} < P_{INT}$  and can be neglected.

An approximate relationship between  $P_D$  and  $T_J$  (if  $P_{I/O}$  is neglected) is :

$$P_D = K : (T_J + 273) \quad (2)$$

Solving equations (1) and (2) for K gives :

$$K = P_D \cdot (T_A + 273) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part K can be determined from equation (3) by measuring  $P_D$  (at equilibrium) for a known  $T_A$ . Using this value of K, the values of  $P_D$  and  $T_J$  can be obtained by solving equations (1) and (2) iteratively for any value of  $T_A$ .

The total thermal resistance of a package ( $\theta_{JA}$ ) can be separated into two components,  $\theta_{JC}$  and  $\theta_{CA}$ , representing the barrier to heat flow from the semiconductor junction to the package (case), surface ( $\theta_{JC}$ ) and from the case to the outside ambient ( $\theta_{CA}$ ). These terms are related by the equation :

$$\theta_{JA} = \theta_{JC} + \theta_{CA} \quad (4)$$

$\theta_{JC}$  is device related and cannot be influenced by the user. However,  $\theta_{CA}$  is user dependent and can be minimized by such thermal management techniques as heat sinks, ambient air cooling and thermal convection. Thus, good thermal management on the part of the user can significantly reduce  $\theta_{CA}$  so that  $\theta_{JA}$  approximately equals  $\theta_{JC}$ . Substitution of  $\theta_{JC}$  for  $\theta_{JA}$  in equation (1) will result in a lower semiconductor junction temperature.

### 3.5 - Mechanical and environment

The microcircuits shall meet all mechanical environmental requirements of either MIL-STD-883 for class B devices or screened according to TCS standards devices.

### 3.6 - Marking

The document where are defined the marking are identified in the related reference documents. Each microcircuit are legible and permanently marked with the following information as minimum :

#### 3.6.1 - Thomson logo

#### 3.6.2 - Manufacturer's part number

#### 3.6.3 - Class B identification

#### 3.6.4 - Date-code of inspection lot

#### 3.6.5 - ESD identifier if available

#### 3.6.6 - Country of manufacturing

## 4 - QUALITY CONFORMANCE INSPECTION

### 4.1 - DESC / MIL-STD-883

Is in accordance with MIL-M-38535 and method 5005 of MIL-STD-883. Group A and B inspections are performed on each production lot. Group C and D inspection are performed on a periodical basis.

## 5 - ELECTRICAL CHARACTERISTICS

### 5.1 - General requirements

All static and dynamic electrical characteristics specified and the relevant measurement conditions are given below. For inspection purpose, refer to relevant specification :

- DESC see § 4.1

(last issue on request to our marketing services).

Table 4 : Static electrical characteristics for all electrical variants. See § 5.2.

Table 6 : Dynamic electrical characteristics for 68332-16 (16.78 MHz) and 68332-20 (20.97 MHz). See § 5.3.

For static characteristics, test methods refer to IEC 748-2 method number, where existing.

For dynamic characteristics, test methods refer to clause 5.4 hereafter of this specification.

### 5.2 - Static characteristics

**Table 4 - DC characteristics**

$V_{DD}$  and  $V_{DDSYN} = 5.0 V_{dc} \pm 10\%$  for 16.78 MHz and  $5.0 V_{dc} \pm 5\%$  for 20.97 MHz ;  
 $V_{SS} = 0 V_{dc}$  ;  $T_c = -55^\circ\text{C}$  to  $+125^\circ\text{C}$  or  $-40^\circ\text{C}$  to  $+85^\circ\text{C}$

Num.	Symbol	Parameter	16.78 MHz		20.97 MHz		Unit
			Min	Max	Min	Max	
1	$V_{IH}$	Input high voltage	0.7 ( $V_{DD}$ )	$V_{DD} + 0.3$	0.7 ( $V_{DD}$ )	$V_{DD} + 0.3$	V
2	$V_{IL}$	Input low voltage	$V_{SS} - 0.3$	0.2 ( $V_{DD}$ )	$V_{SS} - 0.3$	0.2 ( $V_{DD}$ )	V
3	$V_{HYS}$	Input hysteresis (Note 1)	0.5	—	0.5		V
4	$I_{in}$	Input leakage current (Note 2) $V_{in} = V_{DD}$ or $V_{SS}$ Input-only pins	-2.5	2.5	-2.5	2.5	$\mu\text{A}$
5	$I_{OZ}$	High impedance (off-state) leakage current (Note 2) $V_{in} = V_{DD}$ or $V_{SS}$ All input/output and output pins	-2.5	2.5	-2.5	2.5	$\mu\text{A}$

Table 4 - DC characteristics (continued)

$V_{DD}$  and  $V_{DDSYN} = 5.0 V_{dc} \pm 10\%$  for 16.78 MHz and  $5.0 V_{dc} \pm 5\%$  for 20.97 MHz ;  
 $V_{SS} = 0 V_{dc}$  ;  $T_c = -55^\circ\text{C}$  to  $+125^\circ\text{C}$  or  $-40^\circ\text{C}$  to  $+85^\circ\text{C}$

Num.	Symbol	Parameter	16.78 MHz		20.97 MHz		Unit
			Min	Max	Min	Max	
6	$V_{OH}$	CMOS output high voltage (Notes 2 and 3) $I_{OH} = -10.0 \mu\text{A}$ Group 1, 2, 4 input/output and all output pins	$V_{DD} - 0.2$	—	$V_{DD} - 0.2$	—	V
7	$V_{OL}$	CMOS output low voltage (Note 2) $I_{OL} = 10.0 \mu\text{A}$ Group 1, 2, 4 input/output and all output pins	—	0.2	—	0.2	V
8	$V_{OH}$	Output high voltage (Notes 2 and 3) $I_{OH} = -0.8 \text{ mA}$ Group 1, 2, 4 input/output and all output pins	$V_{DD} - 0.8$	—	$V_{DD} - 0.8$	—	V
9	$V_{OL}$	Output low voltage (Note 2) $I_{OL} = 1.6 \text{ mA}$ Group 1 I/O pins, CLKOUT, FREEZE/QUOT, IPIPE	—	0.4	—	0.4	V
		$I_{OL} = 5.3 \text{ mA}$ Group 2, 4 I/O pins, CSBOOT, BG/CS	—	0.4	—	0.4	V
		$I_{OL} = 12 \text{ mA}$ Group 3	—	0.4	—	0.4	V
10	$V_{IH\text{TSC}}$	Three state control input high voltage	$1.6 (V_{DD})$	9.1	$1.6 (V_{DD})$	9.1	V
11	$I_{MSP}$	Data bus mode select pull-up current (Note 5) $V_{in} = V_{IL}$ DATA [15:0]	—	-120	—	-120	$\mu\text{A}$
		$V_{in} = V_{IH}$ DATA [15:0]	-15	—	-15	—	$\mu\text{A}$
12	$I_{DD}$ $I_{DD}$ $S_{IDD}$ $S_{IDD}$	$V_{DD}$ supply current (Note 5) RUN (Note 6)	—	124	—	140	mA
		RUN, TPU emulation mode	—	134	—	150	mA
		LPSTOP, 32.768 kHz crystal, VCO off (STSIM = 0)	—	350	—	350	$\mu\text{A}$
		LPSTOP (external clock input frequency = maximum $f_{sys}$ )	—	5	—	5	mA
13	$V_{DDSYN}$	Clock synthesizer operating voltage	4.5	5.5	4.75	5.25	V
14	$I_{DDSYN}$ $I_{DDSYN}$ $S_{IDDSYN}$ $I_{DDSYN}$	$V_{DDSYN}$ supply current (Note 5) 32.768 kHz crystal, VCO on, maximum $f_{sys}$	—	1	—	2	mA
		External clock, maximum $f_{sys}$	—	5	—	6	mA
		LPSTOP, 32.768 kHz crystal, VCO off (STSIM = 0)	—	150	—	150	$\mu\text{A}$
		32.768 kHz crystal, $V_{DD}$ powered down	—	100	—	100	$\mu\text{A}$
15	$V_{SB}$	RAM standby voltage (Note 7) Specified $V_{DD}$ applied	0.0	5.5	0.0	5.25	V
		$V_{DD} = V_{SS}$	3.0	5.5	3.0	5.25	V
16	$I_{SB}$	RAM standby current (Notes 5, 7 and 8) Normal RAM operation $V_{DD} > V_{SB} - 0.5 \text{ V}$	—	10	—	10	$\mu\text{A}$
		Transient condition $V_{SB} - 0.5 \text{ V} \geq V_{DD} \geq$ $V_{SS} + 0.5 \text{ V}$	—	3	—	3	mA
		Standby operation $V_{DD} < V_{SS} + 0.5 \text{ V}$	—	60	—	50	$\mu\text{A}$
17	$P_D$	Power dissipation (Note 9)	—	690	—	766	mW
18	$C_{in}$	Input capacitance (Notes 2 and 10) All input-only pins	—	10	—	10	pF
		All input/output pins	—	20	—	20	pF

**Table 4 - DC characteristics (continued)**

$V_{DD}$  and  $V_{DDSYN} = 5.0 V_{DC} \pm 10\%$  for 16.78 MHz and  $5.0 V_{DC} \pm 5\%$  for 20.97 MHz ;  
 $V_{SS} = 0 V_{DC}$  ;  $T_C = -55^\circ\text{C}$  to  $+125^\circ\text{C}$  or  $-40^\circ\text{C}$  to  $+85^\circ\text{C}$

Num.	Symbol	Parameter	16.78 MHz		20.97 MHz		Unit
			Min	Max	Min	Max	
19	$C_L$	Load capacitance (Note 2)	—	90	—	90	pF
		Group 1 I/O pins CLKOUT, FREEZE/QUOT, IPIPE	—	100	—	100	pF
		Group 2 I/O pins and CSBOOT, BG/CS	—	130	—	130	pF
		Group 3 I/O pins	—	200	—	200	pF
		Group 4 I/O pins	—	200	—	200	pF

**Note 1 :** Applies to :

Port E [7:4] — SIZ [1:0], AS, DS.  
 Port F [7:0] — IRQ [7:1], MODCLK.  
 Port QS [7:0] — TXD, PCS [3:1], PCS0/SS, SCK, MOSI, MISO.  
 TPUCH [15:0], T2CLK.  
 BKPT/DSCLK, IFETCH, RESET, RXD, TSTME/TSC.  
 EXTAL (when PLL enabled).

**Note 2 :** Input-only pins : EXTAL, TSTME/TSC, BKPT, T2CLK, RXD.

Output-only pins : CSBOOT, BG/CS, CLKOUT, FREEZE/QUOT, IPIPE.

Input/output pins :

Group 1 : DATA [15:0], IFETCH, TPUCH [15:0].  
 Group 2 : Port C [6:0] — ADDR [22:19]/CS [9:6], FC [2:0]/CS [5:3].  
 Port E [7:0] — SIZ [1:0], AS, DS, AVEC, RMC, DSACK [1:0].  
 Port F [8:0] — IRQ [7:1], MODCLK.  
 Port QS [7:3] — TXD, PCS [3:1], PCS0/SS  
 ADDR23/CS10/ECLK, ADDR [18:0], R/W, BERR, BR/CS0, BGACK/CS2.  
 Group 3 : HALT, RESET.  
 Group 4 : MISO, MOSI, SCK.

**Note 3 :** Does not apply to HALT and RESET because they are open drain pins. Does not apply to Port QS [7:0] (TXD, PCS [3:1], PCS0/SS, SCK, MOSI, MISO) in wired-OR mode.

**Note 4 :** Use of an active pulldown device is recommended.

**Note 5 :** Total operating current is the sum of the appropriate  $I_{DD}$ ,  $I_{DDSYN}$ , and  $I_{SB}$  values.  $I_{DD}$  values include supply currents for device modules powered by  $V_{DDE}$  and  $V_{DDI}$  pins.

**Note 6 :** Current measured with system clock frequency of 16.78 MHz, all modules active.

**Note 7 :** The RAM module will not switch into standby mode as long as  $V_{SB}$  does not exceed  $V_{DD}$  by more than 0.5 Volt. The RAM array cannot be accessed while the module is in standby mode.

**Note 8 :** When  $V_{DD}$  is transitioning during power-up or power down sequence, and  $V_{SB}$  is applied, current flows between the  $V_{STBY}$  and  $V_{DD}$  pins, which causes standby current to increase toward the maximum transient condition specification. System noise on the  $V_{DD}$  and  $V_{STBY}$  pins can contribute to this condition.

**Note 9 :** Power dissipation measured at specified system clock frequency, all modules active. Power dissipation can be calculated using the expression :

$$P_D = \text{Maximum } V_{DD} (I_{DD} + I_{DDSYN} + I_{SB})$$

$I_{DD}$  includes supply currents for all device modules powered by  $V_{DDE}$  and  $V_{DDI}$  pins.

**Note 10 :** This parameter is periodically sampled rather than 100 % tested.

**5.3 - Dynamic (Switching) characteristics**

The INTERVAL numbers refer to the timing diagrams..

**Table 5 - Clock control timing**

$V_{DD}$  and  $V_{DDSYN} = 5.0 V_{dc} \pm 10\%$  for 16.78 MHz and  $5.0 V_{dc} \pm 5\%$  for 20.97 MHz ;  
 $V_{SS} = 0 V_{dc}$  ;  $T_c = -55^\circ C$  to  $+125^\circ C$  or  $-40^\circ C$  to  $+85^\circ C$

Num.	Symbol	Parameter	16.78 MHz		20.97 MHz		Unit
			Min	Max	Min	Max	
1	$f_{ref}$	PLL reference frequency range	25	50	25	50	kHz
2	$f_{sys}$	System frequency (Note 1)	dc	16.78	dc	20.97	MHz
		On-chip PLL system frequency	0.131	16.78	0.131	20.97	MHz
		External clock operation	dc	16.78	dc	20.97	MHz
3	$t_{ipll}$	PLL lock time (Notes 2, 3, 4 and 5)	—	20	—	20	ms
4	$f_{VCO}$	VCO frequency (Note 6)	—	2 ( $f_{sys}$ max)	—	2 ( $f_{sys}$ max)	MHz
5	$f_{limp}$	Limp mode clock frequency	—	$f_{sys}$ max / 2	—	$f_{sys}$ max / 2	MHz
		SYNCR X bit = 0	—	$f_{sys}$ max	—	$f_{sys}$ max	MHz
6	$C_{stab}$	CLKOUT stability (Notes 2, 3, 4 and 7)	—0.5	0.5	—0.5	0.5	%
		Short term (5 $\mu s$ interval)	—0.05	0.05	—0.05	0.05	%
		Long term (500 $\mu s$ interval)					

**Note 1 :** All internal registers retain data at 0 Hz.

**Note 2 :** This parameter is periodically sampled rather than 100 % tested.

**Note 3 :** Assumes that a low-leakage external filter network is used to condition clock synthesizer input voltage. Total external resistance from the XFC pin due to external leakage must be greater than 15 M $\Omega$  to guarantee this specification. Filter network geometry can vary depending upon operating environment.

**Note 4 :** Proper layout procedures must be followed to achieve specifications.

**Note 5 :** Assumes that stable  $V_{DDSYN}$  is applied, and that the crystal oscillator is stable. Lock time is measured from the time  $V_{DD}$  and  $V_{DDSYN}$  are valid until RESET is released. This specification also applies to the period required for PLL lock after changing the W and Y frequency control bits in the synthesizer control register (SYNCR) while the PLL is running, and to the period required for the clock to lock after LPSTOP.

**Note 6 :** Internal VCO frequency ( $f_{VCO}$ ) is determined by SYNCR W and Y bit values. The SYNCR X bit controls a divide-by-two circuit that is not in the synthesizer feedback loop. When X = 0, the divider is enabled, and  $f_{sys} = f_{VCO} : 4$ . When X = 1, the divider is disabled, and  $f_{sys} = f_{VCO} : 2$ . X must equal one when operating at maximum specified  $f_{sys}$ .

**Note 7 :** Stability is the average deviation from the programmed frequency measured over the specified interval at maximum  $f_{sys}$ . Measurements are made with the device powered by filtered supplies and clocked by a stable external clock signal. Noise injected into the PLL circuitry via  $V_{DDSYN}$  and  $V_{SS}$  and variation in crystal oscillator frequency increase the  $C_{stab}$  percentage for a given interval. When clock stability is a critical constraint on control system operation, this parameter should be measured during functional testing of the final system.

