INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Projet Logique Programmable

Modélisation et synthèse d'un timer 8254 sur plateforme DE10-Lite

J.G. Cousin, O. Déforges,

Année Universitaire 2018/2019

But du projet

Il s’agit ici dans un premier temps de modéliser et de simuler le comportement d’un timer 8254, à partir de la solution fonctionnelle trouvée en TD de MCSE.

Dans une deuxième étape, le timer sera synthétisé sur FPGA (MAX 10) et intégré à la plateforme DE10-Lite.

L’étude fonctionnelle du timer est rappelée ci-dessous.

CONCEPTION D’UN TIMER

Un exemple de résultat

I. Spécification du problème

I.1 Analyse de l'environnement

L'environnement du timer est essentiellement constitué du CPU permettant de le programmer. Il faut en plus ajouter une entité positionnant l'entrée Gate (ce n'est pas forcément le CPU). Nous considérerons que cette entité de type périphérique reçoit également en entrée la sortie out du timer. Une autre entité fournit de manière indépendante le signal d'horloge.

CPU

De manière à pouvoir simplifier ultérieurement la réalisation, nous allons considérer ici que le signal CS est toujours actif après le positionnement des signaux WR et RD (ce qui sera possible en contrôlant le temps de décodage de l'adresse). Cela conduit à considérer que le signal CS va activer la partie dialogue CPU du timer (relation événementielle), alors que les signaux RD et WR sont eux évalués sur leur niveau.

Le modèle de description au niveau spécification définit trois types de relation : les évènements correspondant à des relations de type synchronisation, les données permanentes qui sont des relations « toujours présentes dans le temps », et qui contiennent une donnée, et les informations qui regroupent à la fois l’aspect évènementiel et le transport de données.

Le modèle à utiliser au niveau des spécifications ne permet de définir que des relations unidirectionnelles. On considère ainsi le bus D7-D0 bidirectionnel comme fonctionnellement constitué de deux relations: une CPU vers timer (DP), l'autre timer vers CPU (DT).

On définit les types suivants:

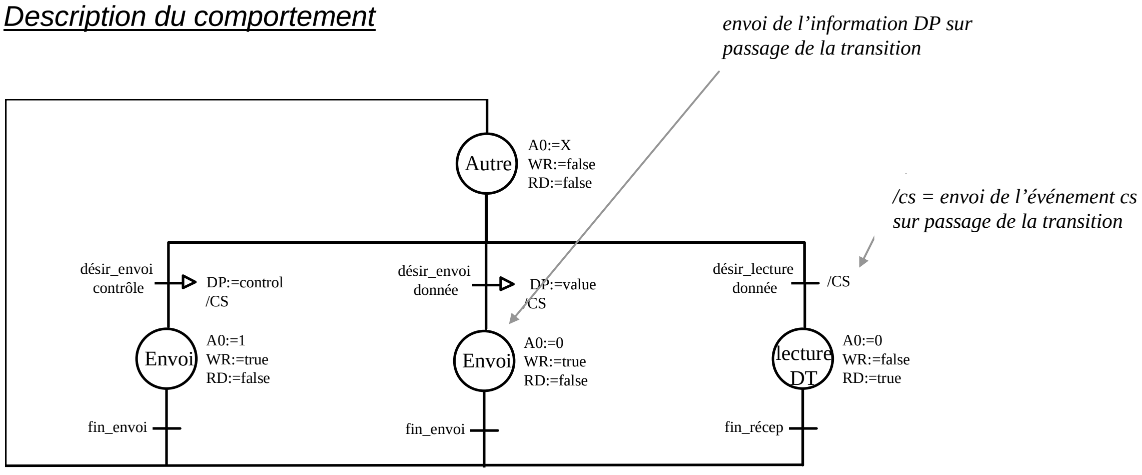
control = (SC+RW+M+BCD)
RW = (Latch | Poids) Poids = (Least , Most , LeastMost)

Définition des entrées/sorties

Entrée	DT : information, type octet.

