8 K x 8 CMOS Dual Port RAM Low Voltage 3.3 Volt

Introduction

The TSM67V05 is a very low power CMOS dual port static RAM organized as 8192 × 8. The TSM67V05 is designed to be used as a stand-alone 8 bit dual port RAM or as a combination MASTER/SLAVE dual port for 16 bit or more width systems. Using the TEMIC MASTER/SLAVE dual port approach in memory system applications results in full speed, error free operation without the need for additional discrete logic.

Master and slave devices provide two independant ports with separate control, address and I/O pins that permit independant, asynchronous access for reads and writes to any location in the memory. An automatic power down feature controlled by $\overline{CS}$ permits the onchip circuitry of each port in order to enter a very low stand-by power mode.

Using an array of eight transistors (8T) memory cell and fabricated with the state of the art 0.65 µm lithography named SCMOS, the TSM67V05 combines an extremely low standby supply current (typ = 1.0 µA) with a fast access time at 35 ns over the full temperature range. All versions offer battery backup data retention capability with a typical power consumption at less than 3.3 µW.

For military/space applications that demand superior levels of performance and reliability the TSM67V05 is processed according to the methods of the latest revision of the MIL STD 883 (class B or S) and/or ESA SCC 9000.

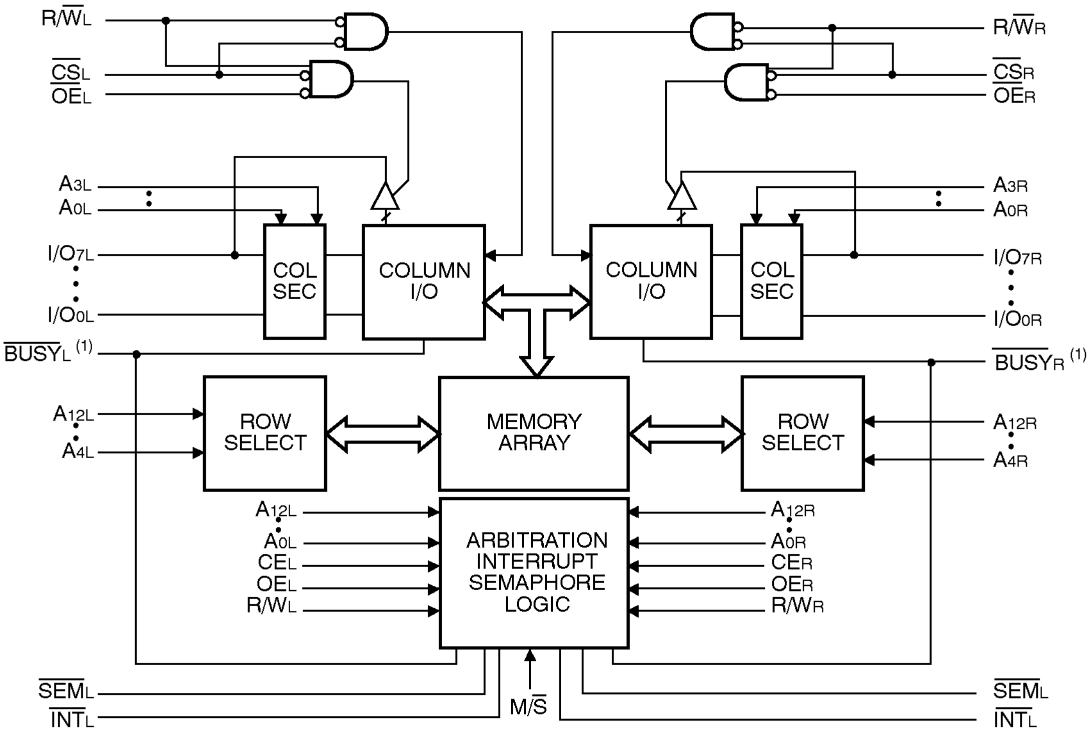
Features

- 3.3 V ± 0.3 V low voltage only
- Fast access time
  35 ns to 70 ns
- Wide temperature range :
  –55 ° to +125 °C
- TSM67V05 L low power
  TSM67V05 V very low power
- Expandable data bus to 16 bits or more using master/slave chip select when using more than one device on chip arbitration logic
- Versatile pin select for master or slave :
  – M/$\overline{S}$ = H for busy output flag on master
  – $\overline{M}$/S = L for busy input flag on slave
- INT flag for port to port communication
- Full hardware support of semaphore signaling between ports
- Fully asynchronous operation from either port
- Battery back up operation : 2 V data retention

Preview

Interface

Block Diagram

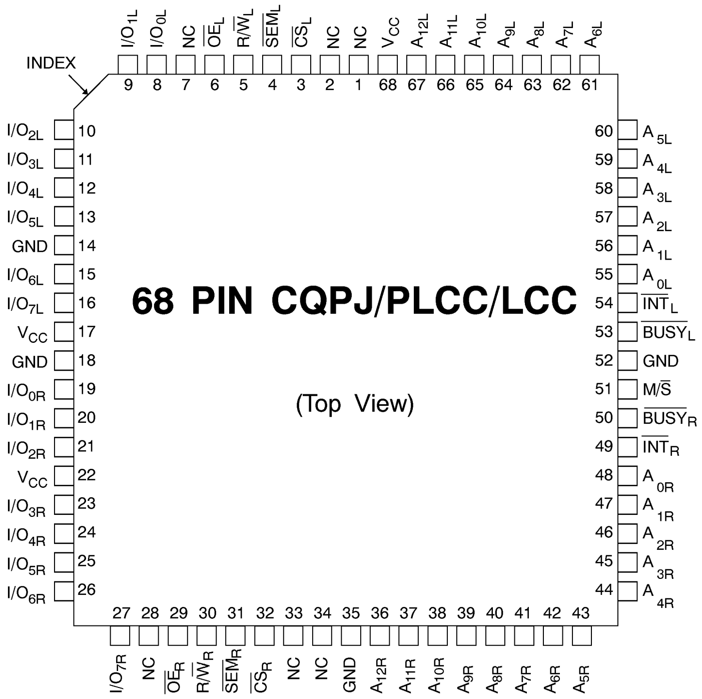


Note : 1. (MASTER) : $\overline{BUSY}$ is output.
(SLAVE) : $\overline{BUSY}$ is input.

Pin Names

LEFT PORT	RIGHT PORT	NAMES
$\overline{CS}_L$	$\overline{CS}_R$	Chip select
$R/\overline{W}_L$	$R/\overline{W}_R$	Read/Write Enable
$\overline{OE}_L$	$\overline{OE}_R$	Output Enable
A_{0L-12L}	A_{0R-12R}	Address
I/O_{0L-7L}	I/O_{0R-7R}	Data Input/Output
$\overline{SEM}_L$	$\overline{SEM}_R$	Semaphore Enable
$\overline{INT}_L$	$\overline{INT}_R$	Interrupt Flag
$\overline{BUSY}_L$	$\overline{BUSY}_R$	Busy Flag
$M/\overline{S}$		Master or Slave Select
Vcc		Power Supply
GND		Ground

Pin Configuration



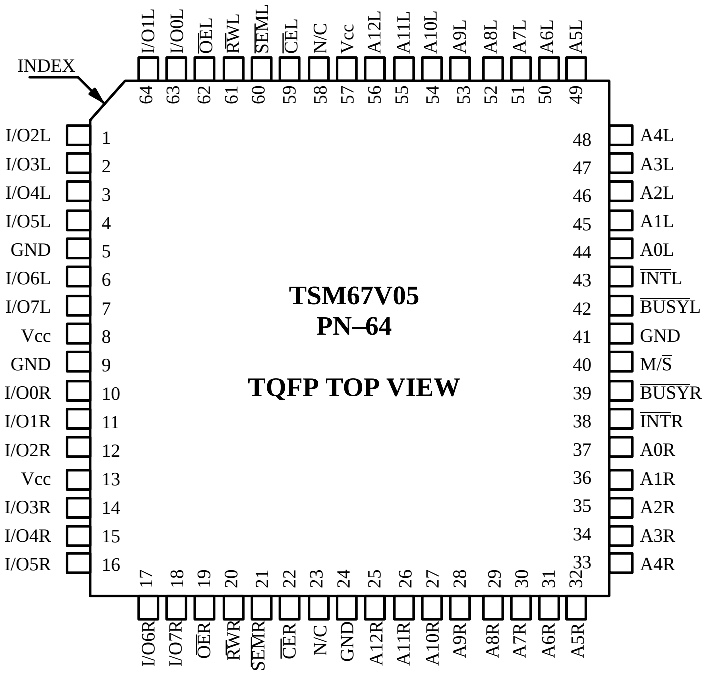
11		51 A_{5L}	50 A_{4L}	48 A_{2L}	46 A_{0L}	44 $\overline{\text{BUSY}}_L$	42 M/$\overline{\text{S}}$	40 $\overline{\text{INT}}_R$	38 A_{1R}	36 A_{3R}	
10	53 A_{7L}	52 A_{6L}	49 A_{3L}	47 A_{1L}	45 $\overline{\text{INT}}_L$	43 GND	41 $\overline{\text{BUSY}}_R$	39 A_{0R}	37 A_{2R}	35 A_{4R}	34 A_{5R}
09	55 A_{9L}	54 A_{8L}								32 A_{7R}	33 A_{6R}
08	57 A_{11L}	56 A_{10L}								30 A_{9R}	31 A_{8R}
07	59 V_{CC}	58 A_{12L}								28 A_{11R}	29 A_{10R}
06	61 NC	60 NC								26 GND	27 A_{12R}
05	63 $\overline{\text{SEM}}_L$	62 $\overline{\text{CE}}_L$								24 NC	25 NC
04	65 $\overline{\text{OE}}_L$	64 $R/\overline{W}_L$								22 $\overline{\text{SEM}}_R$	23 $\overline{\text{CE}}_R$
03	67 I/O_{0L}	66 NC								20 $\overline{\text{OE}}_R$	21 $R/\overline{W}_R$
02	68 I/O_{1L}	1 I/O_{2L}	3 I/O_{4L}	5 GND	7 I/O_{7L}	9 GND	11 I/O_{1R}	13 V_{CC}	15 I/O_{4R}	18 I/O_{7R}	19 NC
01		2 I/O_{3L}	4 I/O_{5L}	6 I/O_{6L}	8 V_{CC}	10 I/O_{0R}	12 I/O_{2R}	14 I/O_{3R}	16 I/O_{5R}	17 I/O_{6R}	

A B C D E F G H I J K L

68 PIN PGA

(Top View)

Notes :
1. All Vcc pins must be connected to power supply.
2. All GND pins must be connected to ground supply.