**Table 6 - AC timing**

$V_{DD}$  and  $V_{DDSYN} = 5.0 V_{dc} \pm 10\%$  for 16.78 MHz and  $5.0 V_{dc} \pm 5\%$  for 20.97 MHz ;  
 $V_{SS} = 0 V_{dc}$  ;  $T_c = -55^\circ C$  to  $+125^\circ C$  or  $-40^\circ C$  to  $+85^\circ C$  (Note 1)

Num.	Symbol	Parameter	16.78 MHz		20.97 MHz		Unit
			Min	Max	Min	Max	
F1	f	Frequency of operation (32.768 kHz crystal) (Note 2)	0.13	16.78	0.13	20.97	MHz
1	$t_{cyc}$	Clock period	59.6	—	47.7	—	ns
1A	$t_{Ecyc}$	ECLK period	476	—	381	—	ns
1B	$t_{Xcyc}$	External clock input period (Note 3)	59.6	—	47.7	—	ns
2, 3	$t_{CW}$	Clock pulse width	24	—	18.8	—	ns
2A, 3A	$t_{ECW}$	ECLK pulse width	236	—	183	—	ns
2B, 3B	$t_{XCHL}$	External clock input high/low time (Note 3)	29.8	—	23.8	—	ns
4, 5	$t_{Crf}$	Clock rise and fall time	—	5	—	5	ns
4A, 5A	$t_{rf}$	Rise and fall time - All outputs except CLKOUT	—	8	—	8	ns
4B, 5B	$t_{XCrf}$	External clock rise and fall time (Note 4)	—	5	—	5	ns

Table 6 - AC timing (continued)

$V_{DD}$  and  $V_{DDSYN} = 5.0 V_{dc} \pm 10\%$  for 16.78 MHz and  $5.0 V_{dc} \pm 5\%$  for 20.97 MHz ;  
 $V_{SS} = 0 V_{dc}$  ;  $T_c = -55^\circ\text{C}$  to  $+125^\circ\text{C}$  or  $-40^\circ\text{C}$  to  $+85^\circ\text{C}$  (Note 1)

Num.	Symbol	Parameter	16.78 MHz		20.97 MHz		Unit
			Min	Max	Min	Max	
6	tCHAV	Clock high to address, FC, SIZE, $\overline{RMC}$ valid	0	29	0	23	ns
7	tCHAZx	Clock high to address, Data, FC, SIZE, $\overline{RMC}$ high impedance	0	59	0	47	ns
8	tCHAZn	Clock high to address, FC, SIZE, RMC invalid	0	—	0	—	ns
9	tCLSA	Clock low to $\overline{AS}$ , $\overline{DS}$ , $\overline{CS}$ asserted	2	25	0	23	ns
9A	tSTSA	$\overline{AS}$ to $\overline{DS}$ or $\overline{CS}$ asserted (read) (Note 5)	-15	15	-10	10	ns
9C	tCLIA	Clock low to $\overline{IFETCH}$ , $\overline{IPIPE}$ asserted	2	22	2	22	ns
11	tAVSA	Address, FC, SIZE, $\overline{RMC}$ valid to $\overline{AS}$ , $\overline{CS}$ (and $\overline{DS}$ read) asserted	15	—	10	—	ns
12	tCLSN	Clock low to $\overline{AS}$ , $\overline{DS}$ , $\overline{CS}$ negated	2	29	2	23	ns
12A	tCLIN	Clock low to $\overline{IFETCH}$ , $\overline{IPIPE}$ negated	2	22	2	22	ns
13	tSNAI	$\overline{AS}$ , $\overline{DS}$ , $\overline{CS}$ negated to address, FC, SIZE invalid (address hold)	15	—	10	—	ns
14	tSWA	$\overline{AS}$ , $\overline{CS}$ (and $\overline{DS}$ read) width asserted	100	—	80	—	ns
14A	tSWAW	$\overline{DS}$ , $\overline{CS}$ width asserted (write)	45	—	36	—	ns
14B	tSWDW	$\overline{AS}$ , $\overline{CS}$ (and $\overline{DS}$ read) width asserted (fast write cycle)	40	—	32	—	ns
15	tSN	$\overline{AS}$ , $\overline{DS}$ , $\overline{CS}$ width negated (Note 6)	40	—	32	—	ns
16	tCHSZ	Clock high to $\overline{AS}$ , $\overline{DS}$ , $\overline{R/W}$ high impedance	—	59	—	47	ns
17	tSNRN	$\overline{AS}$ , $\overline{DS}$ , $\overline{CS}$ negated to $\overline{R/W}$ negated	15	—	10	—	ns
18	tCHRH	Clock high to $\overline{R/W}$ high	0	29	0	23	ns
20	tCHRL	Clock high to $\overline{R/W}$ low	0	29	0	23	ns
21	tRAAA	$\overline{R/W}$ asserted to $\overline{AS}$ , $\overline{CS}$ asserted	15	—	10	—	ns
22	tRASA	$\overline{R/W}$ low to $\overline{DS}$ , $\overline{CS}$ asserted (write)	70	—	54	—	ns
23	tCHDO	Clock high to data out valid	—	29	—	23	ns
24	tDVASN	Data out valid to negating edge of $\overline{AS}$ , $\overline{CS}$ (fast write cycle)	15	—	10	—	ns
25	tSNDI	$\overline{DS}$ , $\overline{CS}$ negated to data out invalid (data out hold)	15	—	10	—	ns
26	tDVSA	Data out valid to $\overline{DS}$ , $\overline{CS}$ asserted (write)	15	—	10	—	ns
27	tDIDL	Data in valid to clock low (data setup)	5	—	5	—	ns
27A	tBELCL	Late $\overline{BERR}$ , $\overline{HALT}$ asserted to clock low (setup time)	20	—	15	—	ns
28	tSNDN	$\overline{AS}$ , $\overline{DS}$ negated to $\overline{DSACK}$ [1:0], $\overline{BERR}$ , $\overline{HALT}$ , $\overline{AVEC}$ negated	0	80	0	60	ns
29	tSNDI	$\overline{DS}$ , $\overline{CS}$ negated to data in invalid (data in hold) (Note 7)	0	—	0	—	ns
29A	tSHDI	$\overline{DS}$ , $\overline{CS}$ negated to data in high impedance (Notes 7 and 8)	—	55	—	48	ns

Table 6 - AC timing (continued)

$V_{DD}$  and  $V_{DDSYN} = 5.0 V_{DC} \pm 10\%$  for 16.78 MHz and  $5.0 V_{DC} \pm 5\%$  for 20.97 MHz ;  
 $V_{SS} = 0 V_{DC}$  ;  $T_c = -55^\circ\text{C}$  to  $+125^\circ\text{C}$  or  $-40^\circ\text{C}$  to  $+85^\circ\text{C}$  (Note 1)

Num.	Symbol	Parameter	16.78 MHz		20.97 MHz		Unit
			Min	Max	Min	Max	
30	t <sub>CLDI</sub>	CLKOUT low to data in invalid (fast cycle hold) (Note 7)	15	—	10	—	ns
30A	t <sub>CLDH</sub>	CLKOUT low to data in high impedance (Note 7)	—	90	—	72	ns
31	t <sub>DADI</sub>	$\overline{\text{DSACK}} [1:0]$ asserted to data in valid (Note 9)	—	50	—	46	ns
33	t <sub>CLBAN</sub>	Clock low to $\overline{\text{BG}}$ asserted/negated	—	29	—	23	ns
35	t <sub>BRAGA</sub>	$\overline{\text{BR}}$ asserted to $\overline{\text{BG}}$ asserted ( $\overline{\text{RMC}}$ not asserted) (Note 10)	1	—	1	—	t <sub>cyc</sub>
37	t <sub>GAGN</sub>	$\overline{\text{BGACK}}$ asserted to $\overline{\text{BG}}$ negated	1	2	1	2	t <sub>cyc</sub>
39	t <sub>GH</sub>	$\overline{\text{BG}}$ width negated	2	—	2	—	t <sub>cyc</sub>
39A	t <sub>GA</sub>	$\overline{\text{BG}}$ width asserted	1	—	1	—	t <sub>cyc</sub>
46	t <sub>RWA</sub>	$\overline{\text{R/W}}$ width asserted (write or read)	150	—	115	—	ns
46A	t <sub>RWAS</sub>	$\overline{\text{R/W}}$ width asserted (fast write or read cycle)	90	—	70	—	ns
47A	t <sub>AI<sub>ST</sub></sub>	Asynchronous input setup time $\overline{\text{BR}}$ , $\overline{\text{BGACK}}$ , $\overline{\text{DSACK}} [1:0]$ , $\overline{\text{BERR}}$ , $\overline{\text{AVEC}}$ , $\overline{\text{HALT}}$	5	—	5	—	ns
47B	t <sub>AI<sub>HT</sub></sub>	Asynchronous input hold time	15	—	12	—	ns
48	t <sub>DABA</sub>	$\overline{\text{DSACK}} [1:0]$ asserted to $\overline{\text{BERR}}$ , $\overline{\text{HALT}}$ asserted (Note 11)	—	30	—	30	ns
53	t <sub>DOCH</sub>	Data out hold from clock high	0	—	0	—	ns
54	t <sub>CHDH</sub>	Clock high to data out high impedance	—	28	—	23	ns
55	t <sub>RADC</sub>	$\overline{\text{R/W}}$ asserted to data bus impedance change	40	—	32	—	ns
56	t <sub>HRPW</sub>	$\overline{\text{RESET}}$ pulse width (reset instruction)	512	—	512	—	t <sub>cyc</sub>
57	t <sub>BNHN</sub>	$\overline{\text{BERR}}$ negated to $\overline{\text{HALT}}$ negated (rerun)	0	—	0	—	ns
70	t <sub>SCLDD</sub>	Clock low to data bus driven (show)	0	29	0	23	ns
71	t <sub>SCLDS</sub>	Data setup time to clock low (show)	15	—	10	—	ns
72	t <sub>SCLDH</sub>	Data hold from clock low (show)	10	—	10	—	ns
73	t <sub>BKST</sub>	$\overline{\text{BKPT}}$ input setup time	15	—	10	—	ns
74	t <sub>BKHT</sub>	$\overline{\text{BKPT}}$ input hold time	10	—	10	—	ns
75	t <sub>MSS</sub>	Mode select setup time	20	—	20	—	t <sub>cyc</sub>

Table 6 - AC timing (continued)

$V_{DD}$  and  $V_{DDSYN} = 5.0 V_{DC} \pm 10\%$  for 16.78 MHz and  $5.0 V_{DC} \pm 5\%$  for 20.97 MHz ;  
 $V_{SS} = 0 V_{DC}$  ;  $T_c = -55^\circ\text{C}$  to  $+125^\circ\text{C}$  or  $-40^\circ\text{C}$  to  $+85^\circ\text{C}$  (Note 1)

Num.	Symbol	Parameter	16.78 MHz		20.97 MHz		Unit
			Min	Max	Min	Max	
76	t <sub>MSH</sub>	Mode select hold time	0	—	0	—	ns
77	t <sub>RSTA</sub>	$\overline{\text{RESET}}$ assertion time (Note 12)	4	—	4	—	t <sub>cyc</sub>
78	t <sub>RSTR</sub>	$\overline{\text{RESET}}$ rise time (Notes 13 and 14)	—	10	—	10	t <sub>cyc</sub>

**Note 1 :** All AC timing is shown with respect to 20 %  $V_{DD}$  and 70 %  $V_{DD}$  levels unless otherwise noted.

**Note 2 :** Minimum system clock frequency is four times the crystal frequency, subject to specified limits.

**Note 3 :** When an external clock is used, minimum high and low times are based on a 50 % duty cycle. The minimum allowable t<sub>XCYC</sub> period is reduced when the duty cycle of the external clock signal varies. The relationship between external clock input duty cycle and minimum t<sub>XCYC</sub> is expressed :

$$\text{Minimum } t_{XCYC} \text{ period} = \text{minimum } t_{XCHL} / (50\% - \text{external clock input duty cycle tolerance})$$

**Note 4 :** Parameters for an external clock signal applied while the internal PLL is disabled (MODCLK pin held low during reset). Does not pertain to an external VCO reference applied while the PLL is enabled (MODCLK pin held high during reset). When the PLL is enabled, the clock synthesizer detects successive transitions of the reference signal. If transitions occur within the correct clock period, rise/fall times and duty cycle are of critical.

**Note 5 :** Specification 9A is the worst-case skew between  $\overline{\text{AS}}$  and  $\overline{\text{DS}}$  or  $\overline{\text{CS}}$ . The amount of skew depends on the relative loading of these signals. When loads are kept within specified limits, skew will not cause AS and DS to fall outside the limits shown in specification 9.

**Note 6 :** If multiple chip selects are used,  $\overline{\text{CS}}$  width negated (specification 15) applies to the time from the negation of a heavily loaded chip select to the assertion of a lightly loaded chip select. The  $\overline{\text{CS}}$  width negated specification between multiple chip selects does not apply to chip selects being used for synchronous ECLK cycles.

**Note 7 :** Hold times are specified with respect to  $\overline{\text{DS}}$  or  $\overline{\text{CS}}$  on asynchronous reads and with respect to CLKOUT on fast cycle reads. The user is free to use either hold time.

**Note 8 :** Maximum value is equal to (t<sub>cyc</sub> / 2) + 25 ns.

**Note 9 :** If the asynchronous setup time (specification 47A) requirements are satisfied, the  $\overline{\text{DSACK}} [1:0]$  low to data setup time (specification 31) and  $\overline{\text{DSACK}} [1:0]$  low to  $\overline{\text{BERR}}$  low setup time (specification 48) can be ignored. The data must only satisfy the data-in to clock low setup time (specification 27) for the following clock cycle.  $\overline{\text{BERR}}$  must satisfy only the late  $\overline{\text{BERR}}$  low to clock low setup time (specification 27A) for the following clock cycle.

**Note 10 :** To ensure coherency during every operand transfer,  $\overline{\text{BG}}$  will not be asserted in response to  $\overline{\text{BR}}$  until after all cycles of the current operand transfer are complete and  $\overline{\text{RMC}}$  is negated.

**Note 11 :** In the absence of  $\overline{\text{DSACK}} [1:0]$ ,  $\overline{\text{BERR}}$  is an asynchronous input using the asynchronous setup time (specification 47A).

**Note 12 :** After external  $\overline{\text{RESET}}$  negation is detected, a short transition period (approximately 2 t<sub>cyc</sub>) elapses, then the SIM drives  $\overline{\text{RESET}}$  low for 512 t<sub>cyc</sub>.

**Note 13 :** External assertion of the  $\overline{\text{RESET}}$  input can overlap internally-generated resets. To insure that an external reset is recognized in all cases,  $\overline{\text{RESET}}$  must be asserted for at least 590 CLKOUT cycles.

**Note 14 :** External logic must pull  $\overline{\text{RESET}}$  high during this period in order for normal MCU operation to begin.

**Note 15 :** Address access time = (2.5 + WS) t<sub>cyc</sub> - t<sub>CHAV</sub> - t<sub>DICL</sub>.

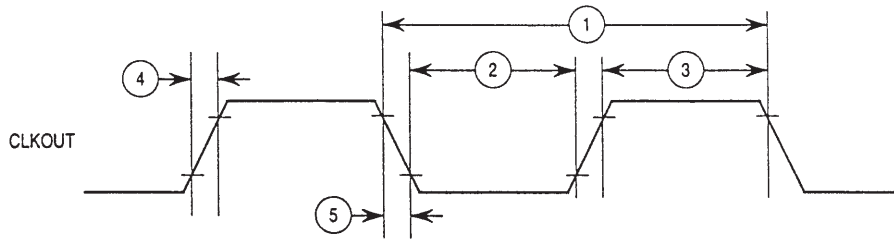
Chip select access time = (2 + WS) t<sub>cyc</sub> - t<sub>CLSA</sub> - t<sub>DICL</sub>.

Where : WS = number of wait states. When fast termination is used (2 clock bus) WS = -1.

5.4 - Test conditions specific to the device

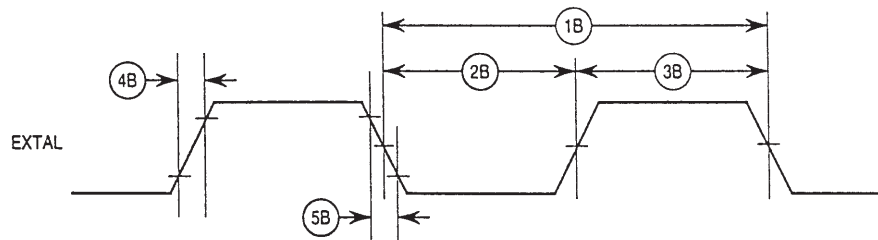
5.4.1 - Time definitions

The times specified in Table 6 as dynamic characteristics are defined in Figures 3 to 14 below, by a reference number given the column «NUM» of the tables together with the relevant figure number.



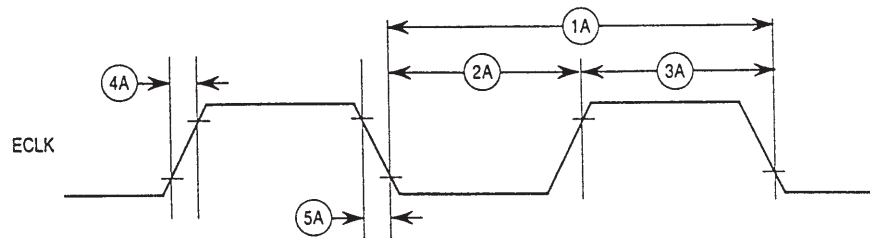
Note : Timing shown with respect to 20 % and 70 %  $V_{DD}$ .

Figure 3 : CLKOUT output timing diagram.



Note : Timing shown with respect to 20 % and 70 %  $V_{DD}$ . Pulse width shown with respect to 50 %  $V_{DD}$ .

Figure 4 : External clock input timing diagram.



Note : Timing shown with respect to 20 % and 70 %  $V_{DD}$ .

Figure 5 : ECLK output timing diagram.

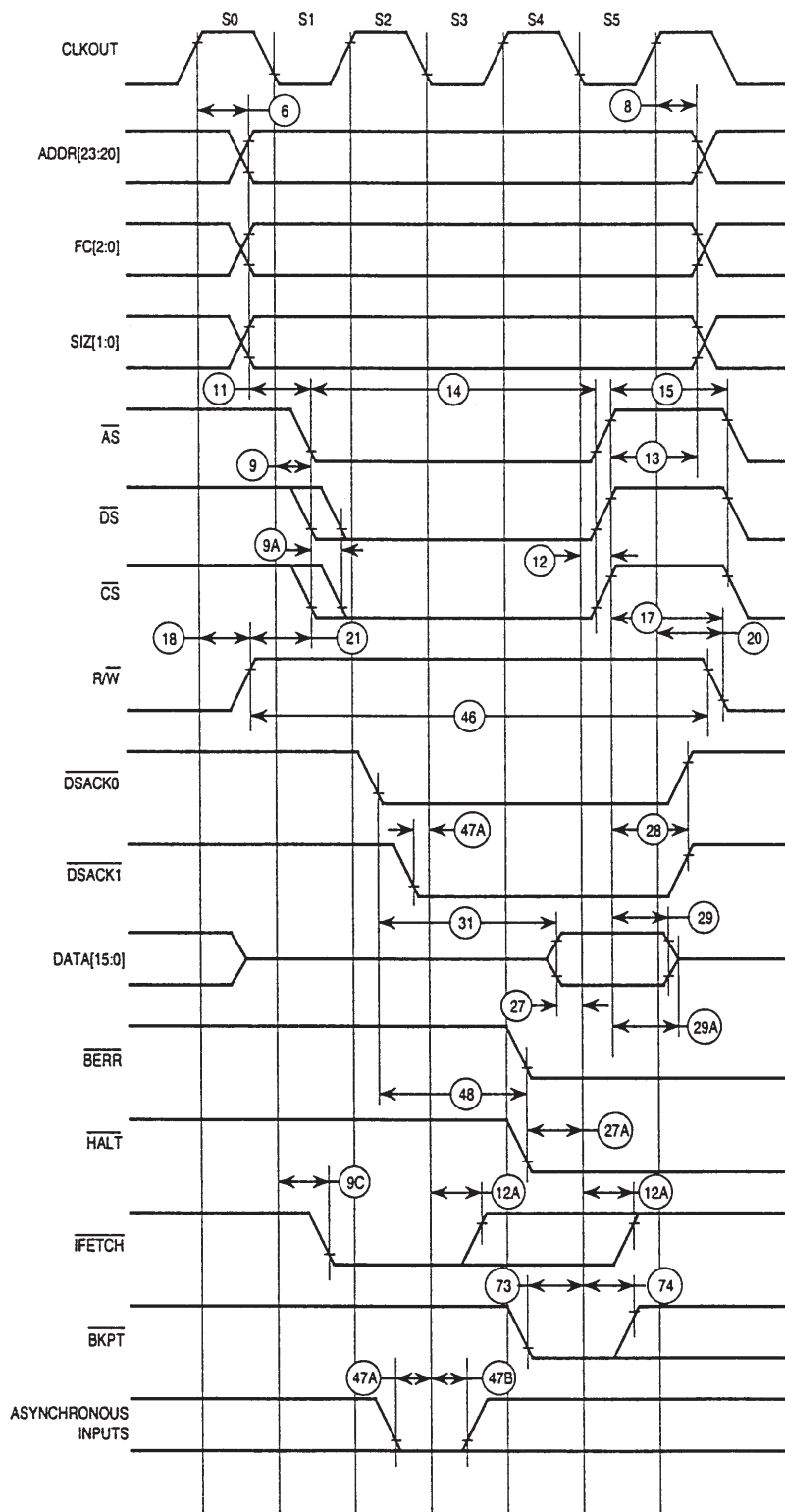


Figure 6 : Read cycle timing diagram.