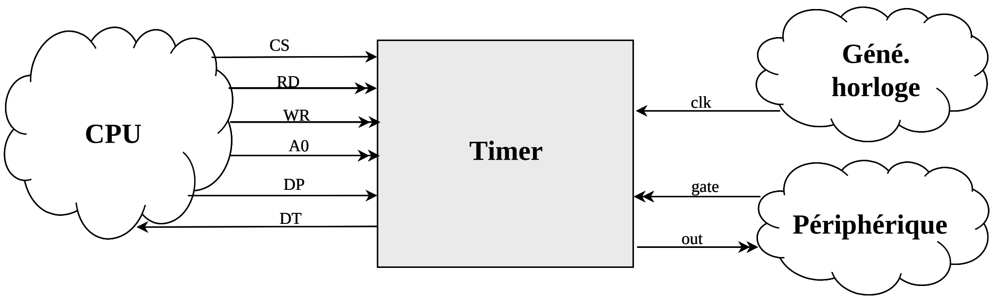
Sorties	WR, RD : données permanentes type booléen.
	A0 : donnée permanente, type bit.
	CS : événement.
	DP : information, type (control | value).

Description du comportement



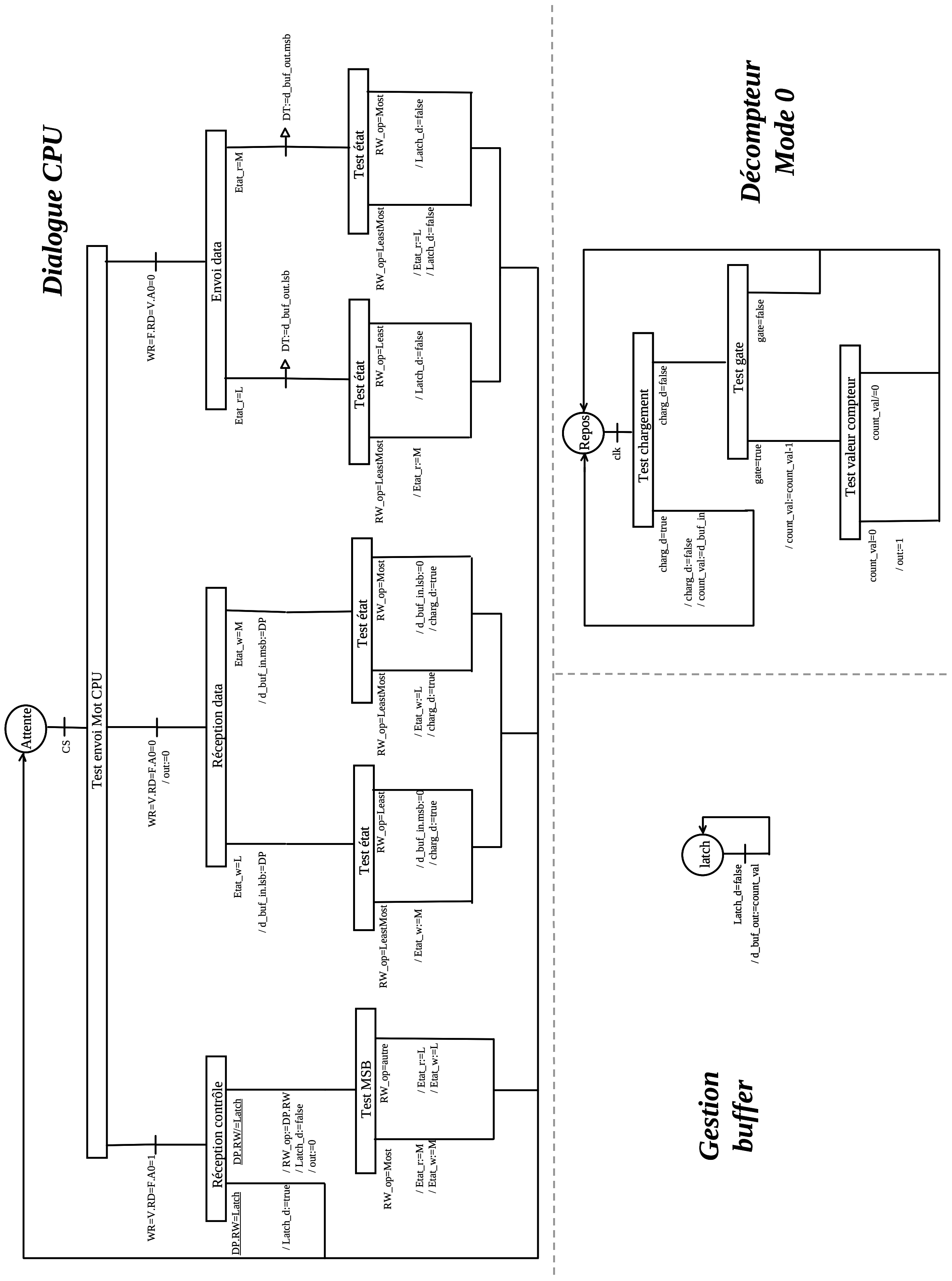
I.2 Délimitation des entrées/sorties du système

L'analyse suivante fournit la vue externe des relations du circuit et de son environnement.