INDEX
64 I/O1L
63 I/O0L
62 OEL
61 RWL
60 SEML
59 CEL
58 N/C
57 Vcc
56 A12L
55 A11L
54 A10L
53 A9L
52 A8L
51 A7L
50 A6L
49 A5L
I/O2L 1
I/O3L 2
I/O4L 3
I/O5L 4
GND 5
I/O6L 6
I/O7L 7
Vcc 8
GND 9
I/O0R 10
I/O1R 11
I/O2R 12
Vcc 13
I/O3R 14
I/O4R 15
I/O5R 16
TSM67V05
PN–64
TQFP TOP VIEW
48 A4L
47 A3L
46 A2L
45 A1L
44 A0L
43 INTL
42 BUSYL
41 GND
40 M/S
39 BUSYR
38 INTR
37 A0R
36 A1R
35 A2R
34 A3R
33 A4R
17 I/O6R
18 I/O7R
19 OER
20 RWR
21 SEMR
22 CER
23 N/C
24 GND
25 A12R
26 A11R
27 A10R
28 A9R
29 A8R
30 A7R
31 A6R
32 A5R

Functional Description

The TSM67V05 has two ports with separate control, address and I/0 pins that permit independent read/write access to any memory location. These devices have an automatic power-down feature controlled by $\overline{CS}$. $\overline{CS}$ controls on-chip power-down circuitry which causes the port concerned to go into stand-by mode when not selected ($\overline{CS}$ high). When a port is selected access to the full memory array is permitted. Each port has its own Output Enable control ($\overline{OE}$). In read mode, the port's $\overline{OE}$ turns the Output drivers on when set LOW. Non-conflicting READ/WRITE conditions are illustrated in table 1.

The interrupt flag ($\overline{INT}$) allows communication between ports or systems. If the user chooses to use the interrupt function, a memory location (mail box or message center) is assigned to each port. The left port interrupt flag (INT_L) is set when the right port writes to memory location 1FFE (HEX). The left port clears the interrupt by reading address location 1FFE. Similarly, the right port interrupt flag (INT_R) is set when the left port writes to memory location 1FFF (hex), and the right port must read memory location 1FFF in order to clear the interrupt flag (INT_R). The 8 bit message at 1FFE or 1FFF is user-defined. If the interrupt function is not used, address locations 1FFE and 1FFF are not reserved for mail boxes but become part of the RAM. See table 3 for the interrupt function.

Arbitration Logic

The arbitration logic will resolve an address match or a chip select match down to a minimum of 5 ns and determine which port has access. In all cases, an active $\overline{BUSY}$ flag will be set for the inhibited port.

The $\overline{BUSY}$ flags are required when both ports attempt to access the same location simultaneously.Should this conflict arise, on-chip arbitration logic will determine which port has access and set the $\overline{BUSY}$ flag for the inhibited port. $\overline{BUSY}$ is set at speeds that allow the processor to hold the operation with its associated address and data. It should be noted that the operation is invalid for the port for which $\overline{BUSY}$ is set LOW. The inhibited port will be given access when $\overline{BUSY}$ goes inactive.

A conflict will occur when both left and right ports are active and the two addresses coincide. The on-chip arbitration determines access in these circumstances. Two modes of arbitration are provided : (1) if the addresses match and are valid before $\overline{CS}$ on-chip control logic arbitrates between $\overline{CS}_L$ and $\overline{CS}_R$ for access ; or (2) if the $\overline{CS}$s are low before an address match, on-chip control logic arbitrates between the left and right addresses for access (refer to table 4). The inhibited port's $\overline{BUSY}$ flag is set and will reset when the port granted access completes its operation in both arbitration modes.

Data Bus Width Expansion

Master/Slave Description

Expanding the data bus width to 16 or more bits in a dual-port RAM system means that several chips may be active simultaneously. If every chip has a hardware arbitrator, and the addresses for each chip arrive at the same time one chip may activate its L $\overline{BUSY}$ signal while another activates its R $\overline{BUSY}$ signal. Both sides are now busy and the CPUs will wait indefinitely for their port to become free.

To overcome this "Busy Lock-Out" problem, TEMIC has developed a MASTER/SLAVE system which uses a single hardware arbitrator located on the MASTER. The SLAVE has $\overline{BUSY}$ inputs which allow direct interface to the MASTER with no external components, giving a speed advantage over other systems.

When dual-port RAMs are expanded in width, the SLAVE RAMs must be prevented from writing until the $\overline{BUSY}$ input has been settled. Otherwise, the SLAVE chip may begin a write cycle during a conflict situation. On the opposite, the write pulse must extend a hold time beyond $\overline{BUSY}$ to ensure that a write cycle occurs once the conflict is resolved. This timing is inherent in all dual-port memory systems where more than one chip is active at the same time.

The write pulse to the SLAVE must be inhibited by the MASTER's maximum arbitration time. If a conflict then occurs, the write to the SLAVE will be inhibited because of the MASTER's $\overline{BUSY}$ signal.

Semaphore Logic

The TSM67V05 is an externaly fast dual-port 8K x 8 CMOS static RAM with an additional 8 address locations dedicated to binary semaphore flags. These flags allow one of the processors on the left or right side of the dual-port RAM to claim priority over the other for functions defined by the system software. For example, the semaphore flag can be used by one processor to inhibit the other from accessing a portion of the dual-port RAM or any other shared resource.

The dual-port RAM has a fast access time, and the two ports are completely independent of one another. This means that the activity on the left port cannot slow the access time of the right port. The ports are identical in function to standard CMOS static RAMs an can be read from, or written to, at the same time with the only possible conflict arising from simultaneous writing to, or a simultaneous READ/WRITE operation on, a non-semaphore location. Semaphores are protected against such ambiguous situations and may be used by the system program to prevent conflicts in the non-semaphore segment of the dual-port RAM. The devices have an automatic power-down feature controlled by $\overline{CS}$, the dual-port RAM select and $\overline{SEM}$, the semaphore enable. The $\overline{CS}$ and $\overline{SEM}$ pins control on-chip-power-down circuitry that permits the port concerned to go into stand-by mode when not selected. These conditions are shown in table 1 where $\overline{CS}$ and $\overline{SEM}$ are both high.

Systems most suitable to use the TSM67V05 are based upon multiple processors or controllers and are typically very high-speed, software controlled or software-intensive systems. These systems can benefit from the performance enhancement offered by the TSM67V05 hardware semaphores, which provide a lock-out mechanism without the need for complex programming.

Software handshaking between processors offers the maximum level of system flexibility by permitting shared resources to be allocated in varying configurations. The TSM67V05 does not use its semaphore flags to control any resources through hardware, thus allowing the system designer total flexibility in system architecture.

An advantage of using sempahores rather than the more usual methods of hardware arbitration is that neither processor ever incurs wait states. This can prove a considerable advantage in very high speed systems.

How the Semaphore Flags Work

The semaphore logic is a set of eight latches independent of the dual-port RAM. These latches can be used to pass a flag or token, from one port to the other to indicate that a shared resource is in use. The semaphores provides the hardware context with the “Token Passing Allocation” method of use assignment. This method uses the state of a semaphore latch as a token indicating that a shared resource is in use. If the left processor needs to use a resource, it requests the token by setting the latch. The processor then verifies that the latch has been set by reading it. If the latch has been set the processor assumes control over the shared resource. If the latch has not been set, the left processor has established that the right processor had set the latch first, has the token and is using the shared resource. The left processor may then either repeatedly query the status of the semaphore, or abandon its request for the token and perform another operation whilst occasionally attempting to gain control of the token through a set and test operation. Once the right side has relinquished the token the left side will be able to take control of the shared resource.

The semaphore flags are active low. A token is requested by writing a zero to a semaphore latch, and is relinquished again when the same side writes a one to the latch.

The eight semaphore flags are located in a separate memory space from the dual-port RAM in the TSM67V05. The address space is accessed by placing a low input on the $\overline{SEM}$ pin (which acts as a chip select for the semaphore flags) and using the other control pins (Address, $\overline{OE}$ and R/$\overline{W}$) as normally used in accessing a standard static RAM. Each of the flags has a unique address accessed by either side through address pins A0-A2. None of the other address pins has any effect when accessing the semaphores. Only data D_O is used when writing to a semaphore. If a low level is written to an unused semaphore location, the flag will be set to zero on that side and to one on the other side (see table 5). The semaphore can now only be modified by the side showing the zero. Once a one is written to this location from tha same side, the flag will be set to one for both sides (unless a request is pending from the other side) and the semaphore can then be written to by either side.

The effect the side writing a zero to a semaphore location has a locking out the other side is the reason for the use of semaphore logic in interprocessor communication. (A thorough discussion of the use of this feature follows below). A zero written to the semaphore location from the locked-out side will be stored in the semaphore request latch for that side until the semaphore is relinquished by the side having control.

When a semaphore flag is read its value is distributed to all data bits so that a flag set at one reads as one in all data bits and a flag set at zero reads as all zeros. The read value is latched into the output register of one side when its sempahore select ($\overline{SEM}$) and output enable ($\overline{OE}$) signals go active. This prevents the semaphore changing state in the middle of a read cycle as a result of a write cycle issued by the other side. Because of this latch, a repeated read of a semaphore flag in a test loop must cause either signal ($\overline{SEM}$ or $\overline{OE}$) to go inactive, otherwise the output will never change.

The semaphore must use a WRITE/READ sequence in order to ensure that no system level conflict will occur. A processor requests access to shared resources by attempting to write a zero to a semaphore location. If the semaphore is already in use, the semaphore request latch will contain a zero, yet the semaphore flag will appear as a one, and the processor will detect this status in the subsequent read (see table 5). For example, assume a processor writes a zero to the left port at a free semaphore location. On a subsequent read, the processor will verify that it has written succesfully to that location and will assume control over the resource concerned. If a processor on the right side then attempts to write a zero to the same semaphore flag it will fail, as will be verified by a subsequent read returning a one from the semaphore location on the right side had a READ/WRITE sequence been used instead, system conflict problems could occurred during the interval between the read and write cycles.

It must be noted that a failed semaphore request needs to be followed by either repeated reads or by writing a one to the same location. The simple logic diagram for the semaphore flag in figure 2 illustrates the reason for this quite clearly. Two semaphore request latches deed into a semaphore flag. The first latch to send a zero to the semaphore flag will force its side of the semaphore flag low and other side high. This status will be maintained until a one is written to the same semaphore request latch. Should a zero be written to the other side's semaphore request latch in the meantime, the semaphore flag will flip over to this second side as soon as a one is written to the first side's request latch. The second side's flag will now stay low until its semaphore request latch is changed to a one. Thus, clearly, if a semaphore flag is requested and the processor requesting it no longer requires access to the resource, the entire system can hang up until a one is written to the semaphore request latch concerned.