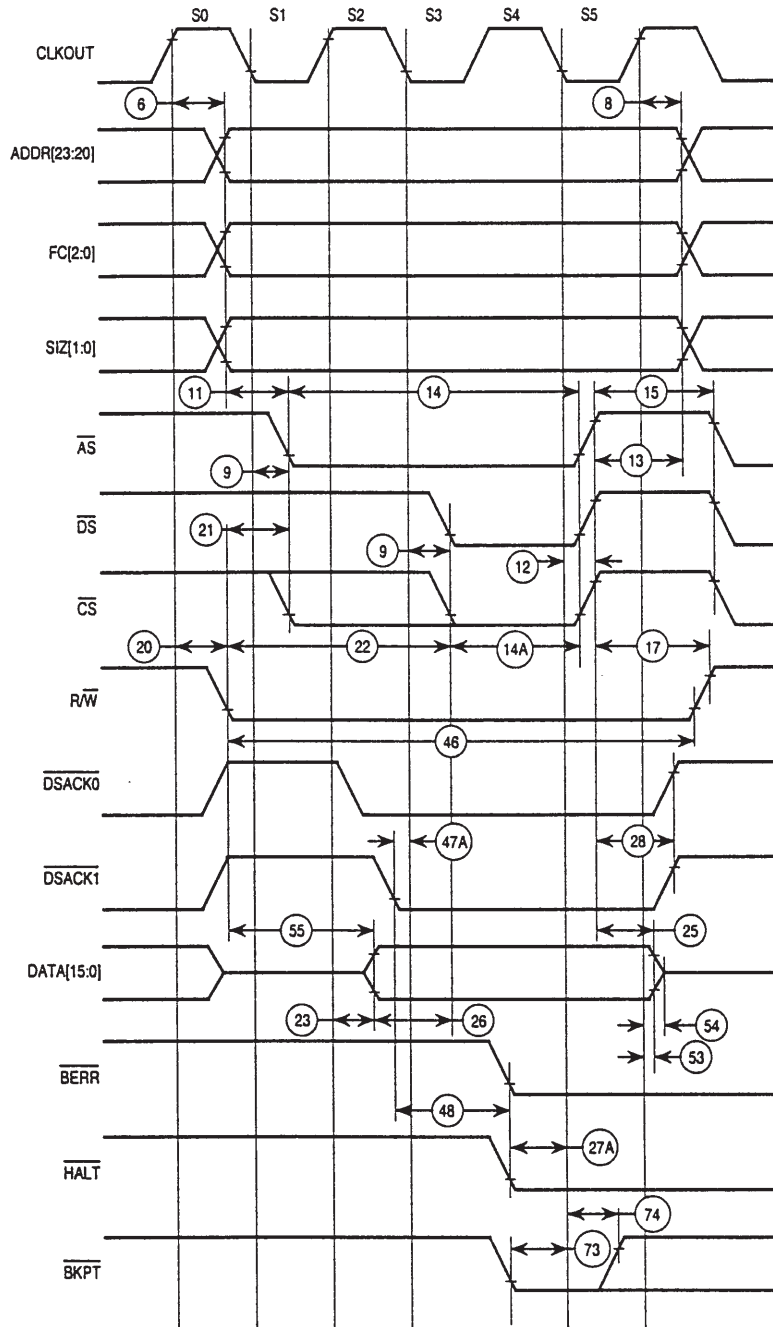


Figure 7 : Write cycle timing diagram.

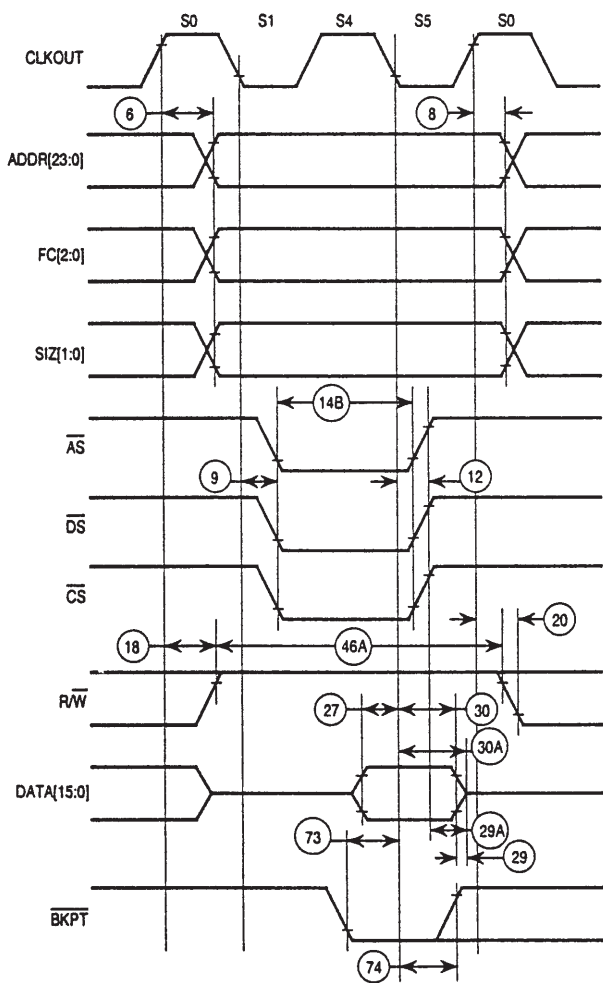


Figure 8 : Fast termination read cycle timing diagram.

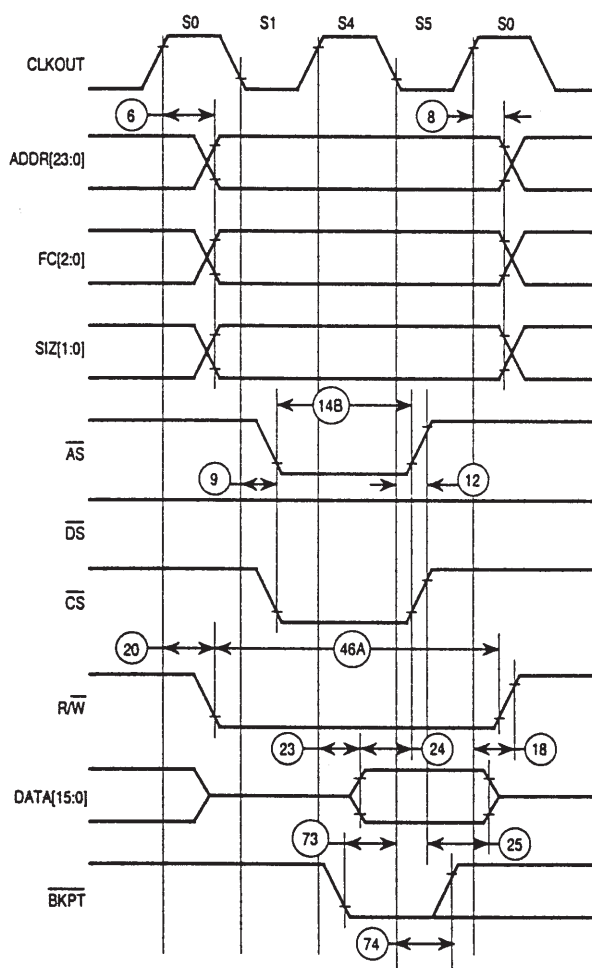


Figure 9 : Fast termination write cycle timing diagram.

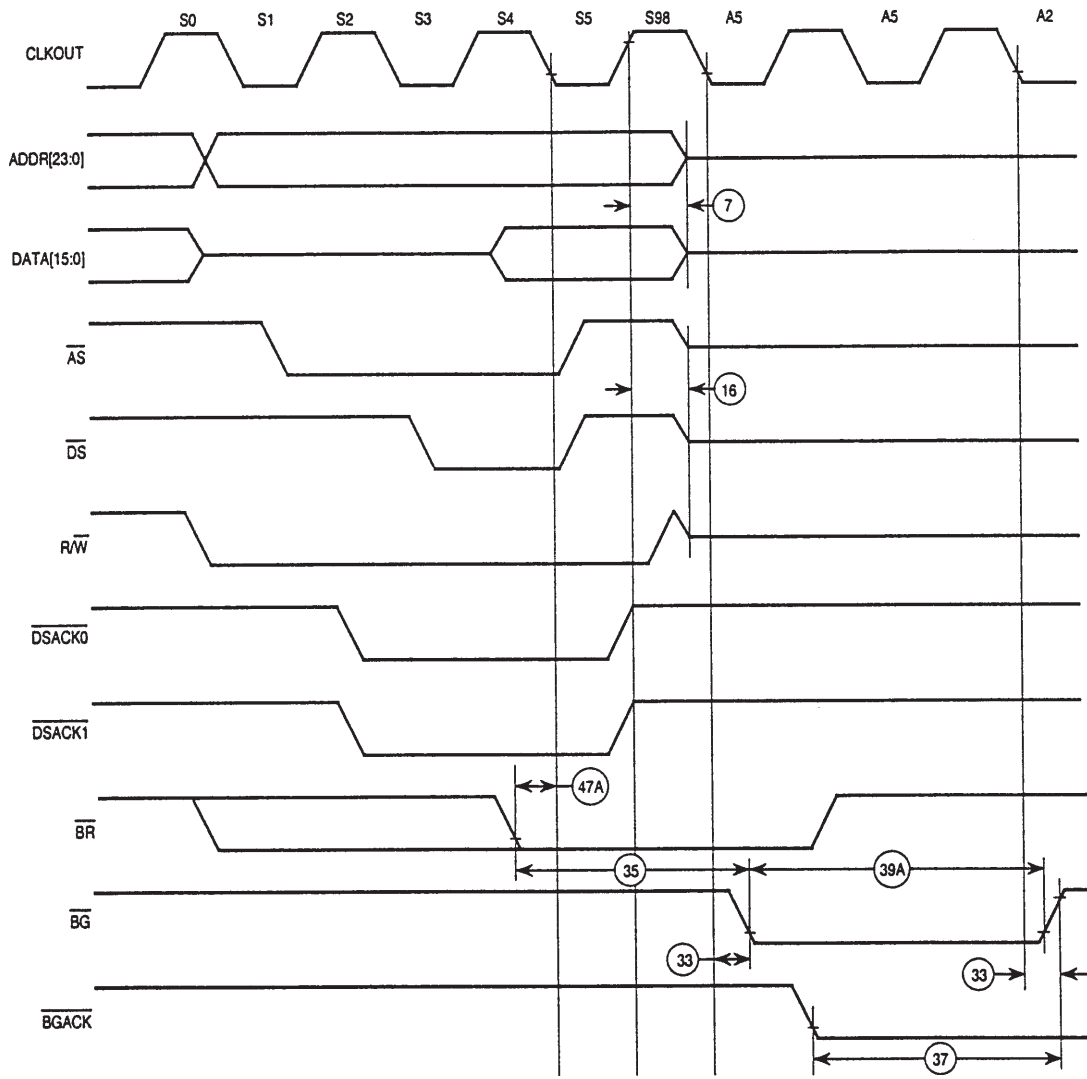


Figure 10 : Bus arbitration timing diagram - Active bus case.

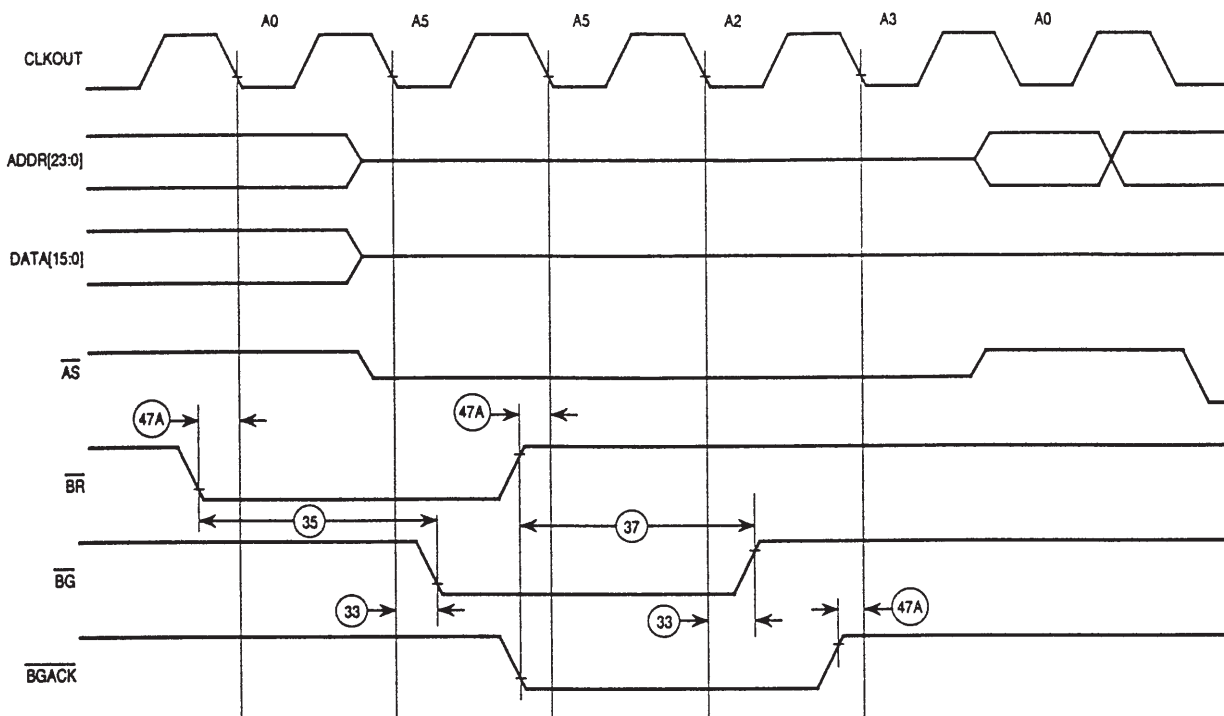
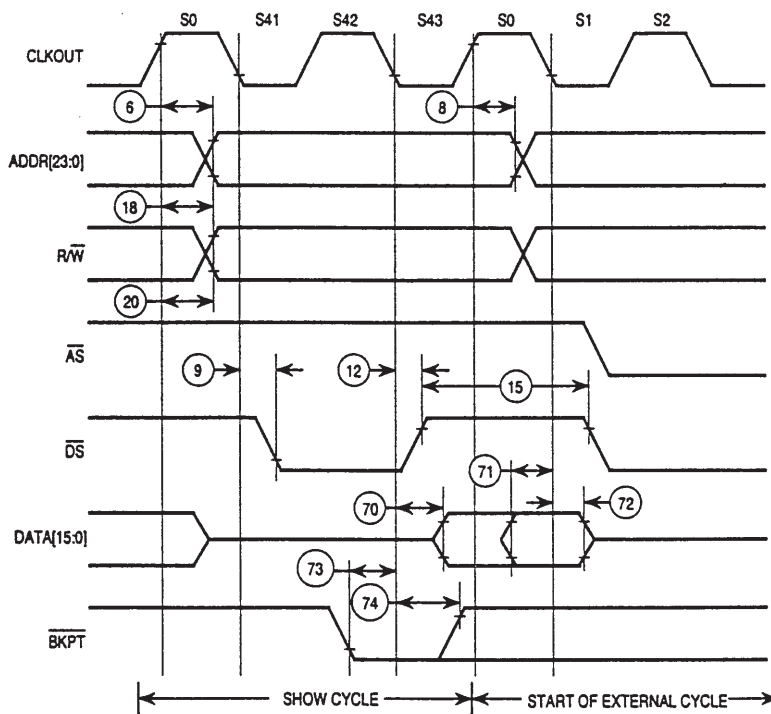
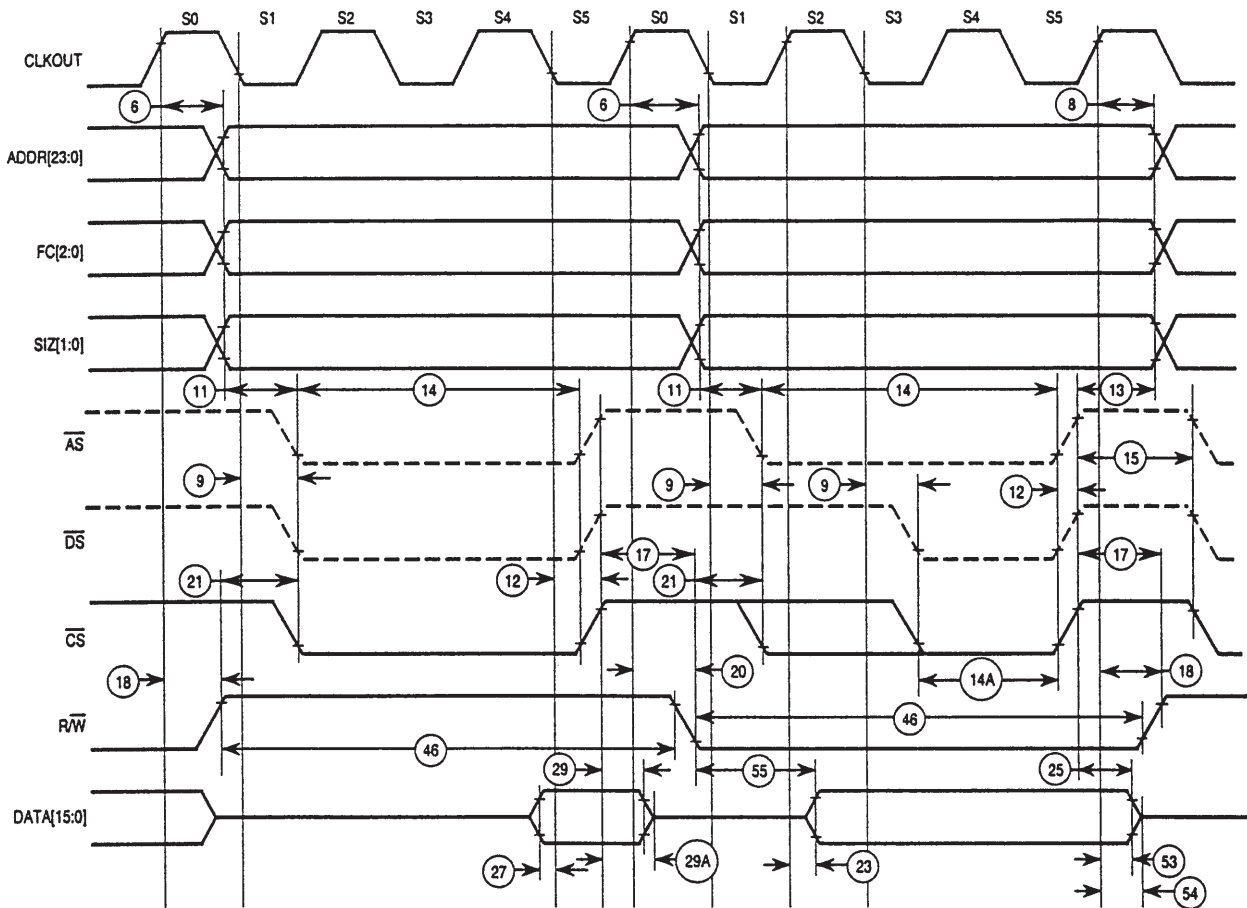