I.3 Spécification fonctionnelle

L'analyse du fonctionnement demandé du timer fait apparaître de manière naturelle deux parties distinctes: l'une concerne le dialogue avec le CPU, l'autre le fonctionnement du timer dans le mode souhaité. Une autre fonction peut être formalisée, correspondant au verrouillage de la valeur du compteur pour la lecture. Le comportement souhaité du circuit est modélisé par les graphes de la page suivante.


Dialogue CPU
Attente
CS
Test envoi Mot CPU
WR=F.RD=V.A0=0
Envoi data
Etat_r=M
DT:=d_buf_out.msb
Test état
RW_op=Most
/ Latch_d:=false
RW_op=LeastMost
/ Etat_r:=L
/ Latch_d:=false
Etat_r=L
DT:=d_buf_out.lsb
Test état
RW_op=Least
/ Latch_d:=false
RW_op=LeastMost
/ Etat_r:=M
WR=V.RD=F.A0=0
/ out:=0
Réception data
Etat_w=M
/ d_buf_in.msb:=DP
Test état
RW_op=Most
/ d_buf_in.lsb:=0
/ charg_d:=true
RW_op=LeastMost
/ Etat_w:=L
/ charg_d:=true
Etat_w=L
/ d_buf_in.lsb:=DP
Test état
RW_op=Least
/ d_buf_in.msb:=0
/ charg_d:=true
RW_op=LeastMost
/ Etat_w:=M
WR=V.RD=F.A0=1
Réception contrôle
DP.RW/=Latch
/ RW_op:=DP.RW
/ Latch_d:=false
/ out:=0
DP.RW=Latch
/ Latch_d:=true
Test MSB
RW_op=autre
/ Etat_r:=L
/ Etat_w:=L
RW_op=Most
/ Etat_r:=M
/ Etat_w:=M
Décompteur
Mode 0
Repos
clk
Test chargement
charg_d=false
charg_d=true
/ charg_d:=false
/ count_val:=d_buf_in
Test gate
gate=false
gate=true
/ count_val:=count_val-1
Test valeur compteur
count_val/=0
count_val=0
/ out:=1
Gestion
buffer
latch
Latch_d=false
/ d_buf_out:=count_val

II. Conception fonctionnelle

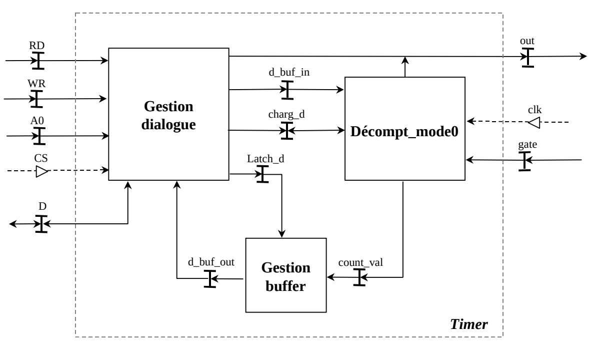
II.1 Délimitation des entrées/sorties du système

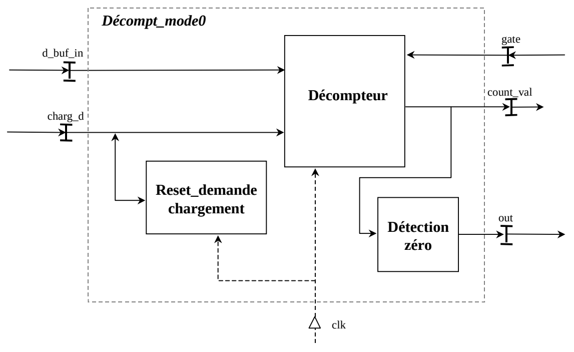
………

II.2 Structure fonctionnelle

Une première structure fonctionnelle peut être déduite directement de la spécification précédente. Elle fait apparaître trois fonctions de base.