Semaphore timing becomes critical when both sides request the same token by attempting to write a zero to it at the same time. Semaphore logic is specially conceived to resolve this problem. The logic ensures that only one side will receive the token if simultaneous requests are made. The first side to make a request will receive the token where request do not arrive at the same time. Where they do arrive at the same time, the logic will assign the token arbitrarily to one of the ports.

It should be noted, however, that semaphores alone do not guarantee that access to a resource is secure. As with any powerful programming technique, errors can be introduced if semaphores are misused or misinterpreted. Code integrity is of the utmost performance when semaphores are being used instead of slower, more restrictive hardware-intensive systems.

Semaphore initialization is not automatic and must therefore be incorporated in the power up initialization procedures. Since any semaphore flag containing a zero must be reset to one, initialization should write a one to all request flags from both sides to ensure that they will be available when required.

Using Semaphores – some examples

Perhaps the simplest application of semaphores is their use as resource markers for the TSM67V05's dual-port RAM. If it is necessary to split the 8 k × 8 RAM into two 4 k × 8 blocks which are to be dedicated to serving either the left or right port at any one time. Semaphore 0 can be used to indicate which side is controlling the lower segment of memory and semaphore 1 can be defined as indicating the upper segment of memory.

To take control of a ressource, in this case the lower 4 k of a dual-port RAM, the left port processor would then write a zero into semaphore flag 0 and then read it back. If succesful in taking the token (reading back a zero rather than a one), the left processor could then take control of the lower 4 k of RAM. If the right processor attempts to perform the same function to take control of the resource after the left processor has already done so, it will read back a one in response to the attempted write of a zero into semaphore 0. At this point the software may choose to attempt to gain control on the second 4 k segment of RAM by writing and then reading a zero in semaphore 1. If successful, it will lock out the left processor.

Once the left side has completed its task it will write a one to semaphore 0 and may then attempt to access semaphore 1. If semaphore 1 is still occupied by the right side, the left side may abandon its semaphore request and perform other operations until it is able to write and then read a zero in semaphore 1. If the right processor performs the same operation with semaphore 0, this protocol would then allow the two processes to swap 4 k blocks of dual-port RAM between one another.

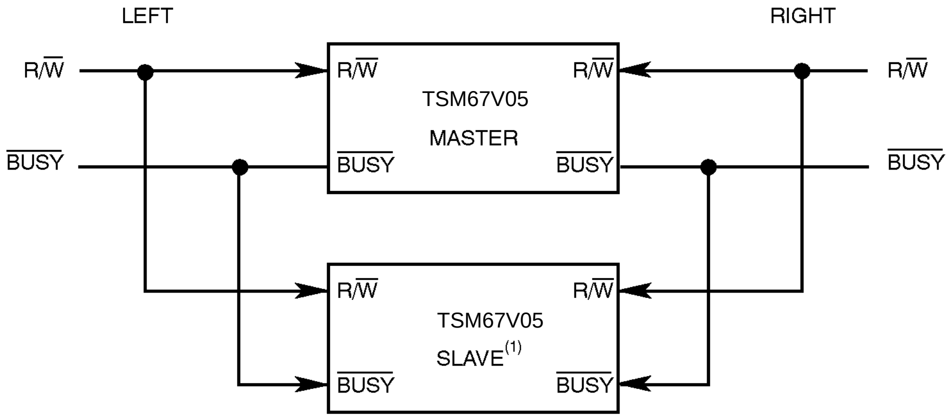
The blocks do not have to be any particular size, and may even be of variable size depending on the complexity of the software using the semaphore flags. All eight semaphores could be used to divide the dual-port RAM or other shared resources into eight parts. Semaphores can even be assigned different meanings on each side, rather than having a common meaning as is described in the above example.

Semaphores are a useful form of arbitration in systems such as disk interfaces where the CPU must be locked out of a segment of memory during a data transfer operation, and the I/0 devices tolerate any wait states. If semaphores are used, both the CPU and the I/0 device can access assigned memory segments, cannot without the need for wait states, once the two devices have determined which memory area is barred to the CPU.

Semaphores are also useful in applications where no memory WAIT state is available on one or both sides. Once a semaphore handshake has been performed, both processors can access their assigned RAM segments at full speed.

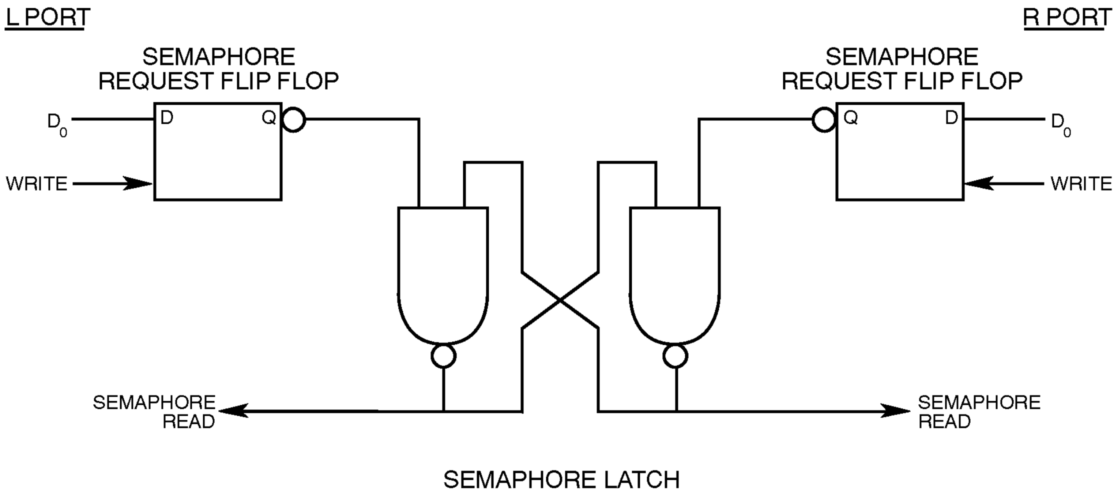
Another application is in complex data structures. Block arbitration is very important in this case, since one processor may be responsible for building and updating a data structure whilst the other processor reads and interprets it. A major error condition may be created if the interpreting processor reads an incomplete data structure. Some sort of arbitration between the two different processors is therefore necessary. The building processor request access to the block, locks it and is then able to enter the block to update the data structure. Once the update is completed the data structure may be released. This allows the interpreting processor, to return to read the complete data structure, thus ensuring a consistent data structure.

Figure 1. 16-Bit Master/Slave Dual-Port Memory Systems.



Note : 1. No arbitration in TSM67V05 (SLAVE). **$\overline{\text{BUSY}}$**-IN inhibits write in TSM67V05 SLAVE.

Figure 2. TSM67V05 – Semaphore Logic.



Preview

Truth Table

Table 1 : Non Contention Read/Write Control.

INPUTS [1]				OUTPUTS	MODE
$\overline{CS}$	$R/\overline{W}$	OE	$\overline{SEM}$	$I/O_{\varnothing}$–I/O_7	
H	X	X	H	Hi–Z	Deselect Power Down
H	H	L	L	$DATA_{OUT}$	Read Data In Sema. Flag
X	X	H	X	Hi–Z	Outputs Disabled
H	↑	X	L	$DATA_{IN}$	Write DIN∅ Into Sema. Flag
L	H	L	H	$DATA_{OUT}$	Read Memory
L	L	X	H	$DATA_{IN}$	Write to Memory
L	X	X	L	—	Not Allowed

Note : 1. $A_{OL} – A_{12L} \neq A_{0R} – A_{12R}$.

Table 2 : Arbitration Options.

OPTION	INPUTS			OUTPUTS	
	$\overline{CS}$	$M/\overline{S}$	$\overline{SEM}$	$\overline{BUSY}$	$\overline{INT}$
Busy Logic Master	L	H	H	Output signal	—
Busy Logic Slave	L	L	H	Input Signal	—
Interrupt Logic	L	X	H	—	Output signal
Semaphore Logic*	H	H	L	H	—
	H	L	L	Hi–Z	—

* Inputs Signals are for Semaphore Flags set and test (Write and read) operations.

Table 3 : Interrupt Flag.

LEFT PORT					RIGHT PORT					FUNCTION
$R/\overline{W}_L$	$\overline{CS}_L$	$\overline{OE}_L$	$A_{OL}–A_{12L}$	$\overline{INT}_L$	$R/\overline{W}_R$	$\overline{CS}_R$	$\overline{OE}_R$	$A_{OR}–A_{12R}$	$\overline{INT}_R$	
L	L	X	1FFF	X	X	X	X	X	L[2]	Set Right $\overline{INT}_R$ Flag
X	X	X	X	X	X	L	L	1FFF	H[3]	Reset Right $\overline{INT}_R$ Flag
X	X	X	X	L[3]	L	L	X	1FFE	X	Set Left $\overline{INT}_L$ Flag
X	L	L	1FFE	H[2]	X	X	X	X	X	Reset Left $\overline{INT}_L$ Flag

Notes :

1. Assumes $\overline{BUSY}_L = \overline{BUSY}_R$ = H.
2. If $\overline{BUSY}_L$ = L, then NC.
3. If $\overline{BUSY}_R$ = L, then NC.
4. H = HIGH, L = LOW, X = DON'T CARE, NC = NO CHANGE.

Table 4 : Arbitration [2]

LEFT PORT		RIGHT PORT		FLAGS (1)		FUNCTION
$\overline{CS}_L$	$A_{0L} - A_{12L}$	$\overline{CS}_R$	$A_{0R} - A_{12R}$	$\overline{BUSY}_L$	$\overline{BUSY}_R$	
H	X	H	X	H	H	No Contention
L	Any	H	X	H	H	No Contention
H	X	L	Any	H	H	No Contention
L	$\neq A_{0R} - A_{12R}$	L	$\neq A_{0L} - A_{12L}$	H	H	No Contention
ADDRESS ARBITRATION WITH $\overline{CE}$ LOW BEFORE ADDRESS MATCH						
L	LV5R	L	LV5R	H	L	L–Port Wins
L	RV5L	L	RV5L	L	H	R–Port Wins
L	Same	L	Same	H	L	Arbitration Resolved
L	Same	L	Same	L	H	Arbitration Resolved
$\overline{CE}$ ARBITRATION WITH ADDRESS MATCH BEFORE $\overline{CS}$						
LL5R	$= A_{0R} - A_{12R}$	LL5R	$= A_{0L} - A_{12L}$	H	L	L–Port Wins
RL5L	$= A_{0R} - A_{12R}$	RL5L	$= A_{0L - A12L}$	L	H	R–Port Wins
LW5R	$= A_{0R - A12R}$	LW5R	$= A_{0L} - A_{12L}$	H	L	Arbitration Resolved
LW5R	$= A_{0R} - A_{12R}$	LW5R	$= A_{0L} - A_{12L}$	L	H	Arbitration Resolved

Notes :
1. $\overline{INT}$ Flags Don’t Care.
2. X = DON’T CARE, L = LOW, H = HIGH.
LV5R = Left Address Valid ≥ 5 ns before right address.
RV5L = Right Address Valid ≥ 5 ns before left address.
Same = Left and Right Addresses match within 5 ns of each other.
LL5R = Left $\overline{CS}$ = LOW ≥ 5 ns before Right $\overline{CS}$.
RL5L = Right $\overline{CS}$ = LOW ≥ 5 ns before left $\overline{CS}$.
LW5R = Left and Right $\overline{CS}$ = LOW within 5 ns of each other.