Figure 11 : Bus arbitration timing diagram - Idle bus case.



Note : Show cycles can stretch during S42 when bus accesses take longer than two cycles due to IMB module wait-state insertion.

Figure 12: Show cycle timing diagram.



Note : AS and DS timing shown for reference only.

Figure 13 : Chip select timing diagram.

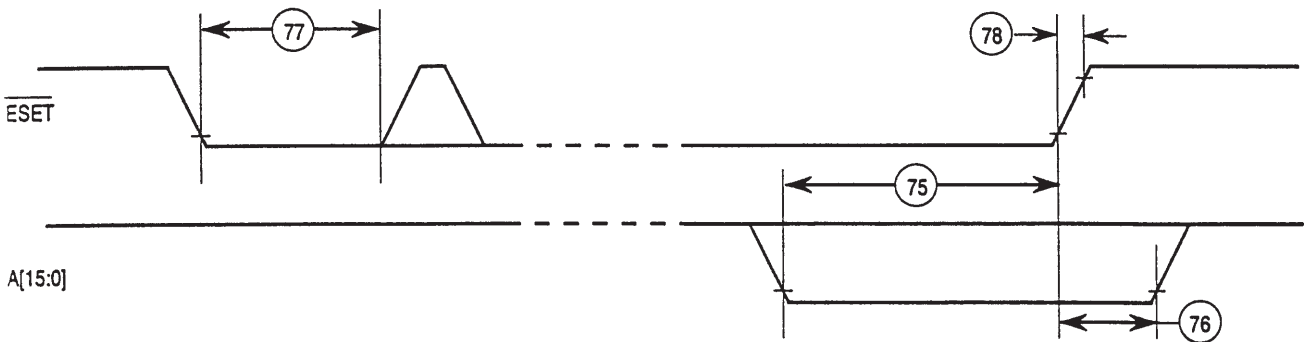
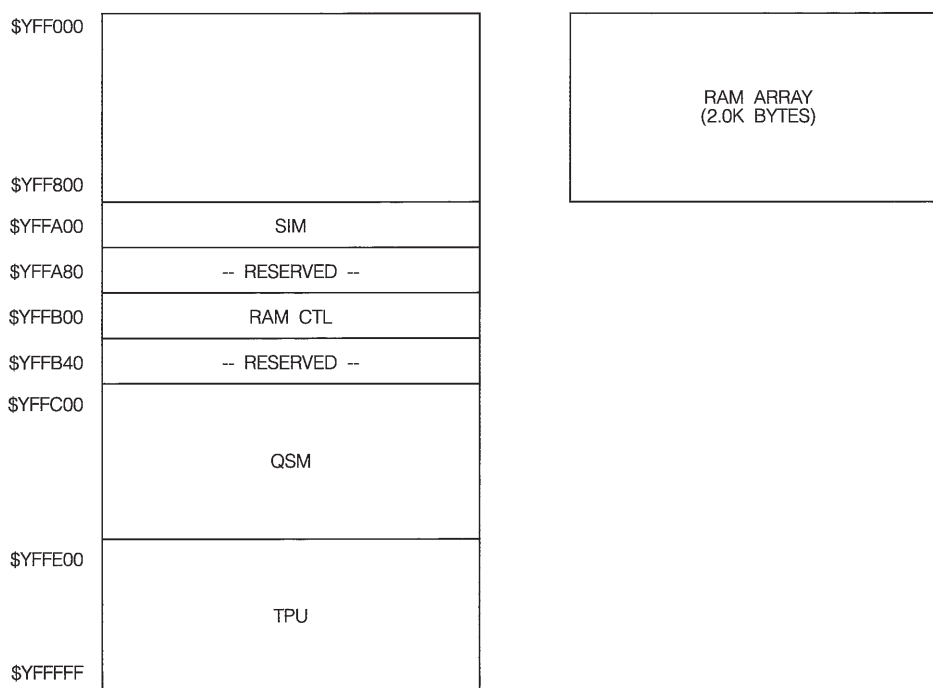


Figure 14 : Reset and mode select timing diagram.

## 6 - FUNCTIONAL DESCRIPTION

### 6.1 - Module memory map

Figure 15 illustrates the memory map of the TS 68332. The RAM array is positioned by the base address register in the RAM CTRL block. Reset forces the RAM array to be disabled. Unimplemented blocks are mapped externally.



Y - M111, where M is the modmap signal state on the IMB, which reflects the state of the modmap bit in the module configuration register of the system integration module (Y=\$7 or \$F).

Figure 15: Module memory map.

### 6.2 - CPU32 overview

This section is an overview of the CPU32. All capabilities and functions of this module are detailed fully in the CPU32 reference manual.

The CPU32, the instruction processing module of the 68300 family, is based on the industry-standard TS 68000 core processor with many features of the 68010 and TS 68020 as well as unique features suited for high-performance controller applications. The CPU32 is designed to provide a significant increase in performance over existing microcontroller CPUs to meet the demand for higher performance requirements for the 1990s, while maintaining source code and binary code compatibility with the 68000 family.

Ease of programming is an important consideration in using a microcontroller. An instruction format implementing a register-memory interaction philosophy predominates in the design, and all data resources are available to all operations requiring those resources.

#### 6.2.1 - Block diagram

Figure 16 shows a block diagram of the CPU32. The major clocks depicted operate in a highly independent fashion that maximizes concurrency of operation while managing the essential synchronization of instruction execution and bus operation. The bus controller loads instructions from the data bus into the decode unit.

The sequencer and control unit provide overall chip control, managing the internal buses, registers, and functions of the execution unit.

#### 6.2.2 - Architecture summary

The CPU32 architecture includes several important features that provide both power and versatility to the user. The CPU32 is source and object code compatible with the TS 68000 and 68010. All user-state programs can be executed unchanged. The major CPU32 features are as follows :

- 32-bit internal data path and arithmetic hardware,
- 32-bit internal address bus, 24-bit external address bus,
- eight 32-bit general-purpose data registers,
- seven 32-bit general-purpose address registers,
- separate user and supervisor stack pointers and address spaces,

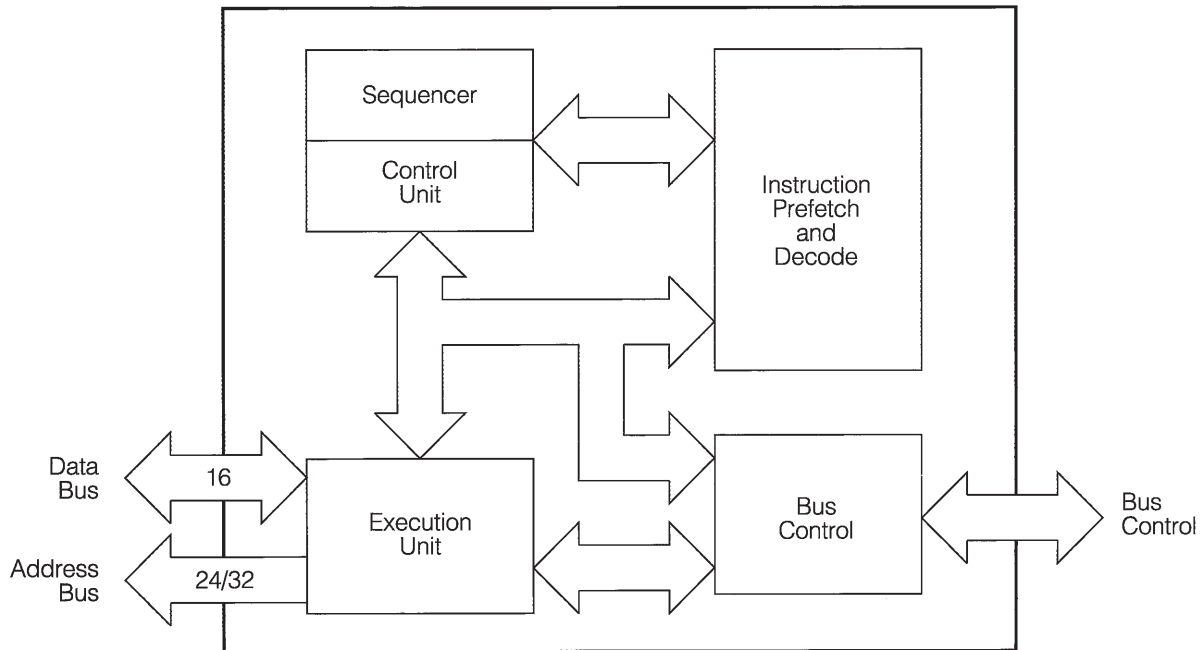


Figure 16 : CPU32 block diagram.

- separate program and data address spaces,
- full interrupt processing,
- fully upward object code compatible with 68000 family,
- virtual memory implementation,
- loop mode of instruction execution,
- fast multiply, divide, and shift instructions,
- fast bus interface with dynamic bus port sizing,
- improved exception handling for controller applications
- enhanced addressing modes :
  - scaled index,
  - address register indirect with base displacement and index,
  - expanded PC relative modes 32-bit branch displacements.
- instruction set enhancements :
  - high-precision multiply and divide,
  - trap on condition codes,
  - upper and lower bounds checking,
  - enhanced breakpoint instruction.
- trace on change of flow,
- table lookup and interpolate instruction,
- low power stop instruction,
- hardware breakpoint signal, background mode,
- 16.78 MHz and 20.97 MHz operating frequency at  $-55^{\circ}\text{C}$  to  $+125^{\circ}\text{C}$ ,
- fully static implementation.

### 6.2.3 - Programmer's model

The programming model of the CPU32 consists of two groups of registers : user model and supervisor model, which correspond to the user and supervisor privilege levels. Executing at the user privilege level, user programs can only use the registers of the user model. Executing at the supervisor level, system software uses the control registers of the supervisor level to perform supervisor functions.

The supervisor level has higher privileges than the user level. Not all instructions are permitted to execute in the lower privileged user level, but all instructions are available at the supervisor level. This scheme allows a separation of supervisor and user levels, and so the supervisor can protect system resources from uncontrolled access. The processor uses the privilege level indicated by the S bit in the status register to select either the user or supervisor privilege level and either the USP or SSP for stack operations.

The user programming model remains unchanged from previous 68000 family microprocessors. The supervisor programming model, which supplements the user programming model is used exclusively by the CPU32 system programmers who utilize the supervisor privilege level to implement sensitive operationg system functions. The supervisor programming model contains all the controls to access and enable the special features of the CPU32. All application software, written to run at the nonprivileged user level, migrates to the CPU32 from any 68000 platform without modification. The programming models are shown in Figures 17 and 18.

**6.2.4 - Registers**

Registers D7-D0 are used as data registers and readily support 8-bit (byte), 16-bit (word) and 32-bit (long word) operand lengths for all operations. Registers A6-A0 and the user and supervisor stack pointers are address registers that may be used as software stack pointers or base address registers. Register A7 is a register designation that applies to the user stack pointer in the user privilege level and to the supervisor stack pointer in the supervisor privilege level. In addition, the address registers may be used for word and long-word operations. All of the 16 general-purpose registers (D7-D0, A7-A0) may be used as index registers.

The PC contains the address of the next instruction to be executed by the CPU32.

The status register (SR) stores the processor status. It contains the condition codes that reflect the results of a previous operation and can be used for conditional instruction execution in a program.

The vector base register (VBR) contains the base address of the exception vector table in memory. The displacement of an exception vector is added to the value in this register to access the vector table.

Alternate function code registers (SFC and DFC) contain 3-bit function codes. Function codes can be considered extensions of the 24-bit linear address that optionally provide as many as eight 16 Mbyte address spaces. These address spaces are designated as either user or supervisor space and as either program or data space. There is a CPU space to allow the CPU to acquire specific control information not usually associated with read or write bus cycles. The function code signals FC2-FC0 select the appropriate address space.

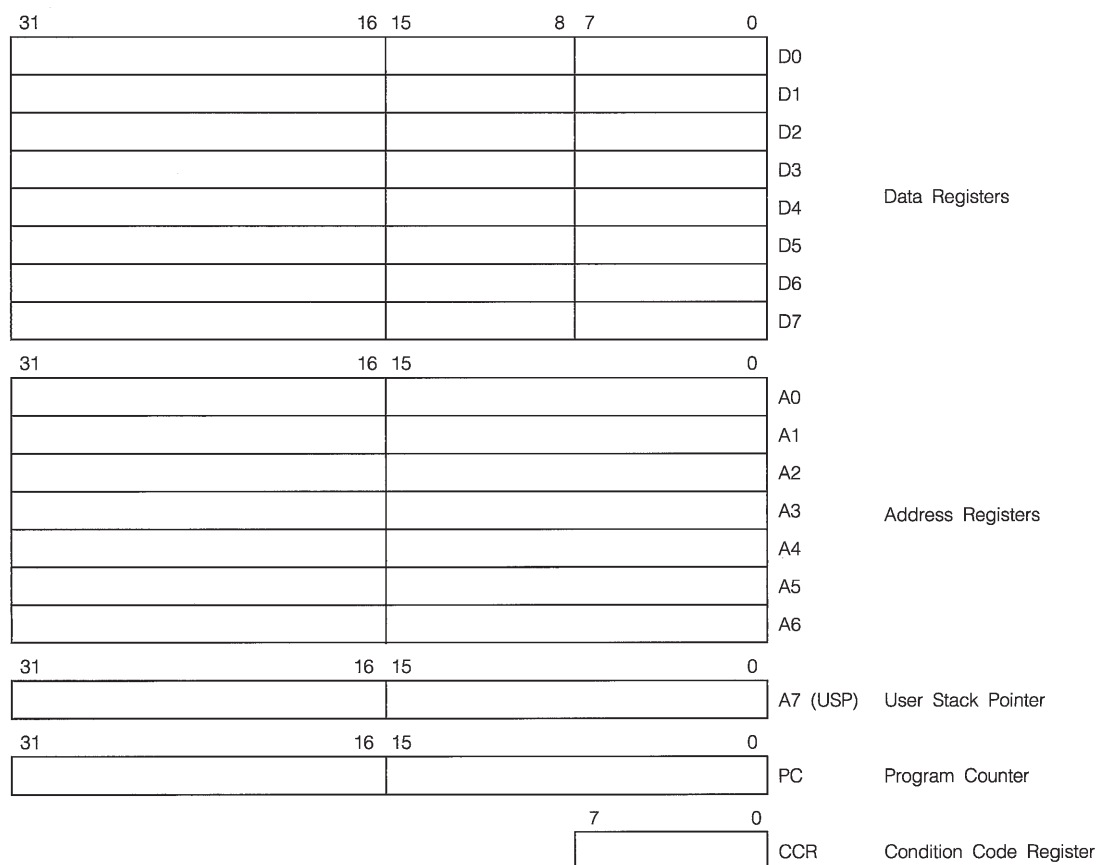


Figure 17 : User programming model.