La description comportementale de chacune des fonctions est ici conforme aux graphes de la spécification fonctionnelle. Si l'on désire mettre en évidence des fonctions plus élémentaires, notamment pour optimiser par la suite l'implantation, la fonction *décompt_mode0* peut être décomposée en trois fonctions de base.





Structure fonctionnelle Niveau 1 *Structure fonctionnelle Niveau 2*

Travail à réaliser

1 – Modélisation du timer dans l'environnement Modelsim

En partant de la structure fonctionnelle de niveau 1, réaliser la modélisation complète du timer en précisant éventuellement des modifications apportées à la description initiale. Donner des résultats de simulation précis en considérant les différents modes de fonctionnement du timer.

On notera les éléments suivants :

- *D* est un bus bidirectionnel trois états,
- *charg_d* et *out* possèdent deux sources d'écriture asynchrones,
- le bloc décompteur continue à décrémenter une fois la valeur 0 atteinte.

Les tests doivent au minimum concernées les fonctionnalités suivantes :

- Ecriture du CPU suivant les 3 modes possibles (LSB, MSB, LSB+MSB) et chargement du décompteur.
- Lecture du CPU suivant les 3 modes possibles (LSB, MSB, LSB+MSB) avec ou sans la commande *latch*.
- Mise à 1 et remise à 0 de *out*.

2 – Synthèse du timer dans l'environnement Quartus

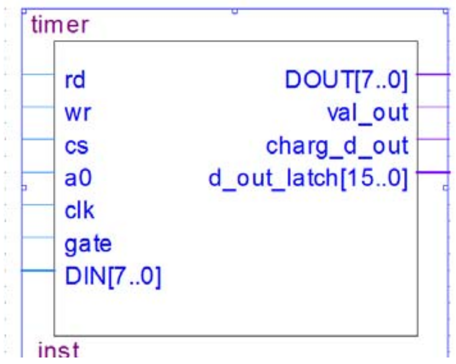
Sous l'outil Quartus, synthétiser le timer en y apportant éventuellement des modifications. Analyser les résultats de la synthèse au niveau RTL. Si nécessaire, apporter des modifications pour améliorer les résultats de la synthèse. On s'intéressera notamment à définit la nature (synchrone ou combinatoire) de la fonction *dialogue*.

3 – Intégration du timer sur la plateforme DE10-Lite

Le portage et le test du système sur la plateforme DE10-Lite nécessitent quelques modifications de l'entité initiale.

1. Le bus bidirectionnel D doit être transformé en un bus d'entrée DIN et un de sortie DOUT
2. Pour mieux analyser le comportent du système, on ajoute deux sorties supplémentaires : *charg_d* (1 bit) et *d_buf_out* (16 bits).

Ainsi, la nouvelle spécification des entrées/sorties est donnée dans la figure suivante.



Nouveau symbole associé au timer

Une fois les modifications du timer réalisées, il faut créer l'entité de niveau supérieur comme en TP. Vous devrez respecter les affectations suivantes :

```
-- Inputs
        rd <= SW(0);
        cs <= KEY(0);
        wr <= SW(1);
        a0 <= SW(2);
        gate <= SW(3);

        DIN(7 downto 2) <= SW(9 downto 4);
        DIN(1 downto 0) <= "00";

-- Display LEDs
        LED(0) <= rd;
        LED(1) <= wr;
        LED(2) <= a0;
        LED(3) <= gate;
        LED(4) <= cs;
        LED(8) <= charg_d_out;
        LED(7) <= val_out;
        LED(9) <= clk;
```

Pour les afficheurs 7 segments :

HEX1 + HEX0	*→ DOUT*
HEX5 + HEX4 + HEX3 +HEX2	*→ d_buf_out*

Tester le système complet suivant les différentes configurations. Si besoin, vous pouvez ajouter des sorties au timer pour visualiser des variables internes.