Table 5 : Example Semaphore Procurement Sequence.

FUNCTION	$D_0 - D_7$ LEFT	$D_0 - D_7$ RIGHT	STATUS
No Action	1	1	Semaphore free
Left Port Writes ”0” to Semaphore	0	1	Left Port has semaphore token
Right Port Writes ”0” to Semaphore	0	1	No change. Right side has no write access to semaphore
Left Port Writes ”1” to Semaphore	1	0	Right port obtains semaphore token
Left Port Writes ”0” to Semaphore	1	0	No change. Left port has no write access to semaphore
Right Port Writes ”1” to Semaphore	0	1	Left port obtains semaphore token
Left Port Writes ”1” to Semaphore	1	1	Semaphore free
Right Port Writes ”0” to Semaphore	1	0	Right port has semaphore token
Right Port Writes ”1” to Semaphore	1	1	Semaphore free
Left Port Writes ”0” to Semaphore	0	1	Left Port has semaphore token
Left Port Writes ”1” to Semaphore	1	1	Semaphore free

Note : 1. This table denotes a sequence of events for only one of the 8 semaphores on the TSM67V05.

Electrical Characteristics

Absolute Maximum Ratings

Supply voltage (VCC-GND) : . . . . -0.3 V to 7.0 V
Input or output voltage applied : . . . . (GND – 0.3 V) to
(VCC + 0.3 V)
Storage temperature : . . . . –65 °C to + 150 °C

**** Notice***
Stresses greater than those listed under ABSOLUTE MAXIMUM RATINGS may cause permanent damage to the device.This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extented periods may affect reliability.

OPERATING RANGE	OPERATING SUPPLY VOLTAGE	OPERATING TEMPERATURE
Military	V_{CC} = 3.3 V ± 0.3 V	– 55 ºC to + 125 ºC
Automotive	V_{CC} = 3.3 V ± 0.3 V	– 40 ºC to + 125 ºC
Industrial	V_{CC} = 3.3 V ± 0.3 V	– 40 ºC to + 85 ºC
Commercial	V_{CC} = 3.3 V ± 0.3 V	0 ºC to + 70 ºC

DC Parameters

Parameter	Description	Version	TSM67V05–35 (*) COM	TSM67V05–45		TSM67V05–55		UNIT	VALUE
				COM	IND MIL AUTO	COM	IND MIL AUTO		
$I_{CCSB\ (1)}$	Standby supply current (Both ports TTL level inputs)	V	5	5	5	5	5	mA	Max
		L	10	10	10	10	10	mA	Max
$I_{CCSB1\ (2)}$	Standby supply current (Both ports CMOS level inputs)	V	100	100	150	100	150	µA	Max
		L	1000	1000	1500	1000	1500	µA	Max
$I_{CCOP\ (3)}$	Operating supply current (Both ports active)	V	100	100	110	95	100	mA	Max
		L	105	105	120	100	105	mA	Max
$I_{CCOP1\ (4)}$	Operating supply current (One port active – One port standby)	V	55	55	60	50	55	mA	Max
		L	60	60	65	55	60	mA	Max

Notes :
1. $CS_L = CS_R \geq 2.2$ V – $\overline{SEM}_R = \overline{SEM}_L \geq 2.2$ V.
2. $CS_L = CS_R \geq$ VCC – 0.2 V – $\overline{SEM}_R = \overline{SEM}_L \geq 2.2$ V – F = 0.
3. Both ports active – Maximum frequency – Outputs open – $\overline{OE}$ = VIH – $\overline{CS} \leq$ VIL – $\overline{SEM} \geq 2.2$ V.
4. One port active (f = fMAX) – Output open –
One port stand-by TTL or CMOS Level Inputs – $\overline{CS}_L = \overline{CS}_R \geq 2.2$ V – $\overline{SEM}_R = \overline{SEM}_L \geq$ Vcc – 0.2 V.

(*). Commercial only.

PARAMETER	DESCRIPTION	TSM67V05–35/45/55	UNIT	VALUE
$II/O_{(5)}$	Input/Output leakage current	± 5	µA	Max
$VIL_{(6)}$	Input low voltage	0.8	V	Max
$VIH_{(6)}$	Input high voltage	2.0	V	Min
$VOL_{(7)}$	Output low voltage (I/O_0–I/O_{15})	0.4	V	Max
$VOH_{(7)}$	Output high voltage	2.4	V	Min
$C\ IN_{(8)}$	Input capacitance	5	pF	Max
$C\ OUT_{(8)}$	Output capacitance	7	pF	Max

Notes :
5. Vcc = 5.5 V, Vin = Gnd to Vcc, $\overline{CS}$ = VIH, Vout = 0 to Vcc.
6. VIH max = Vcc + 0.3 V, VIL min = –0.3 V or –1 V pulse width 50 ns.
7. Vcc min, IOL = 4 mA, IOH = –4 mA.
8. Guaranteed but not tested.

Data-Retention Mode

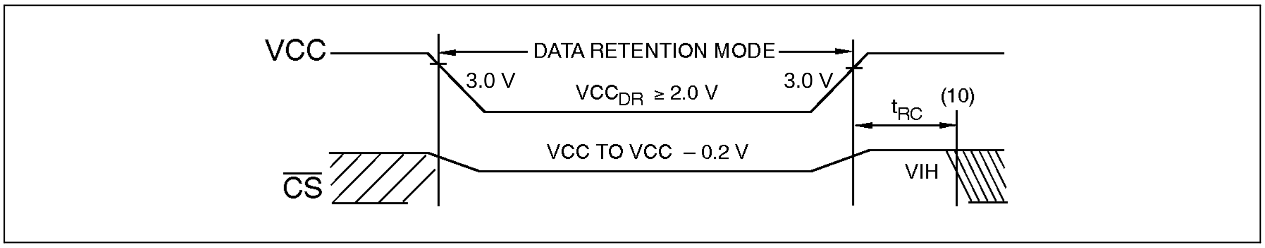
TEMIC CMOS RAMs are designed to work with a battery backup. Data retention voltage and supply current are guaranteed over temperature. The following rules insure data retention :

1 – Chip select ($\overline{CS}$) must be held high during data retention ; within Vcc to VCC – 0.2 V.

2 – $\overline{CS}$ must be kept between V_{CC} – 0.2 V and 70 % of Vcc during the power up and power down transitions.

3 – The RAM can begin operation > tRC after Vcc reaches the minimum operating voltage (3 volts).

Timing



PARAMETER	TEST CONDITIONS (9)		MAX		UNIT
			COM	MIL IND AUTO	
ICC_{DR1}	@ VCC_{DR} = 2 V	V L	75 500	100 750	µA

Notes : 9. $\overline{CS}$ = V_{CC}, Vin = Gnd to V_{CC}.
10. t_{RC} = Read cycle time.

AC Test Conditions

Input Pulse Levels	: GND to 3.0 V
Input Rise/Fall Times	: 5 ns
Input Timing Reference Levels	: 1.5 V
Output Reference Levels	: 1.5 V
Output Load	: see figures 3, 4

Figure 3. AC Output Test Load.

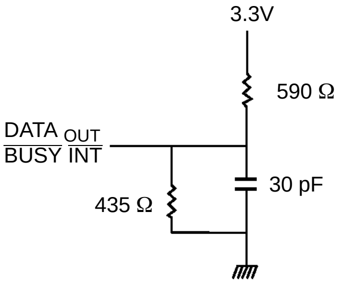
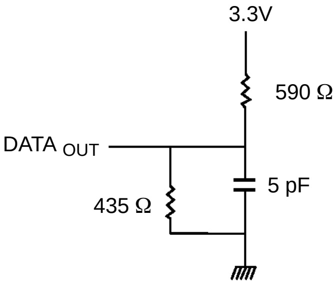


Figure 4. Output Load (for t_{HZ}, t_{LZ}, t_{WZ}, and t_{OW}). Including scope and jig.



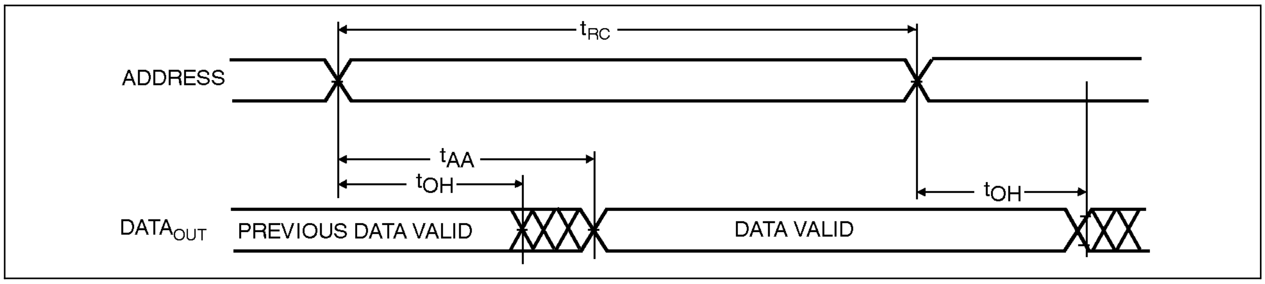
AC Parameters

READ CYCLE		PARAMETER	TSM67V05–35		TSM67V05–45		TSM67V05–55		UNIT
SYMBOL (4)	SYMBOL (5)		MIN.	MAX.	MIN.	MAX.	MIN.	MAX.	
TAVAVR	t_{RC}	Read cycle time	35	–	45	–	55	–	ns
TAVQV	t_{AA}	Address access time	–	35	–	45	–	55	ns
TELQV	t_{ACS}	Chip Select access time (3)	–	35	–	45	–	55	ns
TBLQV	t_{ABE}	Byte enable access time (3)	–	35	–	45	–	55	ns
TGLQV	t_{AOE}	Output enable access time	–	20	–	25	–	30	ns
TAVQX	t_{OH}	Output hold from address change	3	–	5	–	3	–	ns
TELQZ	t_{LZ}	Output low Z time (1, 2)	3	–	5	–	3	–	ns
TEHQZ	t_{HZ}	Output high Z time (1, 2)	–	20	–	20	–	25	ns
TPU	t_{PU}	Chip Select to power up time (2)	0	–	0	–	0	–	ns
TPD	t_{PD}	Chip disable to power down time (2)	–	35	–	35	–	50	ns
TSAA	t_{SAVQV}	Semaphore Address Access Time	–	35	–	45	–	55	ns
TSOP	t_{SOP}	SEM flag update pulse ($\overline{OE}$ or $\overline{SEM}$)	15	–	15	–	15	–	ns