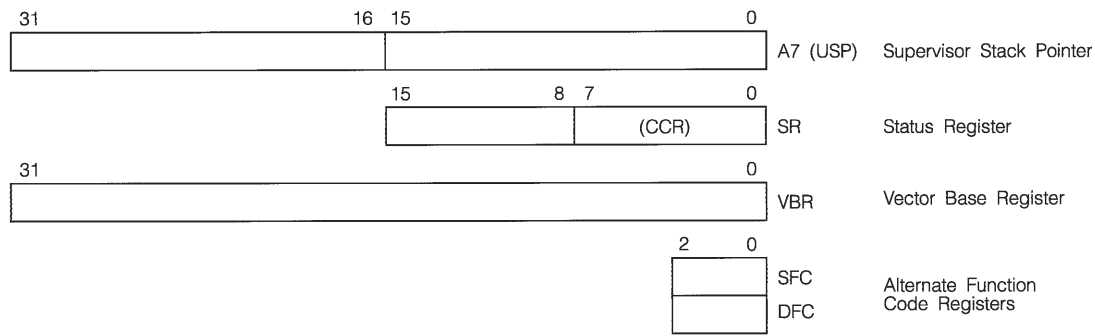


Figure 18 : Supervisor programming model supplement.

### 6.2.5 - Data types

Six basic data types are supported :

- bits,
- packaged binary-coded decimal digits,
- byte integers (8 bits),
- word integers (16 bits),
- long-word integers (32 bits),
- quad-word integers (64 bits).

#### 6.2.5.1 - Organisation in registers

The eight data registers can store data operands of 1, 8, 16, 32 and 64 bits and addresses of 16 or 32 bits. The seven address registers and the two stack pointers are used for address operands of 16 or 32 bits. The PC is 32 bits wide.

### 6.2.6 - System features

The CPU32 includes a number of features to aid system implementation. These include a privilege mechanism, separation of address spaces, multilevel priority interrupts, trap instructions, and a trace facility.

The privilege mechanism provides user and supervisor privilege states, privileged instructions, and external distinction of user and supervisor state references. The processor separates references between program and data space. This permits sharing of code segments that access separate data segments.

The CPU32 supports seven priority levels for 199 memory vectored interrupts. For each interrupt, the vector location can be provided externally or generated internally. The seventh level provides a non-maskable interrupt capability.

To simplify system development, instructions are provided to check internal processor conditions and allow software traps. The trace facility allows instruction-by-instruction tracing of program execution without alteration of the program or special hardware.

### 6.2.7 - Virtual memory

The full addressing range of the CPU32 on the TS 68332 is 16 Mbyte in each on eight address spaces. Even though most systems implement a smaller physical memory, the system can be made to appear to have a full 16 Mbyte of memory available to each user program by using virtual memory techniques.

### 6.2.8 - Loop mode instruction execution

The CPU32 has several features that provide efficient execution of program loops. One of these features is the DBcc looping primitive instruction. To increase the performance of the CPU32, a loop mode has been added to the processor. The loop mode is used by any single word instruction that does not change program flow. Loop mode is implemented in conjunction with the DBcc instruction. Once in loop mode, the processor performs only the data cycles associated with the instruction and suppresses all instruction fetches.

### 6.2.9 - Vector base register

The VBR contains the base address of the 1024-byte exception vector table, consisting of 256 exception vectors. Exception vectors contain memory addresses of routines that begin execution at the completion of exception processing, e.g. an interrupt routine.

### 6.2.10 - Processing states

The processor is always in one of four processing states : normal, exception, halted, or background. The normal processing state is that associated with instruction execution ; the bus is used to fetch instructions and operands and to store results. The exception processing state is associated with interrupts, trap instructions, tracing, and other exception conditions. The exception may be internally generated explicitly by an instruction or by an unusual condition arising during the execution of an instruction. Externally, exception processing can be forced by an interrupt, a bus error, or a reset. The halted processing state is an indication of catastrophic hardware failure. For example, if during the exception processing of a bus error another bus error occurs, the processor assumes that the system is unusable and halts. The background processing state is initiated by breakpoints, execution of special instructions, or a double bus fault. Background processing allows interactive debugging of the system via a simple serial interface.

### 6.2.11 - Addressing modes

Addressing in the CPU32 is register-oriented. Most instructions allow the results of the specified operation to be placed either in a register or directly in memory ; this flexibility eliminates the need for extra instructions to store register contents in memory.

The seven basic addressing modes are as follows :

- register direct,
- register indirect,
- register indirect with index,
- program counter indirect with displacement
- program counter indirect with index,
- absolute,
- immediate

Included in the register indirect addressing modes are the capabilities to post-increment, pre-decrement, and offset. The program counter relative mode also has index and offset capabilities. In addition to these addressing modes, many instructions implicitly specify the use of the status register, stack pointer, and/or program counter.

### 6.2.12 - Instructions

#### 6.2.12.1 - 68000 family compatibility

It is the philosophy of the 68000 family that all user-mode programs can execute unchanged on a more advanced processor, and supervisor-mode programs and exception handlers should require only minimal alteration.

The CPU32 can be thought of as an intermediate member of the 68000 family. Object code from an TS 68000 or 68010 may be executed on the CPU32, and many of the instruction and addressing mode extensions of the TS 68020 are also supported. Refer to the CPU 32 reference manual for a detailed comparison of the CPU32 and TS 68020 instruction set (see also Table 7).

#### 6.2.12.2 - New instructions

Two new instructions have been added to the TS 68000 instruction set for use in controller applications. They are low power stop (LPSTOP) and table lookup and interpolate (TBL).

#### Low Power Stop (LPSTOP)

In applications where power consumption is a consideration, the CPU32 forces the device into a low-power standby mode when immediate processing is not required. The low-power stop mode is entered by executing the LPSTOP instruction. The processor will remain in this mode until a user-specification (or higher) interrupt level or reset occurs.

#### Table lookup and interpolate (TBL)

To maximize throughput for real-time applications, reference data is often «pre-calculated» and stored in memory for quick access. The storage of each data point would require an inordinate amount of memory. The table instruction requires only a sample of data points stored in the array, reducing memory requirements. This single instruction allows intermediate values to be recovered by linear interpolation, thus significantly increasing CPU throughput compared with earlier interpolation methods which used several instructions. The results are optionally rounded with the round-to-nearest algorithm.

### 6.2.13 - Development support

The following features have been implemented on the CPU32 to enhance the instrumentation and development environment :

- 68000 family development support,
- background debug mode,
- deterministic opcode tracking,
- hardware breakpoints.

#### 6.2.13.1 - 68000 family development support

All 68000 family members include features to facilitate applications development. These features include the following :

**Trace on instruction execution** - 68000 family processors include an instruction-by-instruction tracing facility as an aid to program development. The CPU32 also allows the user to trace only those instructions causing a change in program flow.

**Breakpoint instruction** - An emulator may insert software breakpoints into the target code to indicate when a breakpoint has occurred. On the CPU32, this function is provided via illegal instructions, \$4848 – \$484F, to serve as breakpoint instructions.

**Unimplemented instruction emulation** - During instruction execution, when an attempt is made to execute an illegal instruction, an illegal instruction exception occurs. Unimplemented instructions (F-line, A-line,...) utilize separate exception vectors to permit efficient emulation of unimplemented instructions in software.

#### 6.2.13.2 - Background debug mode

Microcomputer systems generally provide a debugger, implemented in software, for system analysis at the lowest level. The background debug mode on the CPU32 is unique in that the debugger has been implemented in CPU microcode. Registers can be viewed and/or altered, memory can be read or written to, and test features can be invoked. Incorporating these capabilities on-chip simplifies the environment in which the in-circuit emulator operates.

6.2.13.3 - Deterministic opcode tracking

CPU 32 function code outputs are augmented by two supplementary signals to monitor the instruction pipeline. The instruction pipe (PIPE) output indicates the start of each new instruction and each mid-instruction pipeline advance. The instruction fetch (FETCH) output identifies the bus cycles in which the operand is loaded into the instruction pipeline. Pipeline flushes are also signalled with IFETCH. Monitoring these two signals allows a bus analyzer to synchronize itself to the instruction stream and monitor its activity.

6.2.13.4 - On-chip breakpoint hardware

An external breakpoint input allows a breakpoint trap on any memory access.

Table 7 - Instruction set summary

Mnemonic	Description
ABCD ADD ADDA ADDI ADDQ ADDX AND ANDI ASL, ASR	Add Decimal with Extend Add Add Address Add Immediate Add Quick Add with Extend Logical AND Logical AND Immediate Arithmetic Shift Left and Right
Bcc BCHG BCLR BGND BKPT BRA BSET BSR BTST	Branch Conditionally Test Bit and Change Test Bit and Clear Background Breakpoint Branch Test Bit and Set Branch to Subroutine Test Bit
CHK, CHK2  CLR CMP CMPA CMPI CMPM CMP2	Check Register Against Upper and Lower Bounds Clear Compare Compare Address Compare Immediate Compare Memory to Memory Compare Register Against Upper and Lower Bounds
DBcc DIVS, DIVSL DIVU, DIVUL	Test Condition, Decrement and Branch Signed Divide Unsigned Divide
EOR EORI EXG EXT, EXTB	Logical Exclusive OR Logical Exclusive OR Immediate Exchange Registers Sign Extend
ILLEGAL	Take Illegal Instruction Trap
JMP JSR	Jump Jump to Subroutine
LEA LINK LPSTOP LSL, LSR	Load Effective Address Link and Allocate Low Power Stop Logical Shift Left and Right

Mnemonic	Description
MOVE MOVE CCR MOVE SR MOVE USP MOVEA MOVEC MOVEM MOVEP MOVEQ MOVES	Move Move Condition Code Register Move Status Register Move User Stack Pointer Move Address Move Control Register Move Multiple Registers Move Peripheral Move Quick Move Alternate Address Space
MULS, MULS.L MULU, MULU.L	Signed Multiply Unsigned Multiply
NBCD NEG NEGX NOP	Negate Decimal with Extend Negate Negate with Extend No Operation
OR ORI	Logical Inclusive OR Logical Inclusive OR Immediate
PEA	Push Effective Address
RESET ROL, ROR ROXL, ROXR RTD RTE RTR RTS	Reset External Devices Rotate Left and Right Rotate with Extend Left and Right Return and De-allocate Return from Exception Return and Restore Codes Return from Subroutine
SBCD Scc STOP SUB SUBA SUBI SUBQ SUBX SWAP	Subtract Decimal with Extend Set Conditionally Stop Subtract Subtract Address Subtract Immediate Subtract Quick Subtract with Extend Swap Register Words
TBLS, TBLSN TBLU, TBLUN TAS TRAP TRAPcc TRAPV TST	Signed/Unsigned Table Lookup and Interpolate Test Operand and Set Trap Trap Conditionally Trap on Overflow Test Operand
UNLK	Unlink

6.3 - Bus operation

This section provides a functional description of the bus and the signals that control it. Operation of the bus is the same whether the MCU or an external device is the bus master ; the names and descriptions of bus cycles are from the point of view of the bus master. The MCU architecture supports byte, word, and long-word operands, allowing access to 8-bit and 16-bit data ports through use of asynchronous cycles controlled by the data transfer (SIZ1 and SIZ0) and data size acknowledge pins (DSACK1 and DSACK0).

### 6.3.1 - Function codes

The function code signals (FC2-FC0) select one of eight 16 Mbyte address space to which the address applies.

### 6.3.2 - Address bus

The address bus signals (A23-A0) define the address of the byte (or the most significant byte) to be transferred during a bus cycle. The address is valid while  $\overline{AS}$  asserted.

### 6.3.3 - Address strobe

The Address Strobe ( $\overline{AS}$ ) is a timing signal that indicates the validity of an address on the address bus and of many control signals.

### 6.3.4 - Data bus

The data signals (D15-D0) comprise a bidirectional, non-multiplexed parallel bus that contains the data being transferred to or from the MCU. A read or write operation may transfer 8 or 16 bits of data (1 or 2 bytes) in one bus cycle.

### 6.3.5 - Data strobe

The Data Strobe ( $\overline{DS}$ ) is a timing signal that applies to the data bus. For a read cycle, the MCU asserts  $\overline{DS}$  to signal the external device to place data on the bus. For a write cycle,  $\overline{DS}$  signals to the external device that the data to be written is valid on the bus.

### 6.3.6 - Bus control signals

The MCU initiates a bus cycle by driving the address, size, function code, and read/write outputs. At the beginning of a bus cycle, the size signals (SIZ1, SIZ0) are driven along with the function code signals. SIZ1 and SIZ0 indicate the number of bytes remaining to be transferred during an operand cycle (consisting of one or more bus cycles). Table 8 shows the encoding of SIZ1 and SIZ0. The read/write ( $\overline{R/W}$ ) signal determines the direction of the transfer during a bus cycle. The read-modify-write cycle signal ( $\overline{RMC}$ ) is asserted at the beginning of the first bus cycle of a read-modify-write operation, and remains asserted until completion of the final bus cycle of the operation.

Table 8 - Size signal encoding

SIZ1	SIZ2	Transfer Size
0	1	Byte
1	0	Word
1	1	3 Byte
0	0	Long Word

### 6.3.7 - Bus cycle termination signals

During bus cycles, external devices assert the data transfer and size acknowledge signals  $\overline{DSACK1}$  and/or  $\overline{DSACK0}$  as part of the bus protocol. During a read cycle, this signals the MCU to terminate the bus cycle and to latch the data. During a write cycle, this indicates that the external device has successfully stored the data and that the cycle may terminate. These signals also indicate to the MCU the size of the port for the bus cycle just completed.

The bus error ( $\overline{BERR}$ ) signal is also a bus cycle termination indicator and can be used in the absence of  $\overline{DSACKx}$  to indicate a bus error condition. It can also be asserted in conjunction with  $\overline{DSACKx}$  to indicate a bus error condition, provided it meets the appropriate timing. Additionally, the  $\overline{BERR}$  and  $\overline{HALT}$  signals can be asserted simultaneously, in lieu of, or in conjunction with, the  $\overline{DSACKx}$  signals.

The internal bus monitor can be used to generate the  $\overline{BERR}$  signal for internal and internal-to-external transfers. An external bus master must provide its own  $\overline{BERR}$  generation and drive the  $\overline{BERR}$  pin, since the internal  $\overline{BERR}$  monitor has no information about transfers initiated by an external bus master.

Finally the autovector ( $\overline{AVEC}$ ) signal can be used to terminate interrupt acknowledge cycles, indicating that the MCU should internally generate a vector number to locate an interrupt handler routine.  $\overline{AVEC}$  is ignored during all other bus cycles.

### 6.3.8 - Dynamic bus sizing

The MCU dynamically interprets the port size of the addressed device during each bus cycle, allowing operand transfers to or from 8- and 16-bit ports. During an operand transfer cycle, the slave device signals its port size (byte or word) and indicates completion of the bus cycle to the MCU through the use of the  $\overline{DSACKx}$  encodings and assertion results. Refer to Table 9 for  $\overline{DSACKx}$  encodings and assertion results. For example, if the MCU is executing an instruction that reads a long-word operand from a 16-bit port, the MCU latches the 16 bits of valid data and runs another bus cycle to obtain the other 16 bits.

Dynamic bus sizing requires that the portion of the data bus for a transfer to or from a particular port size be fixed. For example an 8 bit port must reside on data bus bits 15-8.

The  $\overline{SIZx}$  signals also form part of the bus sizing protocol. These outputs indicate the remaining number of bytes to be transferred during the current bus cycle.

Table 9 - DSACK codes and results

DSACK1	DSACK0	Result
1 (Negated)	1 (Negated)	Insert wait states in current bus cycle
1 (Negated)	0 (Asserted)	Complete cycle - Data bus port size is 8 bits
0 (Asserted)	1 (Negated)	Complete cycle - Data bus port size is 16 bits
0 (Asserted)	0 (Asserted)	Reserved

### 6.3.9 - Bus operation

The MCU bus is used in an asynchronous manner. The external devices connected to the bus can operate at clock frequencies different from the clock for the MCU. Bus operation uses the handshake lines ( $\overline{AS}$ ,  $\overline{DS}$ ,  $\overline{DSACK1}$ ,  $\overline{DSACK0}$ ,  $\overline{BERR}$  and  $\overline{HALT}$ ) to control data transfers. Decoding the size outputs and lower address line A0 provides strobes that select the active portion of the data bus. The slave device (memory or peripheral) then responds by placing the requested data on the correct portion of the data bus for a read cycle or latching the data on a write cycle, and asserting the  $\overline{DSACK1}/\overline{DSACK0}$  combination that corresponds to the port size to end the cycle. If no slave responds or the access is invalid, external control logic asserts the  $\overline{BERR}$ , or  $\overline{BERR}$  and  $\overline{HALT}$  signal(s) to abort or retry the bus cycle, respectively.

### 6.3.10 - Fast termination cycles

With an external device that has a fast access time, the chip-select circuit fast-termination option can provide a two-cycle external bus transfer. Since the chip select circuits are driven from the system clock, the bus cycle termination is inherently synchronized with the system clock.

### 6.3.11 - Bus exception control cycles

The bus architecture requires assertion of  $\overline{DSACKx}$  from an external device to signal that a bus cycle is complete.  $\overline{DSACKx}$  or  $\overline{AVEC}$  is not asserted in these cases :

- The external device does not respond,
- No interrupt vector is provided,
- Various other application-dependent errors occur.

This MCU has a bus error input ( $\overline{BERR}$ ) when no device responds by asserting  $\overline{DSACKx}$  or  $\overline{AVEC}$  within an appropriate period of time after the MCU asserts the  $\overline{AS}$ . This allows the cycle to terminate and the MCU to enter exception processing for the error condition. Another signal that is used for bus exception control is the halt signal ( $\overline{HALT}$ ). This signal can be asserted by an external device for debugging purposes to cause single bus operation or (in combination with  $\overline{BERR}$ ) a retry of a bus cycle in error.

### 6.3.12 - Bus arbitration

The bus design of the MCU provides for a single bus master at any one time : either the MCU or an external device. One or more of the external devices on the bus can have the capability of becoming bus master. Bus arbitration is the protocol by which an external device becomes bus master ; the bus controller in the MCU manages the bus arbitration signals so that the MCU has the lowest priority. External devices that need to obtain the bus must assert the bus arbitration signals in a certain sequence. Systems that include several devices that can become bus master require external circuitry to assign priorities to the devices, so that when two or more external devices attempt to become bus master at the same time, the one having the highest priority becomes bus master first. The protocol is explained fully in the SIM manual, however here is the basic sequence of events :

- An external device asserts the bus request signal ( $\overline{BR}$ ),
- The MCU asserts the bus grant signal to indicate that the bus is available ( $\overline{BG}$ ),
- The external device asserts the bus grant acknowledge signal ( $\overline{BGACK}$ ) to indicate that it has assumed bus mastership.