ANNEXE

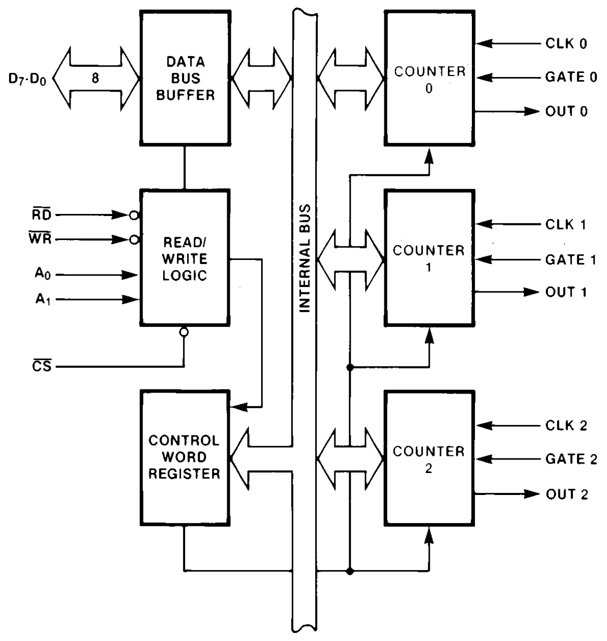
DOCUMENTATION 8085

8254
PROGRAMMABLE INTERVAL TIMER

- **Compatible with All Intel and Most Other Microprocessors**
- **Handles Inputs from DC to 10 MHz**
  — 8 MHz 8254
  — 10 MHz 8254-2
- **Status Read-Back Command**
- **Six Programmable Counter Modes**
- **Three Independent 16-Bit Counters**
- **Binary or BCD Counting**
- **Single +5V Supply**
- **Available in EXPRESS**
  — Standard Temperature Range

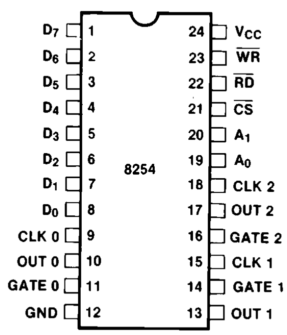
The Intel 8254 is a counter/timer device designed to solve the common timing control problems in microcomputer system design. It provides three independent 16-bit counters, each capable of handling clock inputs up to 10 MHz. All modes are software programmable. The 8254 is a superset of the 8253.

The 8254 uses HMOS technology and comes in a 24-pin plastic or CERDIP package.



231164–1

Figure 1. 8254 Block Diagram



231164–2

Figure 2. Pin Configuration

Table 1. Pin Description

Symbol	Pin No.	Type	Name and Function
D_7–D_0	1–8	I/O	**DATA:** Bi-directional three state data bus lines, connected to system data bus.
CLK 0	9	I	**CLOCK 0:** Clock input of Counter 0.
OUT 0	10	O	**OUTPUT 0:** Output of Counter 0.
GATE 0	11	I	**GATE 0:** Gate input of Counter 0.
GND	12		**GROUND:** Power supply connection.
V_{CC}	24		**POWER:** +5V power supply connection.
$\overline{WR}$	23	I	**WRITE CONTROL:** This input is low during CPU write operations.
$\overline{RD}$	22	I	**READ CONTROL:** This input is low during CPU read operations.
$\overline{CS}$	21	I	**CHIP SELECT:** A low on this input enables the 8254 to respond to $\overline{RD}$ and $\overline{WR}$ signals. $\overline{RD}$ and $\overline{WR}$ are ignored otherwise.
A_1, A_0	20–19	I	**ADDRESS:** Used to select one of the three Counters or the Control Word Register for read or write operations. Normally connected to the system address bus.
CLK 2	18	I	**CLOCK 2:** Clock input of Counter 2.
OUT 2	17	O	**OUT 2:** Output of Counter 2.
GATE 2	16	I	**GATE 2:** Gate input of Counter 2.
CLK 1	15	I	**CLOCK 1:** Clock input of Counter 1.
GATE 1	14	I	**GATE 1:** Gate input of Counter 1.
OUT 1	13	O	**OUT 1:** Output of Counter 1.

A_1	A_0	Selects
0	0	Counter 0
0	1	Counter 1
1	0	Counter 2
1	1	Control Word Register

FUNCTIONAL DESCRIPTION

General

The 8254 is a programmable interval timer/counter designed for use with Intel microcomputer systems. It is a general purpose, multi-timing element that can be treated as an array of I/O ports in the system software.

The 8254 solves one of the most common problems in any microcomputer system, the generation of accurate time delays under software control. Instead of setting up timing loops in software, the programmer configures the 8254 to match his requirements and programs one of the counters for the desired delay. After the desired delay, the 8254 will interrupt the CPU. Software overhead is minimal and variable length delays can easily be accommodated.