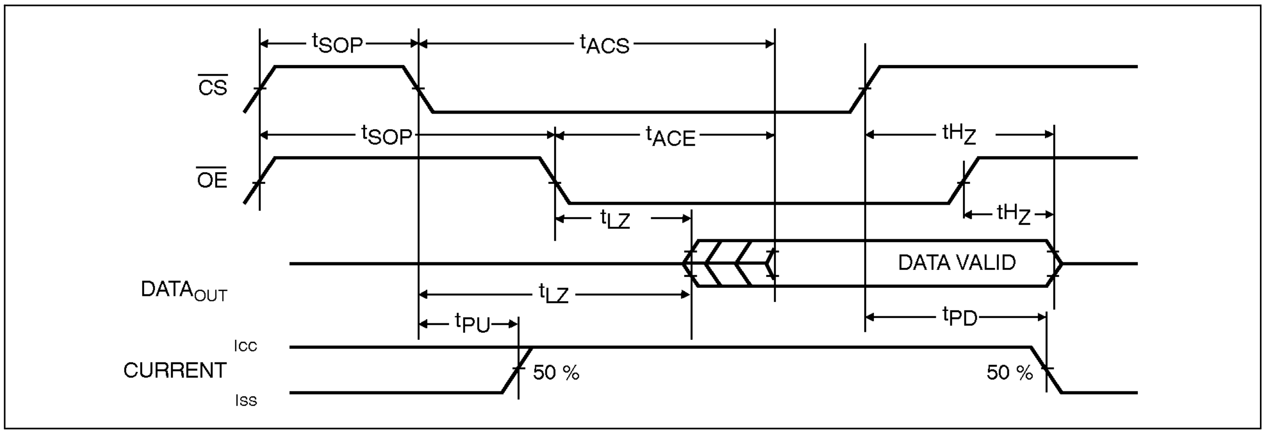
Notes :
1. Transition is measured ± 500 mV from low or high impedance voltage with load (figures 1 and 2).
2. This parameter is guaranteed but not tested.
3. To access RAM $\overline{CS} = V_{IL}$, $\overline{SEM} = V_{IH}$. To access semaphore $\overline{CS} = V_{IH}$, $\overline{SEM} = V_{IL}$. Refer to table 1.
4. STD symbol.
5. ALT symbol.

(*). Commercial only.

Timing Waveform of Read Cycle n° 1, Either Side (1, 2, 4)



Timing Waveform of Read Cycle n° 2, Either Side



AC Parameters

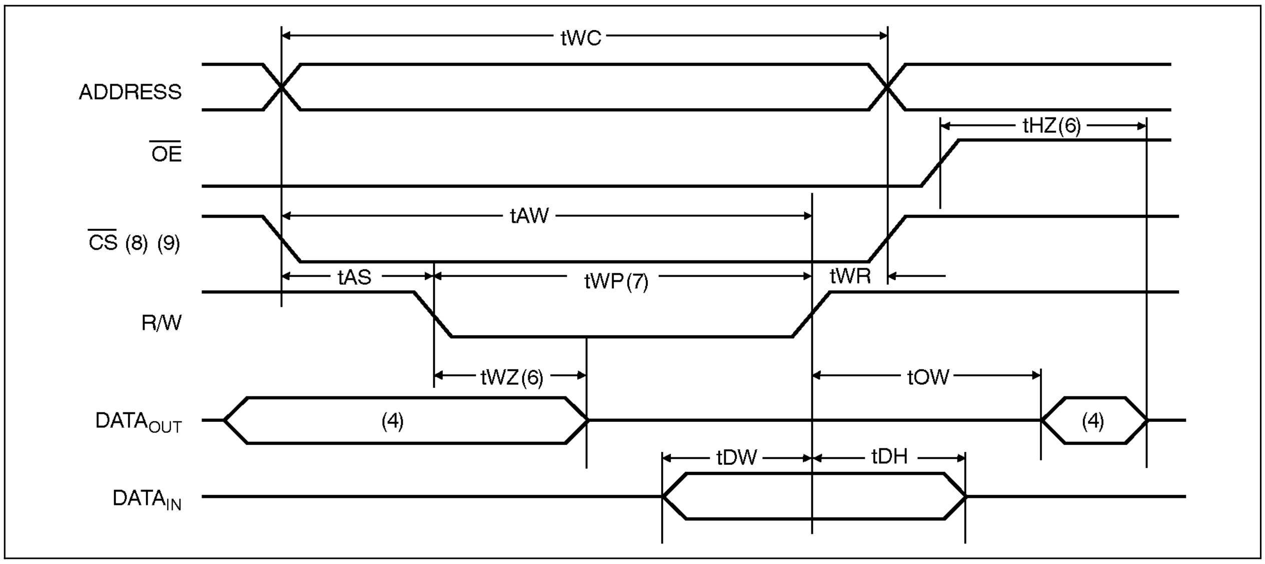
WRITE CYCLE		PARAMETER	TSM67V05–35(*)		TSM67V05–45		TSM67V05–55		UNIT
SYMBOL (5)	SYMBOL (6)		MIN.	MAX.	MIN.	MAX.	MIN.	MAX.	
TAVAVW	t_{WC}	Write cycle time	35	–	45	–	55	–	ns
TELWH	t_{EW}	Chip select to end of write (3)	30	–	40	–	45	–	ns
TAVWH	t_{AW}	Address valid to end of write	30	–	40	–	45	–	ns
TAVWL	t_{AS}	Address Set–up Time	0	–	0	–	0	–	ns
TWLWH	t_{WP}	Write Pulse Width	25	–	30	–	40	–	ns
TWHAX	t_{WR}	Write Recovery Time	0	–	0	–	0	–	ns
TDVWH	t_{DW}	Data Valid to end of write	20	–	30	–	30	–	ns
TGHQZ	t_{HZ}	Output high Z time (1, 2)	–	20	–	20	–	25	ns
TWHDX	t_{DH}	Data hold time (4)	0	–	0	–	0	–	ns
TWLQZ	t_{WZ}	Write enable to output in high Z (1, 2)	–	20	–	20	–	25	ns
TWHQX	t_{OW}	Output active from end of write (1, 2, 4)	0	–	0	–	0	–	ns
TSWRD	t_{SWRD}	SEM flag write to read time	5	–	5	–	5	–	ns
TSPS	t_{SPS}	SEM flag contention window	5	–	5	–	5	–	ns

Notes :

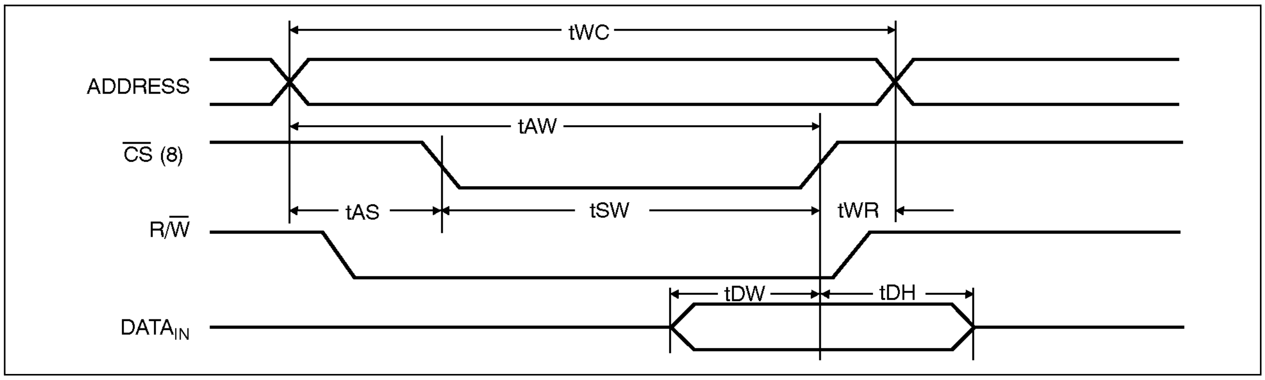
1. Transition is measured ± 500 mV from low or high impedance voltage with load (figures 1 and 2).
2. This parameter is guaranteed but not tested.
3. To access RAM $\overline{CS}$ = V_{IL}, $\overline{SEM}$ = V_{IH}. To access semaphore $\overline{CS}$ = V_{IH}, $\overline{SEM}$ = V_{IL}. This condition must be valid for entire t_{SW} time.
4. The specification for t_{DH} must be met by the device supplying write data to the RAM under all operating conditions. Although t_{DH} and t_{OW} values vary over voltage and temperature, the actual t_{DH} will always be smaller than the actual t_{OW}.
5. STD symbol.
6. ALT symbol.

(*). Commercial only.

Timing Waveform of Write Cycle n° 1, R/$\overline{W}$ Controlled Timing (1, 2, 3, 7)



Timing Waveform of Write Cycle n° 2, $\overline{CS}$ Controlled Timing (1, 2, 3, 5)



Notes :

1. R/$\overline{W}$ must be high during all address transitions.
2. A write occurs during the overlap (t_{SW} or t_{WP}) of a low $\overline{CS}$ or $\overline{SEM}$ and a low R/$\overline{W}$.
3. t_{WR} is measured from the earlier of $\overline{CS}$ or R/$\overline{W}$ (or $\overline{SEM}$ or R/$\overline{W}$) going high to the end of write cycle.
4. During this period, the I/O pins are in the output state, and input signals must not be applied.
5. If the $\overline{CS}$ or $\overline{SEM}$ low transition occurs simultaneously with or after the R/$\overline{W}$ low transition, the outputs remain in the high impedance state.
6. Transition is measured ± 500 mV from steady state with a 5pF load (including scope and jig).This parameter is sampled and not 100 % tested.
7. If $\overline{OE}$ is low during a R/$\overline{W}$ controlled write cycle, the write pulse width must be the larger of t_{WP} or (t_{WZ} + t_{DW}) to allow the I/O drivers to turn off and data to be placed on the bus for the required t_{DW}. If $\overline{OE}$ is high during an R/$\overline{W}$ controlled write cycle, this requirement does not apply and the write pulse can be as short as the specified t_{WP}.
8. To access RAM, $\overline{CS}$ = V_{IL}. $\overline{SEM}$ = V_{IH}.