Bus arbitration requests are recognized during normal processing,  $\overline{HALT}$  assertion, when the CPU has halted due to a double bus fault.

### 6.3.13 - Reset operation

The MCU has reset control logic to determine the cause of reset and synchronize it if necessary. If an external device drives the  $\overline{RESET}$  pin low, the reset control logic holds  $\overline{RESET}$  asserted internally until the external  $\overline{RESET}$  is released. When the reset control logic detects that the external  $\overline{RESET}$  is no longer being driven, it drives  $\overline{RESET}$  low for an additional 512 cycles to guarantee this length of reset to the entire system.

If  $\overline{RESET}$  is asserted from any other source, the reset control logic asserts  $\overline{RESET}$  for a minimum of 512 cycles and until the source of reset is negated.

Figure 19 is a timing diagram of the power-up reset operation, showing the relationships between  $\overline{\text{RESET}}$ ,  $V_{\text{DD}}$ , and bus signals. During the reset period, the entire bus (except for non-tri-statable signals, which are driven to their inactive state three-states). Once  $\overline{\text{RESET}}$  negates, all control signals are driven to their inactive state, the data bus is in read mode, and the address bus is driven. After this, the first bus cycle for  $\overline{\text{RESET}}$  exception processing begins.

$\overline{\text{RESET}}$  should be asserted for at least 590 clock periods to ensure that the MCU resets. Resetting the MCU causes any bus cycle in progress to terminate as if  $\overline{\text{DSACKx}}$  or  $\overline{\text{BERR}}$  has been asserted. In addition, the MCU initializes registers appropriately for a reset exception.

For further information refer to the System Integration Modul Manual.

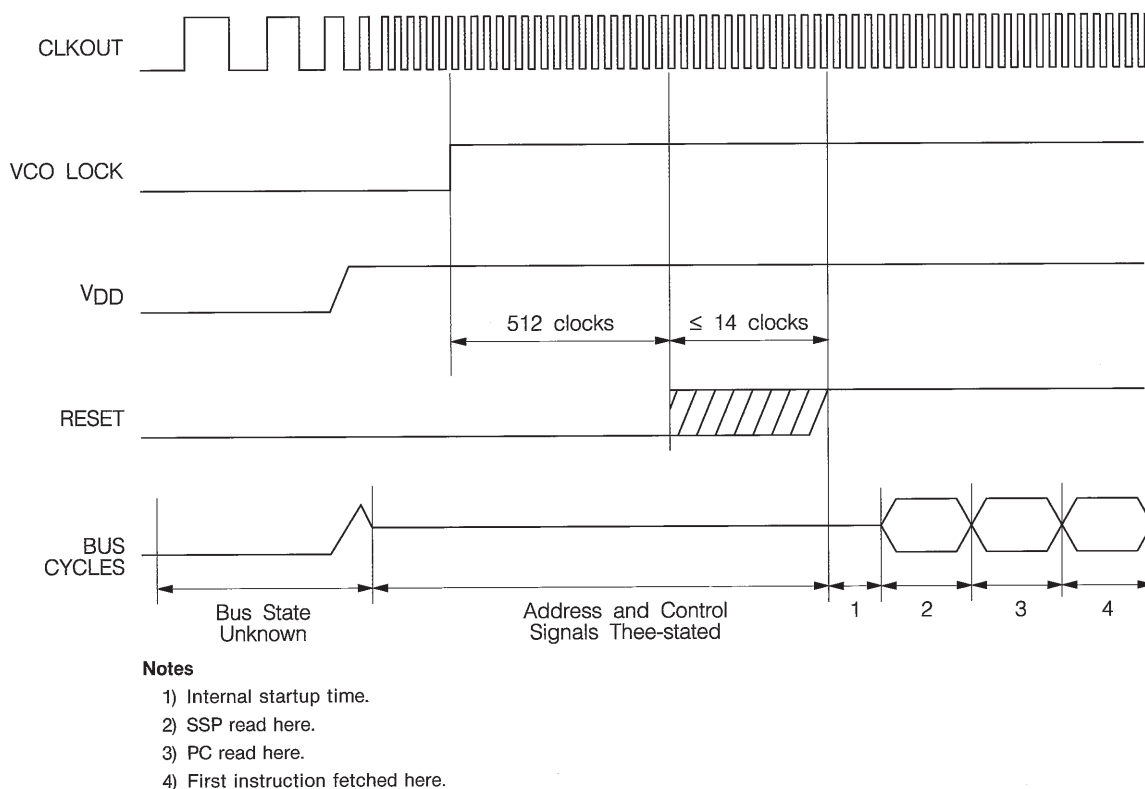


Figure 19 : Initial reset operation timing.

## 6.4 - System integration module

The TS 68332 system integration module (SIM) consists of five submodules that control the microcontroller unit (MCU) system start-up, initialization, configuration, and external bus with a minimum of external devices. The five submodules that make up the SIM, shown in Figure 20, are as follows :

- System configuration and protection,
- Clock synthesizer,
- Chip selects,
- External bus interface,
- System test.

### 6.4.1 - System configuration and protection submodule

The SIM module allows the user to control some features of system configuration by writing bits in the Module Configuration Register. This register also contains read-only status bits that show the state of some of the SIM features.

This MCU is designed with the concept of providing maximum system safe-guards. Many of the functions that normally must be provided in external circuits are incorporated in this MCU. The features provided in the system configuration and protection submodule are as follows :

#### System configuration

The module configuration register allows the user to configure the system according to the particular system requirements.

#### Internal bus monitor

The MCU provides an internal bus monitor to monitor the  $\overline{\text{DSACKx}}$  response time for all internal bus accesses. An option allows the monitoring of internal to external bus accesses. There are four selectable response times that allow for the response speed of peripherals used in the system. A bus error ( $\overline{\text{BERR}}$ ) signal is asserted internally if the  $\overline{\text{DSACKx}}$  response time is exceeded. When operating as a bus master, the  $\overline{\text{BERR}}$  signal is not asserted externally.

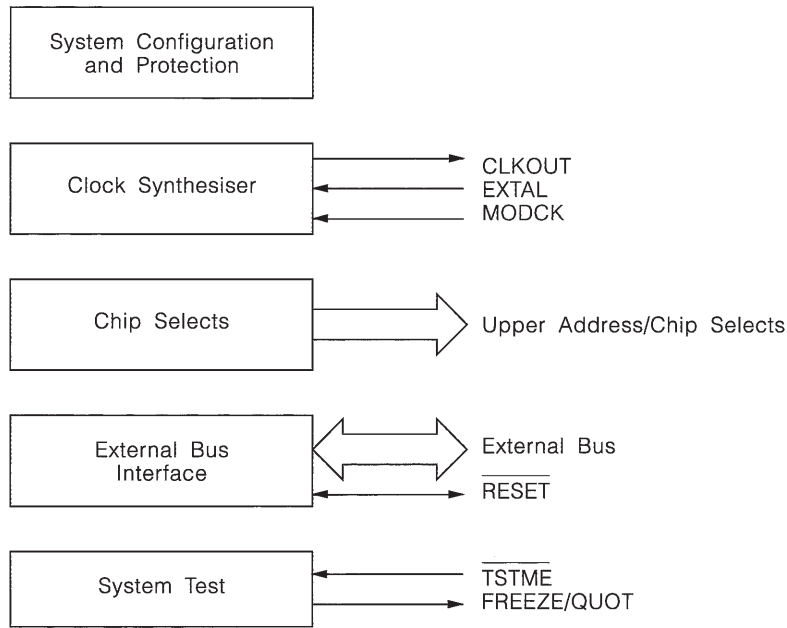


Figure 20 : System integration module block diagram.

**Halt monitor**

A halt monitor causes a reset to occur if the internal halt ( $\overline{HALT}$ ) is asserted by the CPU.

**Spurious interrupt monitor**

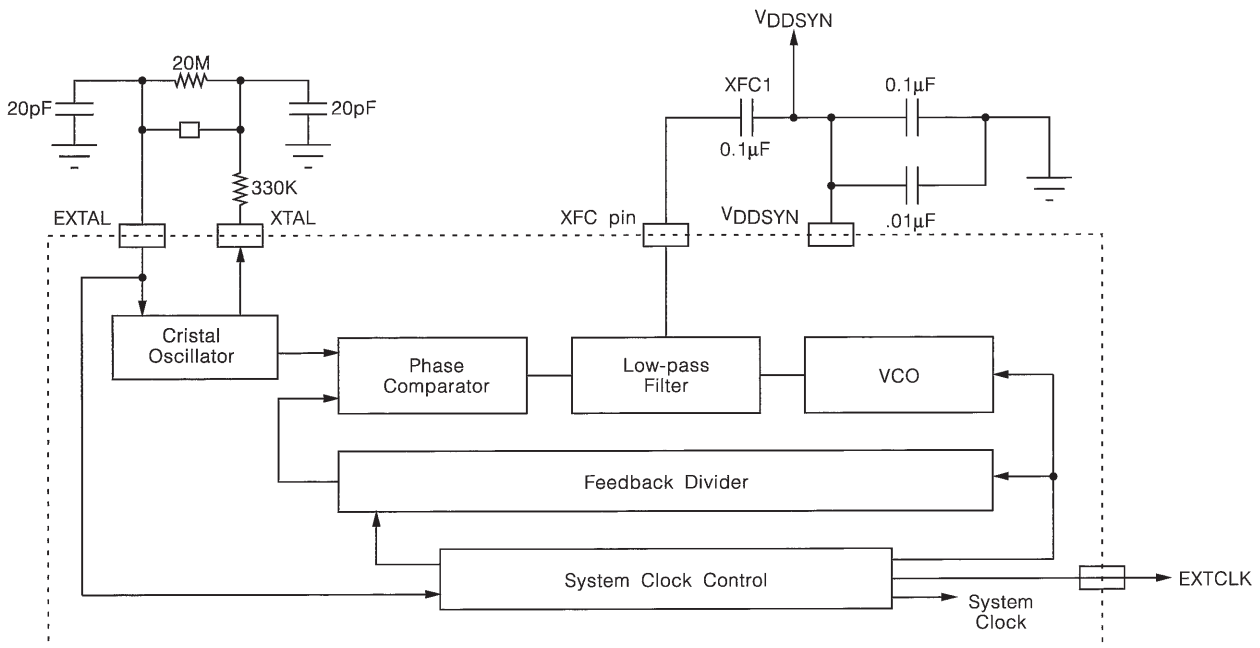
If no interrupt arbitration occurs during an interrupt acknowledge (IACK) cycle, the  $\overline{BERR}$  signal is asserted internally.

**Software watchdog**

The watchdog asserts  $\overline{RESET}$  if the software fails to service the software watchdog for a designated period of time (presumably because it is trapped in a loop or lost). There are four selectable timeout periods, and a prescaler may be used for long timeout periods.

**Periodic interrupt timer**

The MCU provides a timer to generate periodic interrupts. The periodic interrupt time period can vary from 122  $\mu$ s - 15.94 s (with a 32.768 kHz crystal used to generate the system clock).



Note : Must be low leakage capacitor.

Figure 21 : Clock submodule block diagram.

### 6.4.3 - Clock synthesiser

The clock synthesiser (Figure 21) can operate from an on-chip phase locked loop (PLL) using an external crystal connected between the EXTAL and XTAL pins as a reference frequency source. A 32.768 kHz watch crystal provides an inexpensive reference, but the reference crystal frequency can be any frequency from 25-50 kHz. Outside the 25-50 kHz range, an external oscillator can be used with the on-chip synthesiser and VCO, or the frequency can be driven directly into the EXTAL pin (the XTAL pin should be left floating for this case).

The system clock frequency is programmable from 131 kHz to the maximum clock frequency with a resolution of 131 kHz. A separate power pin ( $V_{DDSYN}$ ) is used to allow the clock circuits to run with the rest of the MCU powered down and to provide increased noise immunity for the clock circuits. If for some reason the external crystal signal is removed from the device then the clock synthesiser will generate its own internal clock signal to allow the device to enter some kind of error recovery routine. This is known as LIMP mode. The clock frequency generated will not have an associated timing spec but should be around 9 MHz.

### 6.4.4 - Chip-select submodule

Typical microcomputer systems require external hardware to provide select signals to external peripherals. This MCU integrates these functions on-chip in order to provide the cost, speed, and reliability benefits of a higher level of integration. The chip-select signals can also be programmed as output enable, read or write strobe, or IACK signals.

Since initialization software would probably reside in a peripheral memory device controlled by the chip-select circuits, a CSBOOT register provides default reset values to support bootstrap operation.

The chip-select submodule supports the following programmable features :

#### Twelve programmable chip-select circuits

Twelve chip select signals are available ( $\overline{CSBOOT}$  and  $\overline{CS10}$  to  $\overline{CS0}$ ). These signals use the  $\overline{CSBOOT}$  pin, bus arbitration pins  $\overline{BR}$ ,  $\overline{BG}$ , and  $\overline{BGACK}$ , function code pins FC2-FC0, and address pins A23-A19. The  $\overline{CSBOOT}$  pin is dedicated to a single function because it must function after a reset with no initialization, the other chip select circuits share functions on their output pins. All 12 chip select circuits are independently programmable from the same list of selectable features. Each chip select circuit has an individual base register and option register which contain the programmable characteristics of that chip select. Using these address lines as chip select signals does not restrict the large linear address space of the MCU since the chip select logic always uses the internal address lines.

#### Variable block sizes

The block size starting from the specified base address can be programmed as 2 K, 8 K, 16 K, 64 K, 128 K, 256 K, 512 Kbytes or 1 Mbyte.

#### Both 8-bit and 16-bit ports supported

Eight-bit ports are accessible on both odd and even addresses when connected to data bus bits 15-8. Sixteen-bit ports can be accessed as odd bytes, even bytes, or words.

#### Read only, write only, or read/write capability

Chip selects can be asserted synchronized with read, write, or both read and write.

#### Address strobe and data strobe timing option

Chip-select signals can be synchronized with either address strobe or data strobe, so that control signals such as output enable or write enable can be easily generated.

#### Internal $\overline{DSACK}$ generation with wait states

The port size programmed in the pin assignment register can be referenced for generating  $\overline{DSACK}$  and the proper number of wait states for a particular device programmed by the user.

#### Address space checking

Supervisor, user, and CPU space accesses can be optionally checked.

#### Interrupt priority level checking

In the IACK cycle, the acknowledged interrupt level can be compared with the user-specified level programmed in the option field. If autovector option is selected, AVEC is internally asserted.

#### Discrete output

Port C pins A22-A19 and FC2-FC0 can be programmed for discrete output, with data stored in the pin data register (CSPDR).

#### 68000-type peripheral support

68000-type peripherals that require an E clock for synchronization can be supported. Chip select is asserted, synchronized with the E clock on pin A23, providing correct data bus timing for the MCU.

### 6.4.5 - Test submodule

The test submodule is a primary tool to support all types of testing, such as production test and user self-test, that is integrated into the MCU. The submodule supports scan-based testing of the various modules in the MCU. The scan test employed here consists of the test submodule performing the following steps :

- serially shifting stimulus data to an idle module under test (MUT),
- activating the module under test,
- serially shifting response data back from the module under test,
- latching the response data for interrogation by the bus master.

The further information to the System Integration Module Manual.

## 6.5 - QSM Queued Serial Module

The queued serial module (QSM) provides the micro-controller unit (MCU) with two serial communication interfaces divided into two submodules: the queued serial peripheral interface (QSPI) and the serial communications interface (SCI). The QSPI is a full-duplex, synchronous serial interface for communicating with peripherals and other MCUs. It is enhanced by the addition of a RAM queue for receive and transmit data. The SCI is a full-duplex universal asynchronous receiver transmitter (UART) serial interface. These submodules operate independently (see Figure 22).

### 6.5.1 - QSM pins

The QSM has nine external pins. Eight of these pins can be used as general-purpose I/O pins, if the pin is not being used for its submodule function. The ninth pin, RXD, is an input-only pin used exclusively by the SCI submodule. The pins are identified as follows:

MISO - Master In Slave Out,  
 MOSI - Master Out Slave In,  
 SCK - Serial Clock,  
 PCS0/SS - Peripheral Chip Select 0/ Slave Select,  
 PCS3-PCS1 - Peripheral Chip Selects 3-1,  
 TXD - Transmit Data,  
 RXD - Receive Data.

### 6.5.2 - QSPI submodule

The QSPI submodule communicates with external peripherals and other MCUs via a synchronous serial bus. The QSPI is fully compatible with the serial peripheral interface (SPI) systems found on other TCS devices such as the 68HC11 and 68HC05 families. It has all of the capabilities of the standard SPI system as well as several new features. The following paragraphs describe the main features of the QSPI.

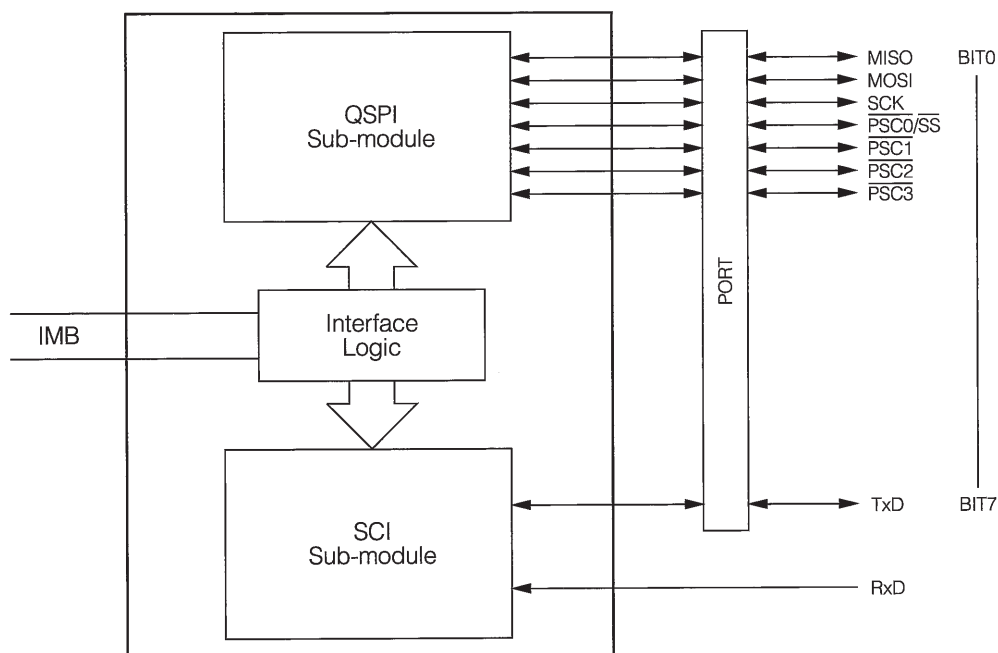


Figure 22 : QSM block diagram.

### 6.5.3 - QSPI features

Standard SPI features are listed below, followed by a list of the additional features offered on the QSPI:

- full duplex, three-wire synchronous transfers,
- half-duplex, two-wire synchronous transfers,
- master or slave operation,
- programmable master bit rates,
- programmable clock polarity and phase,
- end-of-transmission interrupt flag,
- master-master mode fault flag,
- easily interfaces to simple expansion parts (A/D converters, EEPROMs, display drivers, etc.).

### 6.5.3.1 - QSPI enhanced features

#### Programmable queue

A programmable queue allows the QSPI to perform up to 16 serial transfers without CPU intervention. Each transfer corresponds to a queue entry containing all the information needed by the QSPI to independently complete one serial transfer. This unique feature greatly reduces CPU/QSPI interaction, resulting in increased CPU and system throughput.

Once the CPU has set up the queue of QSPI commands and enabled the QSPI, the QSPI operates independently of the CPU. The QSPI executes all of the commands in its queue, sets a flag indicating that it has finished, and then either interrupts the CPU or waits for CPU intervention.

#### Programmable peripheral chip selects

Four peripheral chip-select pins allow the QSPI to access up to 16 independent peripherals by decoding the four peripheral chip-select signals. Up to four independent peripherals can be selected by direct connection to a chip-select pin. The peripheral chip selects simplify interfacing to two or more serial peripherals by providing dedicated peripheral chip-select signals and thus alleviating the need for CPU intervention.

#### Wraparound transfer mode

Wraparound transfer mode allows for automatic, continuous re-execution of the preprogrammed queue entries. Newly transferred data replaces previously transferred data. Wraparound simplifies interfacing with A/D converters by automatically providing the CPU with the latest conversions in the QSPI RAM. Consequently, serial peripherals appear as memory-mapped parallel devices to the CPU.

#### Programmable transfer length

The number of bits in a serial transfer is programmable from 8 to 16 bits, inclusive. For example, 10 bits could be used for communicating with an external 10-bit A/D converter. Likewise, a vacuum fluorescent display driver might require a 12-bit serial transfer. The programmable length simplifies interfacing to serial peripherals that require different data lengths.

#### Programmable transfer delay

An inter-transfer delay may be programmed from approximately 1 to 500  $\mu\text{s}$  (using a 16.78 MHz system clock). For example, an A/D converter may require time between transfers to complete a new conversion. The default delay is 1  $\mu\text{s}$ . The programmable length of delay simplifies interfacing to serial peripherals that require delay time between data transfers.

#### Programmable queue pointer

The QSPI has a pointer that points to the queue location containing the data for the next serial transfer. The CPU can switch from one task to another in the QSPI by writing to the queue pointer, changing the location in the queue that is to be transferred next. Otherwise, the pointer increments after each serial transfer. By segmenting the queue, multiple-task support can be provided by the QSPI.

#### Continuous transfer mode

The continuous transfer mode allows the user to exchange an uninterrupted bit stream with a peripheral. A minimum of 8 bits and a maximum of 256 bits may be transferred in a single burst without CPU intervention. Longer transfers are possible; however, minimal CPU intervention is required to prevent loss of data. A 1 microsecond pause (using a 16.78 MHz system clock) is inserted between each entry transfer.

#### QSPI RAM

The QSPI uses an 80-byte block of dual-access static RAM that can be accessed by both the QSPI and the CPU. Because of sharing, the length of time taken by the CPU to access the QSPI RAM, when the QSPI is enabled, may be longer than when the QSPI is disabled. From one to four CPU wait states may be inserted by the QSPI in the process of reading or writing.

The RAM is divided into three segments: receive data, transmit data, and command control. Receive data is information received from a serial device external to the MCU. Transmit data is information stored by the CPU for transmission to an external peripheral chip. Command control contains all the information needed by the QSPI to perform the transfer. Figure 23 illustrates the organization of the RAM.

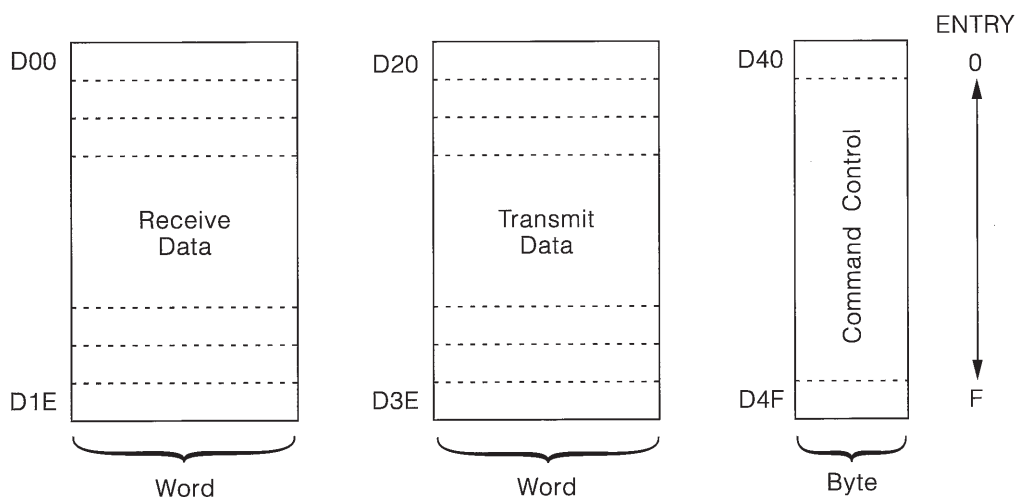


Figure 23 : Organization of the QSPI RAM.

### 6.5.4 - SCI Submodule

The SCI submodule is used to communicate with external devices and other MCUs via an asynchronous serial bus. The SCI is fully compatible with the SCI systems found on other TCS MCUs such as the 68HC11 and 68HC05 Families. It has all of the capabilities of previous SCI systems as well as several significant new features.

#### 6.5.4.1 - Features

Standard SCI features are listed below, followed by a list of additional features offered :

##### Standard SCI two-wire systems features

- standard Non Return to Zero (NRZ) mark/space format,
- advanced error detection mechanism (detects noise duration up to 1/16 of a bit-time),
- full-duplex operation,
- software selectable word length (8- or 9-bit words),
- separate transmitter and receiver enable bits,
- may be interrupt driven,
- four separate interrupt enable bits.

##### Standard SCI receiver features

- receiver wake up function (idle or address mark bit),
- idle-line detect,
- framing error detect,
- noise detect,
- overrun detect,
- receive data register full flag.

##### Standard SCI transmitter features

- transmit data register empty flag,
- transmit complete flag,
- send break.

##### QSM-enhanced SCI two-wire systems features

- 13-bit programmable baud rate modulus counter,
- even/odd parity generation and detection.

##### QSM-enhanced SCI receiver features

- two idle-line detect modes,
- receiver active flag.

##### 13-bit programmable baud rate modulus counter

A baud rate modulus counter has been added to provide the user with more flexibility in choosing the crystal frequency for the system clock. The modulus counter allows the SCI baud rate generator to produce standard transmission frequencies for a wide range of system clocks. The user is no longer constrained to select crystal frequencies based on the desired serial baud rate. This counter provides baud rates from 64 baud to 524 baud with a 16.78 MHz system clock.

##### Even/odd parity generation and detection

The user now has the choice either of seven or eight data bits plus one parity bit, or of eight or nine data bits with no parity bit. Even or odd parity is available. The transmitter automatically generates the parity bit for a transmitted byte. The receiver detects when a parity error has occurred on a received byte and sets a parity error flag.

##### Two idle-line detect modes

Standard TCS SCI systems detect an idle line when 10 or 11 consecutive bit-times are all ones. Used with the receiver wake up mode, the receiver can be awakened prematurely if the message preceding the start of the idle line contained ones in advance of its stop bit. The new (second) idle-line detect mode only starts counting idle time after a valid stop bit is received, which ensures correct idle-line detection.

##### Receiver Active Flag (RAF)

RAF indicates the status of the receiver. It is set when a possible start bit is detected and is cleared when an idle line is detected. RAF is also cleared if the start bit is determined to be line noise. This flag can be used to prevent collisions in systems with multiple masters.

For further information refer to the System Integration Module Manual.

### 6.6 - Standby RAM (with TPU emulation)

The TS 68332 contains 2 K bytes of standby RAM. This section describes the operation and control of the RAM module.

#### 6.6.1 - Overview

The RAM module contains 2048 bytes of fully static RAM, powered by  $V_{DD}$  in normal operation. The entire array may be used as standby RAM if power is supplied to the  $V_{STBY}$  pin. Switching between  $V_{DD}$  and  $V_{STBY}$  occurs automatically.

The RAM may be used as general-purpose memory for the microcontroller unit (MCU), providing fast, two-clock accesses to the CPU. Typically, the RAM is used for program control stacks and frequently modified data variables. The CPU may read or write byte, word, or long-word data.

The RAM may also be used as microcode control memory for the time processor unit (TPU). The TPU must be placed in emulation mode to use the RAM in this manner which allows users to develop their own TPU microcode primitives.

### 6.6.2 - RAM array addressing

The RAM array can be placed anywhere in the address map of the MCU by means of the array base address (RAMBAR), provided that it is placed on a 2 kbyte boundary and does not overlap the three RAM module control registers used for control and testing. RAMBAR can be written only once after reset. This prevents the RAM array from being accidentally remapped by software.

### 6.6.3 - TPU emulation mode operation

The RAM array may be used as the microcode control store for the TPU module. This mode of operation is selected from within the TPU. See **Development support** in the TPU manual for a complete description.

The TPU is connected to the RAM via a dedicated bus. While in emulation mode, the access timing of the RAM module matches the timing of the TPU microinstruction ROM to ensure accurate emulation. Normal accesses via the IMB are inhibited and the control register have to effect, allowing external RAM to emulate the 2 K RAM array at the same addresses.

The further information refer to the System Integration Module Manual.

## 6.7 - TPU overview

The time processing unit (TPU) performs simple as well as complex timing tasks, independently from the CPU, making it the latest advance in timer systems. Viewed as a special purpose microcomputer, this processor performs two operations, match and capture, on one operand: TIME. Every occurrence of either action is called an event. The servicing of these events by the TPU replaces the servicing of interrupts by the host central processing unit (CPU). The timing functions currently synthesized are the following:

- discrete input/output,
- input capture/input transition counter,
- output compare,
- pulse width modulation,
- synchronised pulse width modulation,
- period measurement with additional transition detect,
- period measurement with missing transition detect,
- position-synchronized pulse generator,
- stepper motor,
- period/pulse-width accumulator.

The advanced TPU affords for the first time high-resolution timing and multiple time function capability (flexibility) in the timer system pins.

### 6.7.1 - High-resolution timing

High-resolution timing is limited by CPU overhead required for servicing timing tasks such as period measurement, pulse measurement, pulse-width modulated waveform generation, etc. On the TPU, high-resolution timing is achieved by two main capabilities:

- reduced latency,
- reduced service time, which free the CPU to focus on other responsibilities.

The TPU provides a higher resolution than the CPU could achieve, and creates no CPU overhead for servicing timing tasks.

#### 6.7.1.1 - Latency

Latency is the interval of time from an even to the start of event servicing. The ability of the TPU to service its own interrupts or events reduces latency and the CPU is not required to service each input transition capture that occurs on a pin, or to determine each match time required for waveform synthesis. Once configured by the host CPU, the self-contained TPU performs complex time functions requiring high resolution with little or no CPU intervention.

#### 6.7.1.2 - Service time

Service time is the time expended servicing an event. In older microcontroller unit (MCU) timer functions, the service time is constrained because the MCU instruction set is not optimized for time function synthesis. The TPU instruction set is optimized, and time functions are synthesized with fewer instructions than the CPU. Instructions execute faster and service time is reduced. Instructions executed by the TPU are not user software, but firmware, special-purpose microcode written by TCS to perform as set of time functions. Microcode is placed into the TPU control store (ROM) when the device is manufactured.

### 6.7.2 - Features

- 16 channels ; each channel associated with a pin.
- Each channel can perform any time function,
- Each time function may be assigned to more than one channel at a given time.
- Each channel has an event register comprised of the following :
  - 16-bit capture register,
  - 16-bit compare/match register,
  - 16-bit greater-than or equal-to comparator.
- Each channel can be synchronized to one or both of the two 16-bit free-running timer count registers (TCR1 and TCR2).

- TCR1 is clocked from the output of a prescaler. The prescaler’s input is the internal TPU system clock divided by either 4 or 32. The four settings of the prescaler are divide by 1, 2, 4 and 8. Channels using TCR1 have the capability to resolve down to the TPU system clock divided by four.
- TCR2 is clocked from the output of a prescaler. The prescaler’s input is the external TCR2 pin. The four settings of the prescaler are divide by 1, 2, 4 and 8. Channels using TCR2 have the capability to resolve down to the TPU system clock divided by 8.
- TCR2 may be used as a hardware pulse accumulator clocked from the external TCR2 pin, or as a gated pulse accumulator of the clock that increments TCR1.
- All channels have at least six 16-bit parameter registers. Channels 14 and 15 each have eight 16-bit parameter registers. All parameter registers are contained in a dual-port RAM, accessible from both the TPU and CPU.
- A scheduler with three priority levels segregates high, middle, and low-priority time functions. Any channel may be assigned to one of these three priority levels.
- All time functions are microcoded.
- Emulation and development support is provided for all time function features such as breakpoint, freeze and single step, giving internal register accessibility.
- Coherent transfer capability for two parameter is provided in hardware.
- Coherent transfer capability for N parameters may be performed as a TPU microcode function. (Refer to Development support in the TPU reference manual for further details on this feature).

**6.7.3 - General concept**

The TPU is an intelligent, semi-autonomous peripheral dedicated to timing control. Its intelligence enables the servicing of timing events without CPU intervention. This device uses a private microengine for a processor, a scheduler, input/output channels, ROM instructions, and shared-access data RAM to operate independently and simultaneously with the CPU (see Figure 24). Consequently, the setup and service time for each timer event is minimized.

A «time-of-delay» approach is used where all time functions are related to one of two 16-bit free-running TCRs. Time functions are synthesized by combining the two time primitives, match and capture events. By performing these time primitives in hardware, the TPU can precisely determine the time when a match event is to occur and then specify the state of the output pin accordingly. The TPU can also accurately record the time at which an input transition occurs and can perform calculations based on the time of the occurrence. An event register for each channel provides for simultaneity of match/capture-event occurrences on all channels.

When a match or input capture event requiring service occurs on a channel, the channel generates a service request to the scheduler. The scheduler prioritizes the request with other pending service requests. When the microengine is idle, the scheduler causes the microengine to execute a microcode sequence. When the microengine is busy, the new sequence begins when the code being executed ends. The microengine performs the function, which is defined by the content of the control store, using parameters from the parameter RAM and from the event registers, etc., as needed. Following is an example.

Channel X is generating a periodic waveform and presently the output is high. When the value of the TCR used by that channel increments to match the value of the event register of channel X, a match event occurs. The event switches the output to low and generates a new service request to the scheduler. The scheduler then schedules and initiates service of channel X by the microengine.

When execution of the sequence begins, the microengine uses the execution unit :

- to obtain (from the parameter RAM) the value representing the duration of counts for which channel X should remain low, and
- to add to this value the value from the content of the event register of channel X. The content of the event register is then replaced by this sum ; the channel control is set for a match event on the same TCR ; and the pin control is set to cause the output pin for channel X to switch high when the event occurs. A channel interrupt, which signals the end of service to the CPU, may be asserted (if the time function provides for it and the interrupt is enabled). The microengine is then free to service the next event determined by the scheduler.

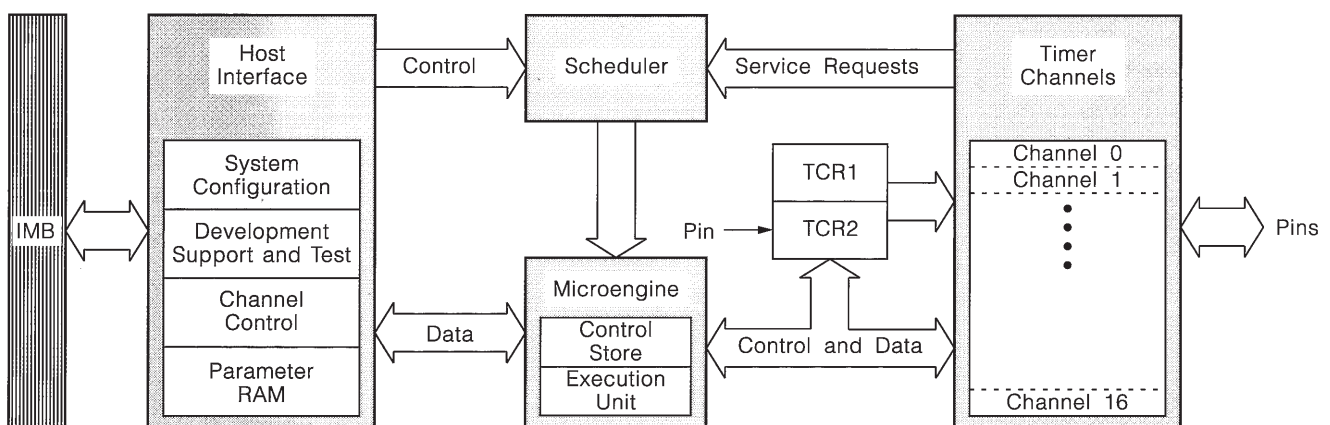


Figure 24 : TPU simplified block diagram.