AC Parameters

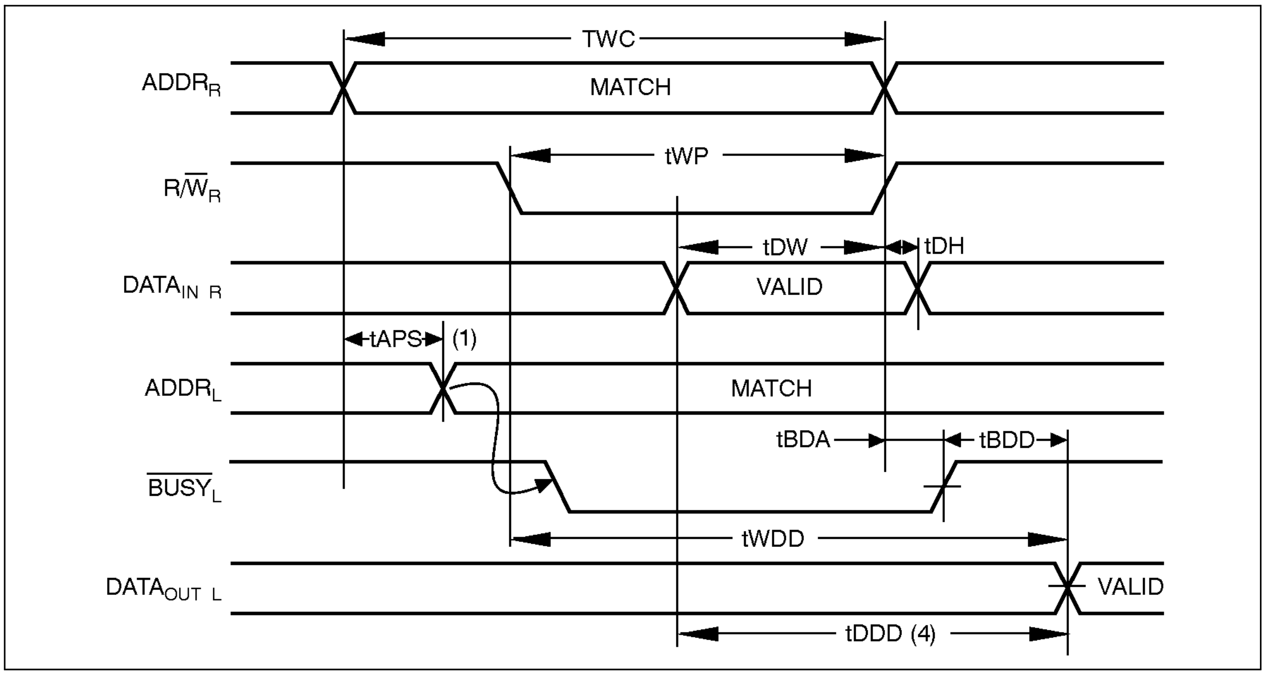
SYMBOL	PARAMETER	TSM67V05–35(*)		TSM67V05–45		TSM67V05–55		UNIT
		MIN.	MAX.	MIN.	MAX.	MIN.	MAX.	
BUSY TIMING (For Master TSM67V05 only)								
t_{BAA}	$\overline{\text{BUSY}}$ Access time to address	–	35	–	40	–	45	ns
t_{BDA}	$\overline{\text{BUSY}}$ Disable time to address	–	35	–	40	–	45	ns
t_{BAC}	$\overline{\text{BUSY}}$ Access time to Chip Select	–	35	–	40	–	45	ns
t_{BDC}	$\overline{\text{BUSY}}$ Disable time to Chip Select	–	35	–	40	–	45	ns
t_{WDD}	Write Pulse to data delay (1)	–	65	–	75	–	85	ns
t_{DDD}	Write data valid to read data delay (1)	–	60	–	70	–	80	ns
t_{APS}	Arbitration priority set–up time (2)	5	–	5	–	5	–	ns
t_{BDD}	$\overline{\text{BUSY}}$ disable to valid data	–	Note 3	–	Note 3	–	Note 3	ns
BUSY TIMING (For Slave TSM67V05 only)								
t_{WB}	Write to $\overline{\text{BUSY}}$ input (4)	0	–	0	–	0	–	ns
t_{WH}	Write hold after $\overline{\text{BUSY}}$ (5)	25	–	25	–	25	–	ns
t_{WDD}	Write pulse to data delay (6)	–	65	–	75	–	85	ns
t_{DDD}	Write data valid to read data delay (6)	–	60	–	70	–	80	ns

Notes :

1. Port-to-port delay through RAM cells from writing port to reading port, refer to “Timing Waveform of Read with $\overline{\text{BUSY}}$ (For Master TSM67V05 only)”.
2. To ensure that the earlier of the two ports wins.
3. t_{BDD} is a calculated parameter and is the greater of 0, $t_{WDD} - t_{WP}$ (actual) or $t_{DDD} - t_{DW}$ (actual).
4. To ensure that the write cycle is inhibited during contention.
5. To ensure that a write cycle is completed after contention.
6. Port-to-port delay through RAM cells from writing port to reading port, refer to “Timing Waveforms of Read with Port to port delay (For Slave TSM67V05 only)”.

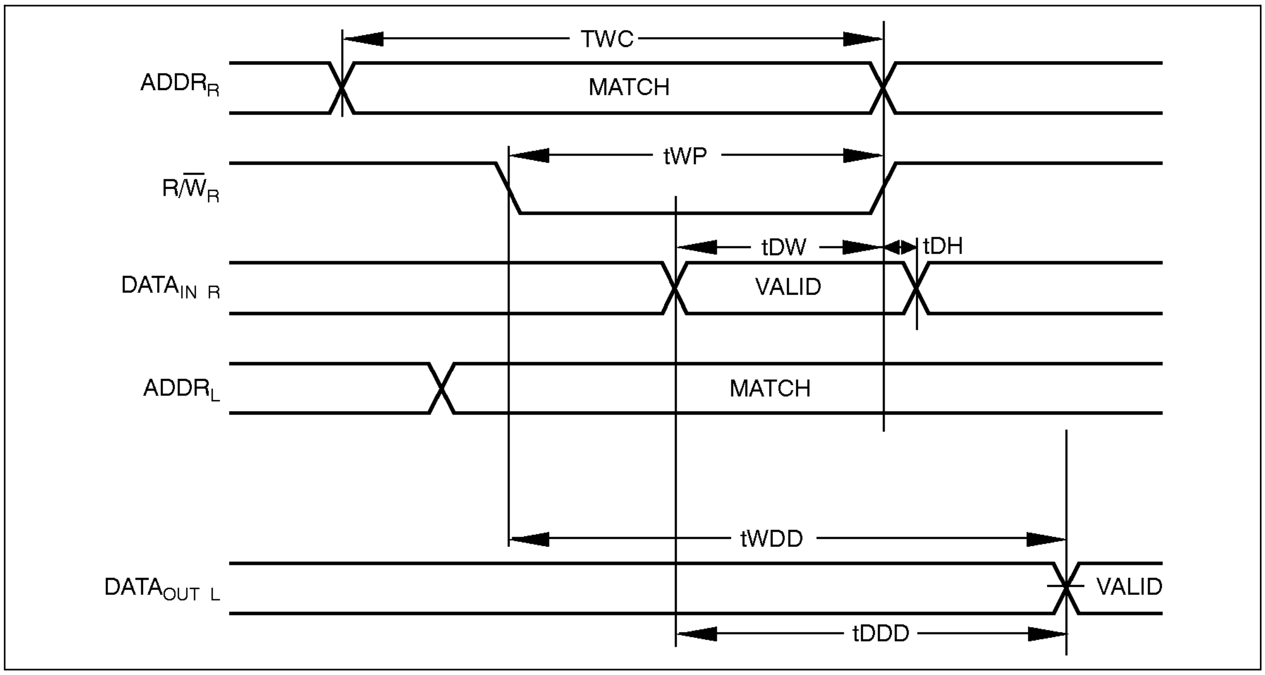
(*). Commercial only.

Timing Waveform of Read with $\overline{\text{BUSY}}$ (2, 3) (For Master TSM67V05)



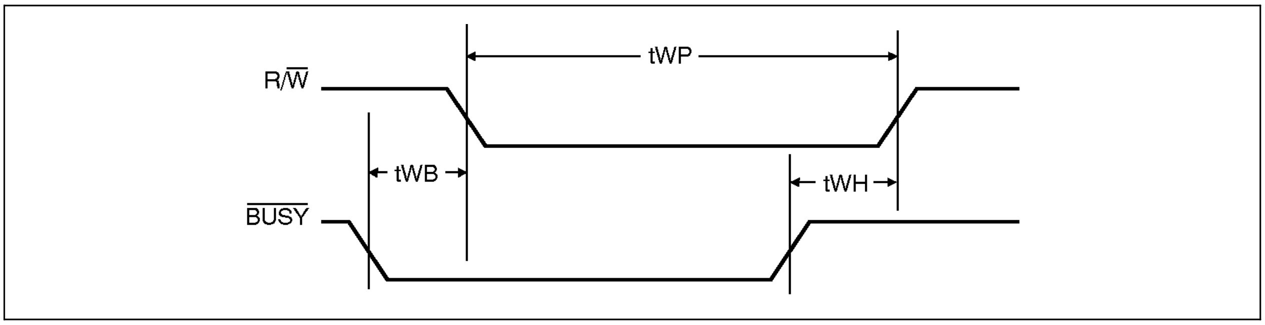
Notes :
1. To ensure that the earlier of the two port wins.
2. Write cycle parameters should be adhered to, to ensure proper writing.
3. Device is continuously enabled for both ports.
4. $\overline{\text{OE}}$ = L for the reading port.