### 6.7.4 - Flexibility

The TPU has the flexibility to be configured to directly solve the user's timer requirements. This flexibility is attained through five capabilities :

- channel orthogonality,
- inter-channel communication,
- programmable channel service priority,
- selection of timing functions,
- emulation capability.

#### 6.7.4.1 - Channel orthogonality

Traditionally, timer systems have been limited by the specific functions of channel pins dedicated to perform time functions such as input capture, output compare, or pulse accumulation. All channels of the TPU contain identical hardware and are functionally equivalent in operation, such that any channel can be configured to perform any time function. The user controls the combination of time functions ; the only constraint is the number of pins available for timing functions.

#### 6.7.4.2 - Inter-channel communication

The TPU's ability service itself requires a continuous flow of direct and indirect communication. Direct communication is accomplished through a «change channel» feature in which any channel of the TPU can operate an another channel to affect its state. Indirect communication is provided by a link feature in which any channel can lonk to one more channels, including itself, to signal a need for future service. As a result, the user can reference the operation of one channel to the occurrence of a specific action on another channel.

#### 6.7.4.3 - Programmable channel service priority

Applications may require different priorities of event service. The channel service priority may be programmed to one of three levels : high, middle, and low. The scheduler allows calculation of worst-case latency for event servicing and ensures servicing of all channels by preventing permanent blockage.

#### 6.7.4.4 - Selection of timing functions

The available timing functions can be programmed to operate on any channel. Parameter registers associated with each channel are used as general-purpose time operands.

#### 6.7.4.5 - Emulation capability

The TPU cannot resolve all timer problems using predefined time functions alone ; therefore, development of user-defined time functions is allowed in emulation mode. Using the RAM module of the MCU as a «writable control store» provides TPU emulation. In TPU emulation mode, an auxiliary bus connection is made between the RAM module and the TPU module, and access to the RAM module via the intermodule bus is disabled. A 9-bit address bus, a 32-bit data bus, and control lines transfer information between the modules. To ensure exact emulation, the access timing of the RAM module remains consistent with the TPU ROM control store.

### 6.7.5 - Applications

The TPU's high speed, versatile architecture, and time functions facilitate its use in many control applications, such as stepper motors and angle-based engine control. Control of a stepper motor or an angle-based automotive engine usually requires high CPU overhead. These applications show how the SM, PMA/PPM, and PSP time functions minimize the overhead associated with these applications, and provide sophistication and flexibility for a wide variety of applications.

Further detailed information on the TPU is found in the TPU reference manuel.

## 7 - PREPARATION FOR DELIVERY

### 7.1 - Packaging

Microcircuit are prepared for delivery in accordance with MIL-M-38510.

### 7.2 - Certificate of compliance

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

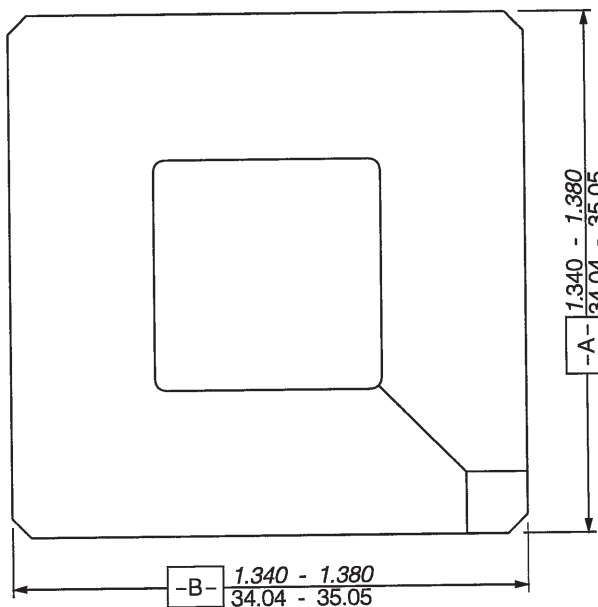
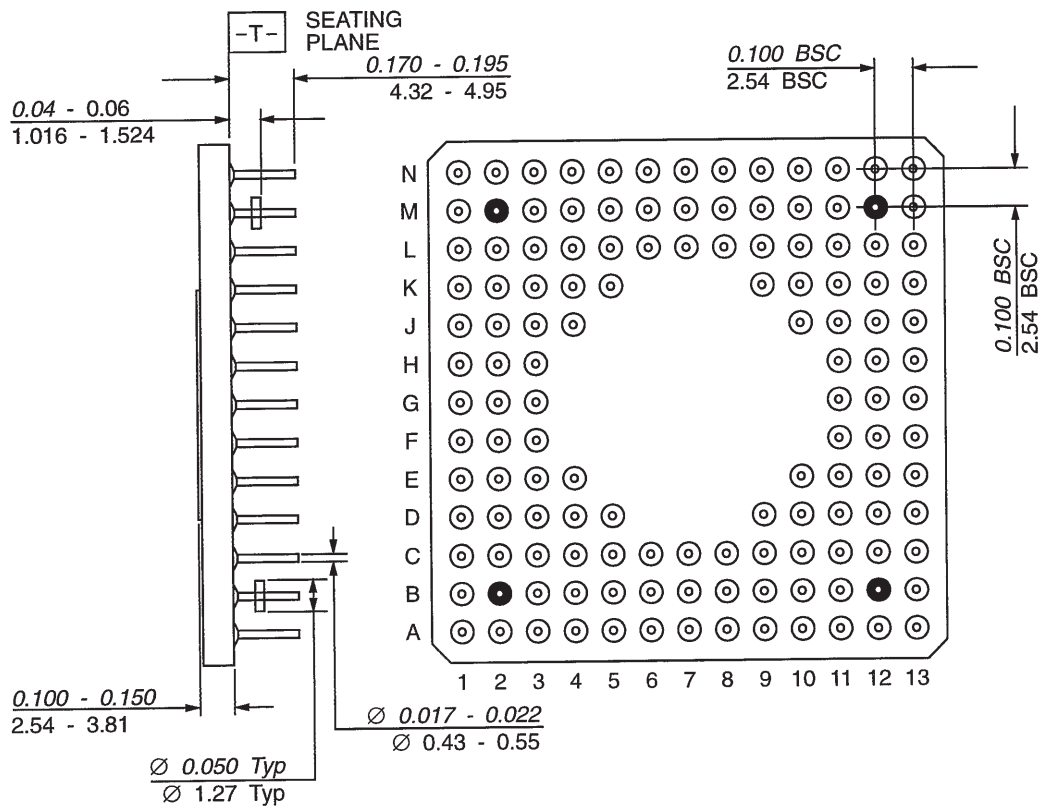
## 8 - HANDLING

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 :

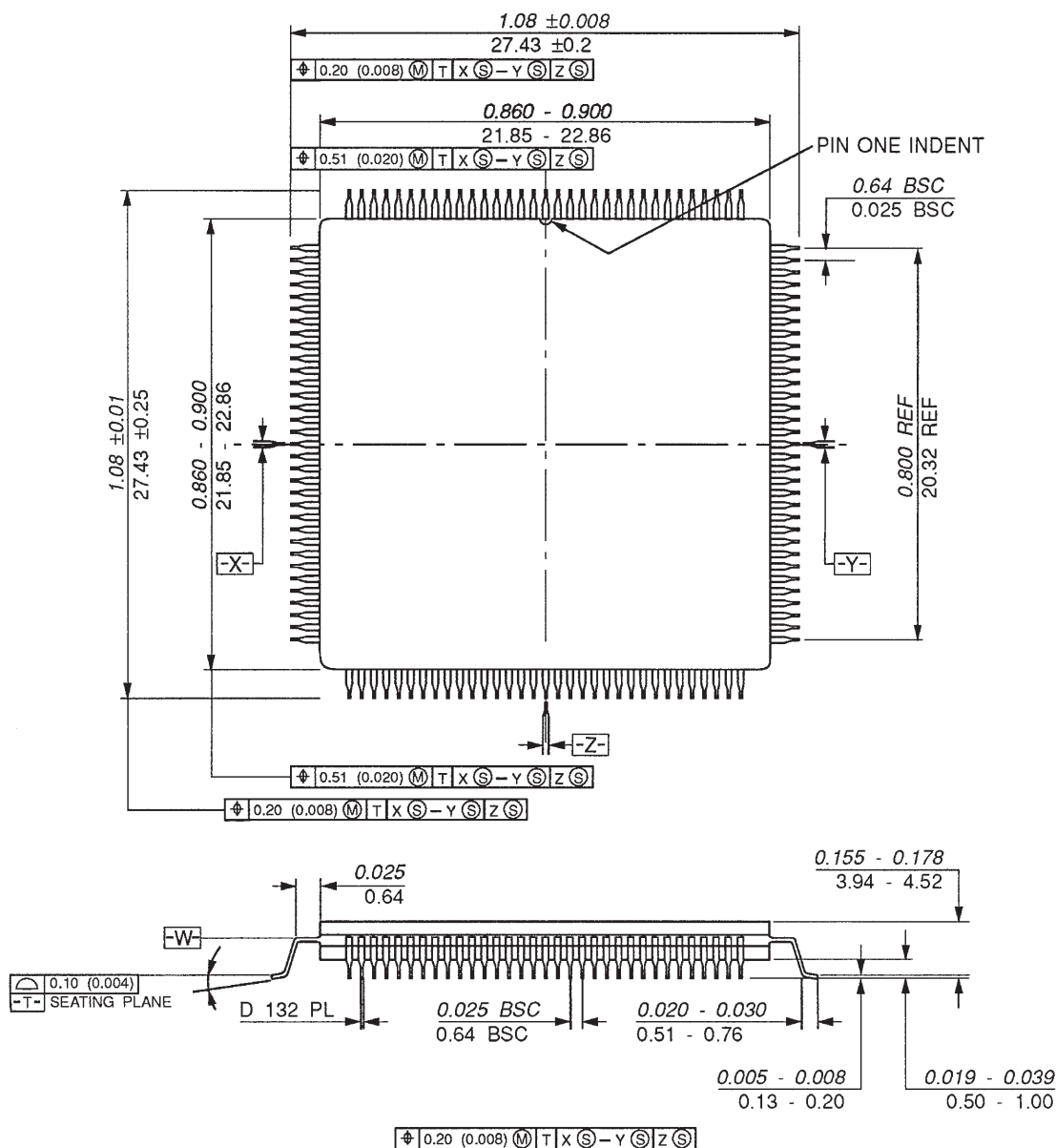
- a) Devise should be handled on benches with conductive and grounded surface.
- b) Ground test equipment, tools and operator.
- c) Do not handle devices by the leads.
- d) Store devices in conductive foam or carriers.
- e) Avoid use of plastic, rubber, or silk in MOS areas.
- f) Maintain relative humidity above 50 %, if practical.

9 - PACKAGE MECHANICAL DATA

9.1 - 132 pins - Ceramic Pin Grid Array



9.2 - 132 pins - Ceramic



Note 1 : Dimensionning and tolerancing per ansi Y14.5M, 1982.

Note 2 : Controlling dimensions : inch.

Note 3 : Dim A and B define maximum ceramic body dimensions including glass protrusion and mismatch of ceramic body top and bottom.

Note 4 : Datum plane -W- is located at the underside of leads where leads exit package body.

Note 5 : Datums X-Y and Z to be determined where center leads exit package body at datum -W-.

Note 6 : Dim S and V to be determined at seating plane, datum -T-.

Note 7 : Dim A and B to be determined at datum plane -T-.

10. - TERMINAL CONNECTIONS

10.1 - 132 pins - Ceramic Pin Grid Array

See Figure 2.1 page 4.

10.2 - 132 pins - Ceramic Ledged Chip Carrier / CERQUAD

See Figure 2.2 page 5.

## 11 - ORDERING INFORMATION

## 11.1 - Hi-REL product

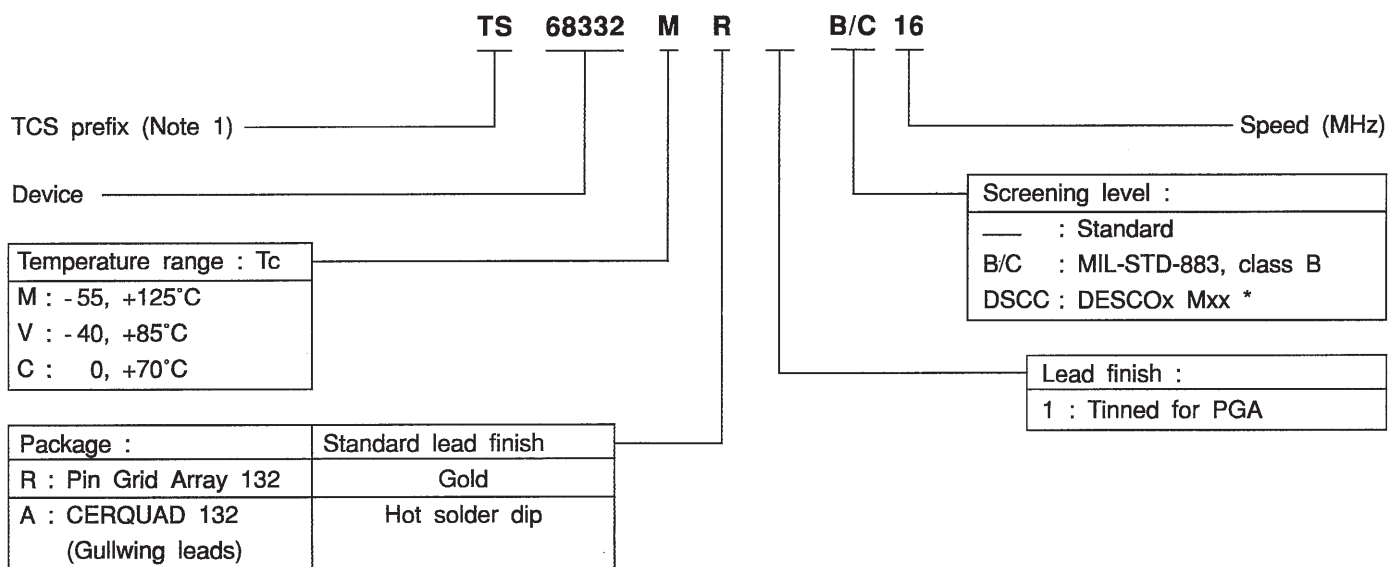
Commercial TCS Part-Number (see Note)	Norms	Package	Temperature range T <sub>C</sub> (°C)	Frequency (MHz)	Drawing number
TS68332MRB/C16	MIL-STD-883	PGA 132	- 55 / + 125	16.78	TCS data-sheet
TS68332MR1B/C16	MIL-STD-883	PGA 132 tin	- 55 / + 125	16.78	TCS data-sheet
TS68332MAB/C16	MIL-STD-883	CERQUAD 132	- 55 / + 125	16.78	TCS data-sheet
TS68332MRB/C20	MIL-STD-883	PGA 132	- 55 / + 125	20.97	TCS data-sheet
TS68332MR1B/C20	MIL-STD-883	PGA 132 tin	- 55 / + 125	20.97	TCS data-sheet
TS68332MAB/C20	MIL-STD-883	CERQUAD 132	- 55 / + 125	20.97	TCS data-sheet
TS68332DESC01MZA	MIL-STD-883	PGA 132 tin	- 55 / + 125	16.78	5962-9150101MZA
TS68332DESC01MZC	MIL-STD-883	PGA 132	- 55 / + 125	16.78	5962-9150101MZC
TS68332DESC02MZC (tbc)	MIL-STD-883	PGA 132	- 55 / + 125	20.97	5962-9150102MZC
TS68332DESC01MXA (tbc)	MIL-STD-883	CERQUAD 132	- 55 / + 125	16.78	5962-9150101MXA
TS68332DESC02MXA (tbc)	MIL-STD-883	CERQUAD 132	- 55 / + 125	20.97	5962-9150102MXA

**Note :** THOMSON-CSF SEMICONDUCTEURS SPECIFIQUES.

## 11.2 - Standard product

Commercial TCS Part-Number (see Note)	Norms	Package	Temperature range T <sub>C</sub> (°C)	Frequency (MHz)	Drawing number
TS68332VR16	TCS Standard	PGA 132	- 40 / + 85	16.78	TCS data-sheet
TS68332MR16	TCS Standard	PGA 132	- 55 / + 125	16.78	TCS data-sheet
TS68332VA16	TCS Standard	CERQUAD 132	- 40 / + 85	16.78	TCS data-sheet
TS68332MA16	TCS Standard	CERQUAD 132	- 55 / + 125	16.78	TCS data-sheet
TS68332VR20	TCS Standard	PGA 132	- 40 / + 85	20.97	TCS data-sheet
TS68332MR20	TCS Standard	PGA 132	- 55 / + 125	20.97	TCS data-sheet
TS68332VA20	TCS Standard	CERQUAD 132	- 40 / + 85	20.97	TCS data-sheet
TS68332MA20	TCS Standard	CERQUAD 132	- 55 / + 125	20.97	TCS data-sheet

**Note :** THOMSON-CSF SEMICONDUCTEURS SPECIFIQUES.



\* cf DSCC detailed specification or Hi-Rel ordering information p. 42

Information furnished is believed to be accurate and reliable. However THOMSON-CSF SEMICONDUCTEURS SPECIFIQUES assumes no responsibility for the consequences of use of such information nor for any infringement of patents or other rights of third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of THOMSON-CSF SEMICONDUCTEURS SPECIFIQUES. Specifications mentioned in this publication are subject to change without notice. This publication supersedes and replaces all information previously supplied. THOMSON-CSF SEMICONDUCTEURS SPECIFIQUES products are not authorized for use as critical components in life support devices or systems without express written approval from THOMSON-CSF SEMICONDUCTEURS SPECIFIQUES. © 1997 THOMSON-CSF SEMICONDUCTEURS SPECIFIQUES - Printed in France - All rights reserved.

This product is manufactured by THOMSON-CSF SEMICONDUCTEURS SPECIFIQUES - 38521 SAINT-EGREVE / FRANCE.

For further information please contact : THOMSON-CSF SEMICONDUCTEURS SPECIFIQUES - Route Départementale 128 - B.P. 46 - 91401 ORSAY Cedex / FRANCE - Tél. : (33)(0)1.69.33.00.00 / Téléfax : (33)(0)1.69.33.03.21.