Timing Waveform of Write with Port-to-Port (1, 2, 3) (For Slave TSM67V05 Only)

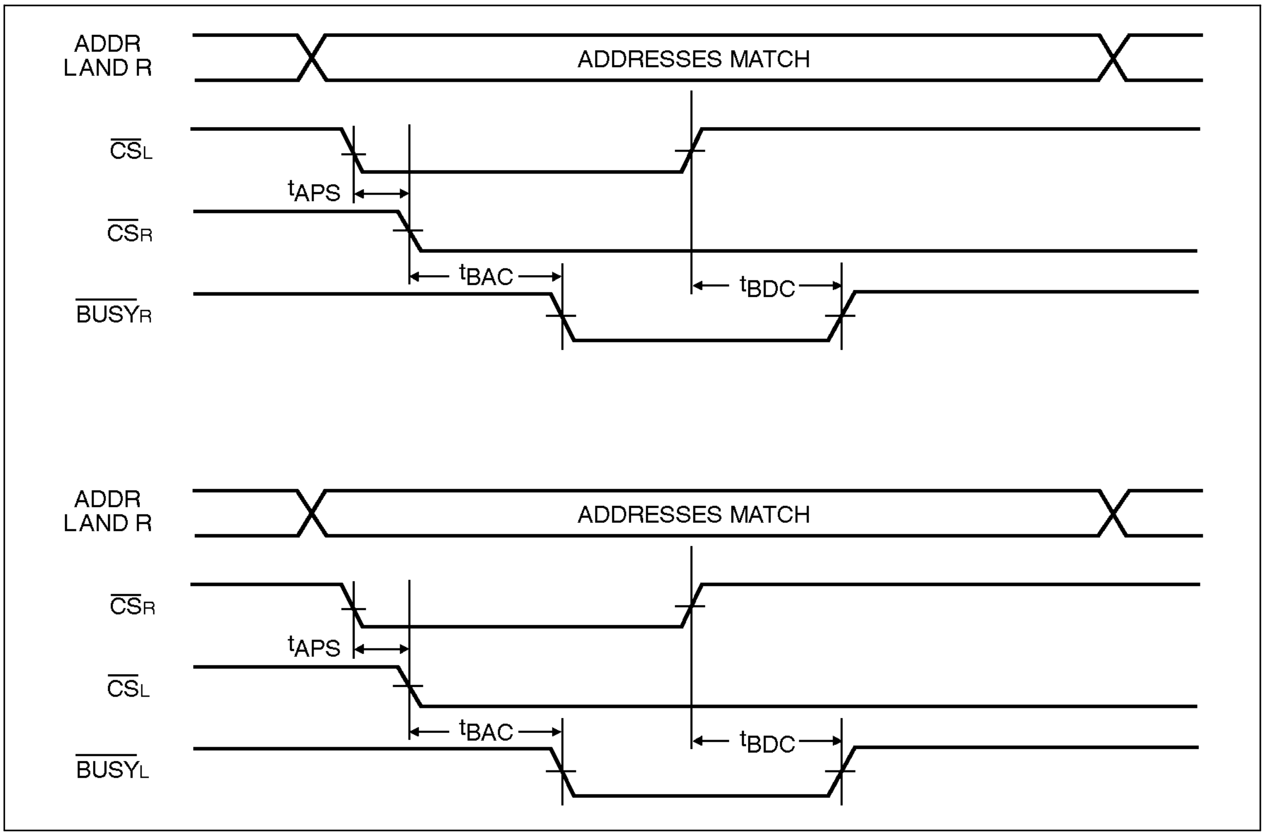


Notes :
1. Assume $\overline{\text{BUSY}}$ = H for the writing port, and $\overline{\text{OE}}$ = L for the reading port.
2. Write cycle parameters should be adhered to, to ensure proper writing.
3. Device is continuously enabled for both ports.

Timing Waveform of Write with $\overline{\text{BUSY}}$ (For Slave TSM67V05)

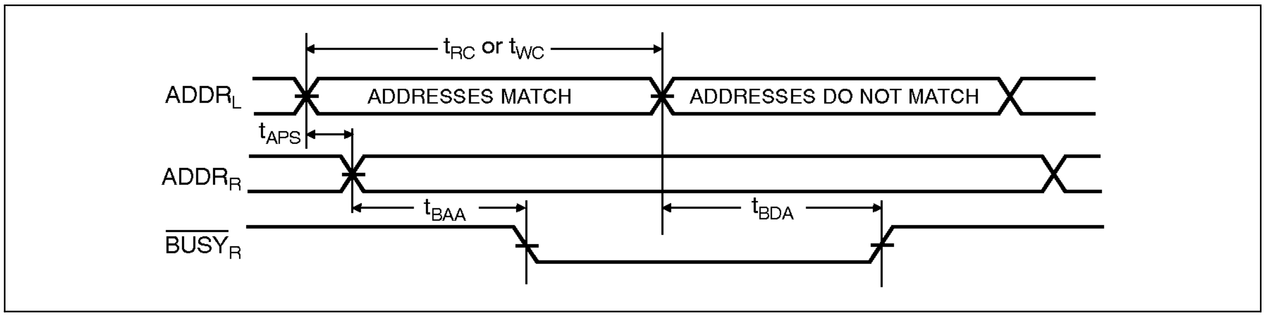


Timing Waveform of Contention Cycle n° 1, $\overline{\text{CS}}$ Arbitration (For Master TSM67V05 only)

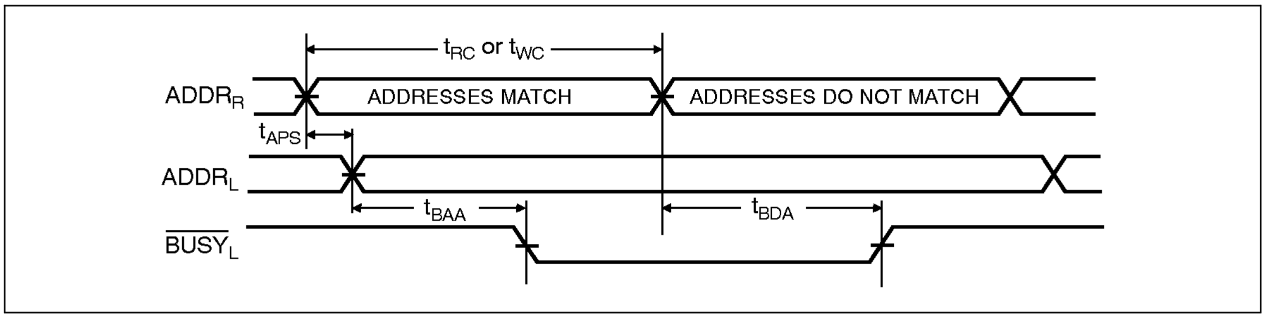


Timing Waveform of Contention Cycle n° 2, Address Valid Arbitration (For Master TSM67V05 only) (1)

Left Address Valid First :

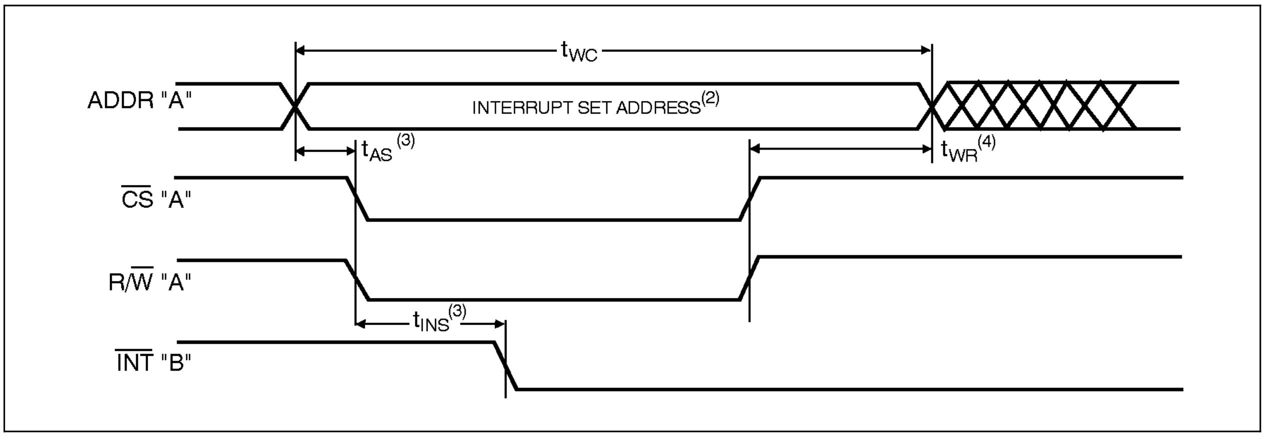


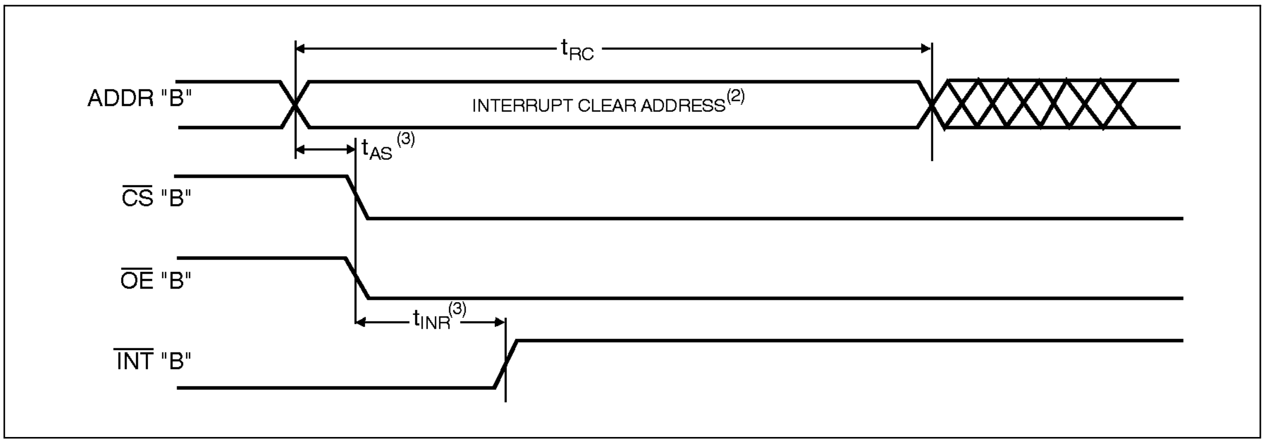
Right Address Valid First :



Note : 1. $\overline{CS}_L = \overline{CS}_R = V_{IL}$

Waveform of Interrupt Timing [1]





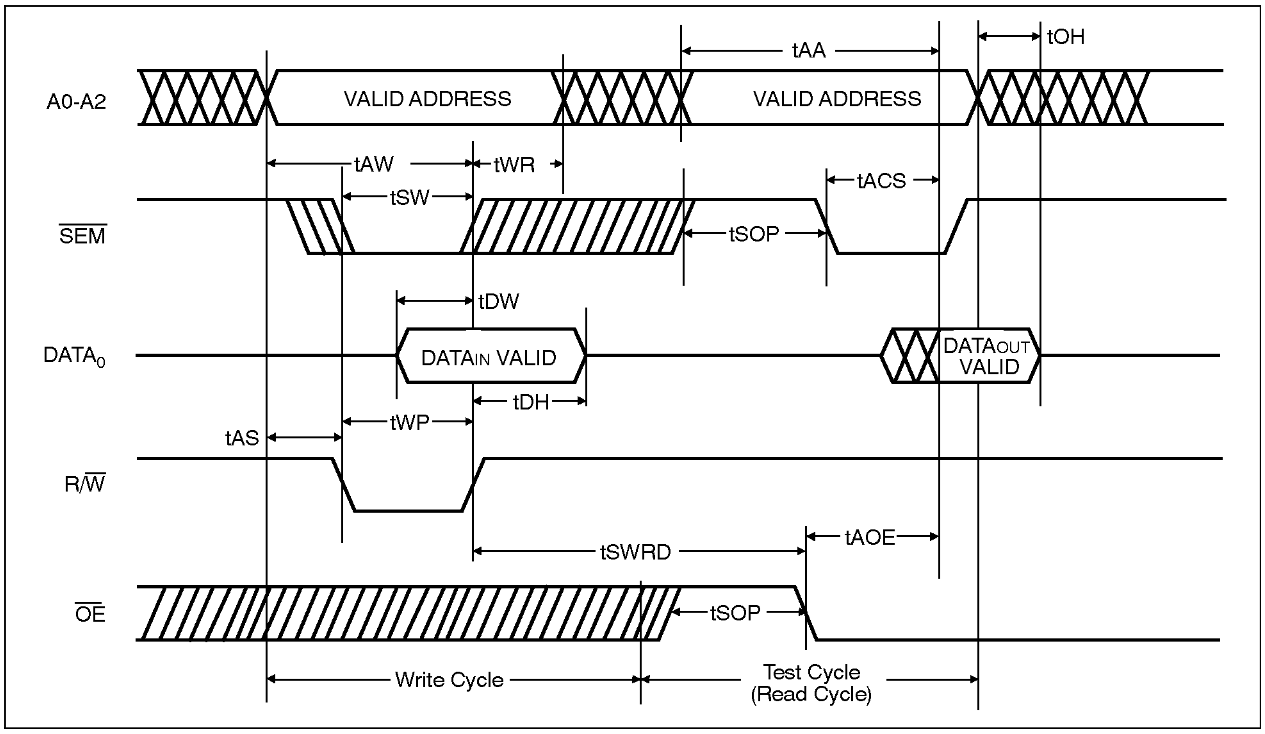
Notes :
1. All timing is the same for left and right ports. Port “A” may be either the left or right port. Port “B” is the port opposite from “A”.
2. See interrupt truth table.
3. Timing depends on which enable signal is asserted last.
4. Timing depends on which enable signal is de-asserted first.

AC Parameters

INTERRUPT TIMING	PARAMETER	TSM67V05–35(*)		TSM67V05–45		TSM67V05–55		UNIT
SYMBOL		MIN.	MAX.	MIN.	MAX.	MIN.	MAX.	
t_{AS}	Address set–up time	0	—	0	—	0	—	ns
t_{WR}	Write recovery time	0	—	0	—	0	—	ns
t_{INS}	Interrupt set time	—	30	—	35	—	40	ns
t_{INR}	Interrupt reset time	—	35	—	40	—	45	ns

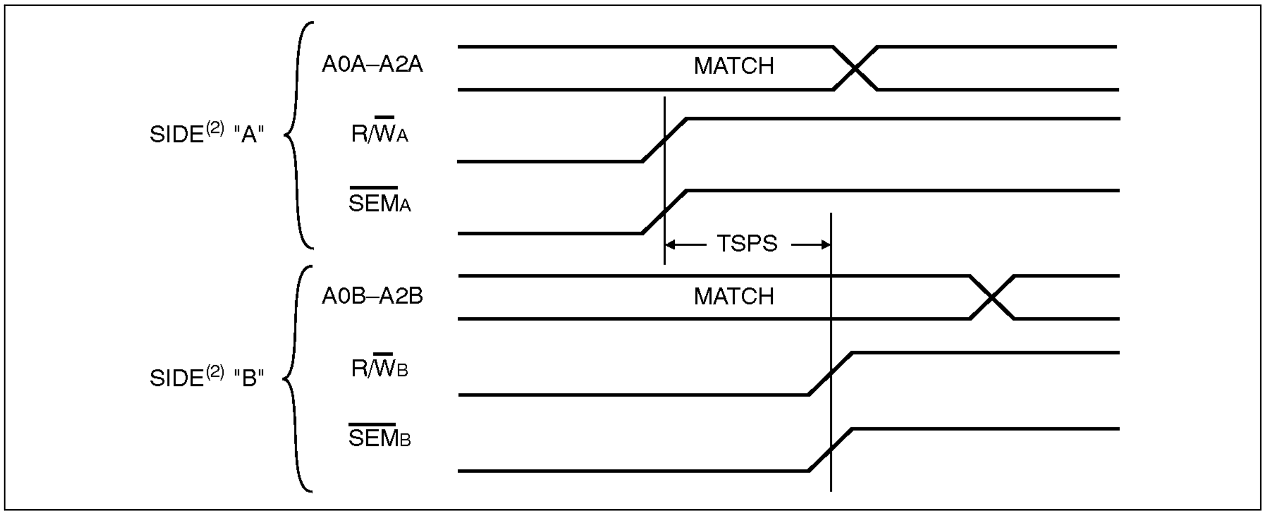
(*). Commercial only.

Timing Waveform of Semaphore Read after Write Timing, Either Side [1]



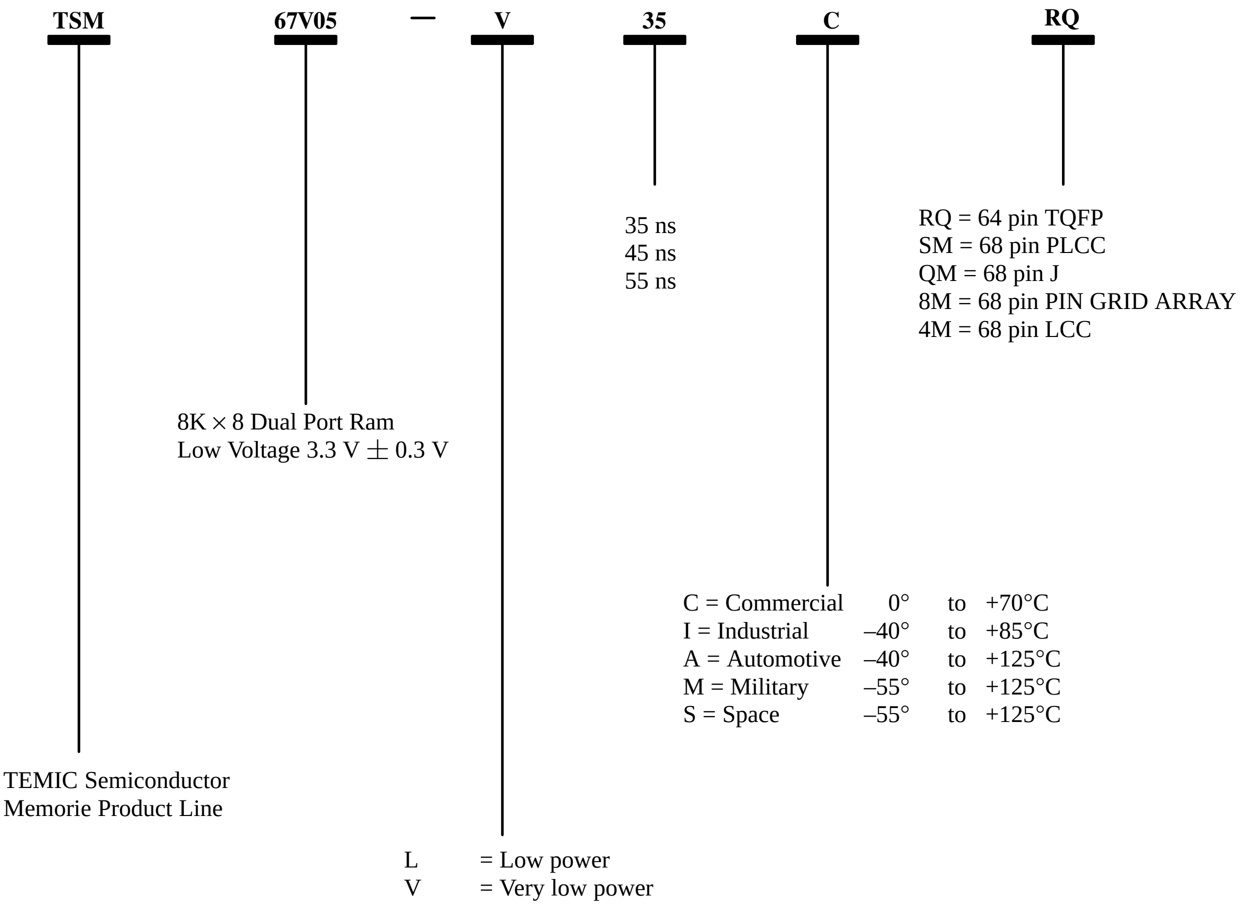
Note : 1. $\overline{CS}$ = V_{IH} for the duration of the above timing (both write and read cycle).

Timing Waveform of Semaphore Contention [1, 3, 4]



Notes :
1. $D_{OR} = D_{OL} = V_{IL}$, $\overline{C}_{SIR} = \overline{C}_{SL} = V_{IH}$, semaphore Flag is released from both sides (reads as ones from both sides) at cycle start.
2. Either side “A” = left and side “B” = right, or side “A” = right and side “B” = left.
3. This parameter is measured from the point where $R/\overline{W}_A$ or $\overline{SEM}_A$ goes high until $R/\overline{W}_B$ or $\overline{SEM}_B$ goes high.
4. If t_{SPS} is violated, the semaphore will fall positively to one side or the other, but there is no guaranted which side will obtain the flag.

Ordering Information



TSM 67V05 – V 35 C RQ

TEMIC Semiconductor
Memorie Product Line

8K × 8 Dual Port Ram
Low Voltage 3.3 V ± 0.3 V

L = Low power
V = Very low power

35 ns
45 ns
55 ns

C = Commercial	0°	to	+70°C
I = Industrial	–40°	to	+85°C
A = Automotive	–40°	to	+125°C
M = Military	–55°	to	+125°C
S = Space	–55°	to	+125°C

RQ = 64 pin TQFP
SM = 68 pin PLCC
QM = 68 pin J
8M = 68 pin PIN GRID ARRAY
4M = 68 pin LCC

The information contained herein is subject to change without notice. No responsibility is assumed by TEMIC SA for using this publication and/or circuits described herein : nor for any possible infringements of patents or other rights of third parties which may result from its use.