Some of the other counter/timer functions common to microcomputers which can be implemented with the 8254 are:

- Real time clock
- Event-counter
- Digital one-shot
- Programmable rate generator
- Square wave generator
- Binary rate multiplier
- Complex waveform generator
- Complex motor controller

Block Diagram

DATA BUS BUFFER

This 3-state, bi-directional, 8-bit buffer is used to interface the 8254 to the system bus (see Figure 3).

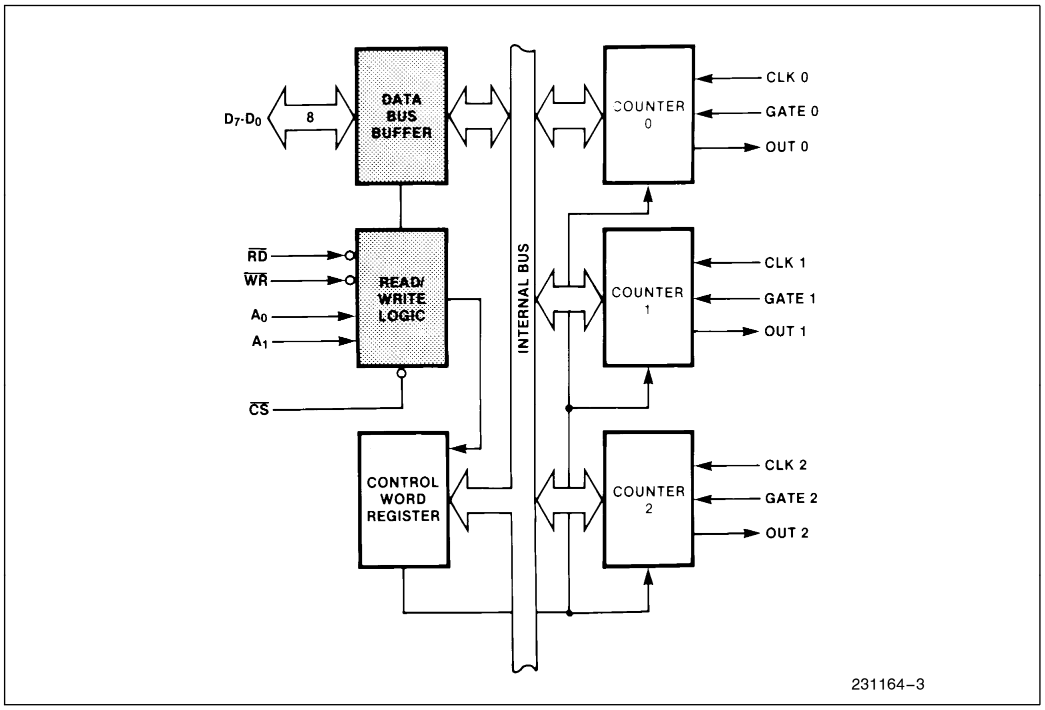


Figure 3. Block Diagram Showing Data Bus Buffer and Read/Write Logic Functions

READ/WRITE LOGIC

The Read/Write Logic accepts inputs from the system bus and generates control signals for the other functional blocks of the 8254. A_1 and A_0 select one of the three counters or the Control Word Register to be read from/written into. A "low" on the $\overline{RD}$ input tells the 8254 that the CPU is reading one of the counters. A "low" on the $\overline{WR}$ input tells the 8254 that the CPU is writing either a Control Word or an initial count. Both $\overline{RD}$ and $\overline{WR}$ are qualified by $\overline{CS}$; $\overline{RD}$ and $\overline{WR}$ are ignored unless the 8254 has been selected by holding $\overline{CS}$ low.

CONTROL WORD REGISTER

The Control Word Register (see Figure 4) is selected by the Read/Write Logic when $A_1,A_0 = 11$. If the CPU then does a write operation to the 8254, the data is stored in the Control Word Register and is interpreted as a Control Word used to define the operation of the Counters.

The Control Word Register can only be written to; status information is available with the Read-Back Command.

COUNTER 0, COUNTER 1, COUNTER 2

These three functional blocks are identical in operation, so only a single Counter will be described. The internal block diagram of a single counter is shown in Figure 5.

The Counters are fully independent. Each Counter may operate in a different Mode.

The Control Word Register is shown in the figure; it is not part of the Counter itself, but its contents determine how the Counter operates.

The status register, shown in Figure 5, when latched, contains the current contents of the Control Word Register and status of the output and null count flag. (See detailed explanation of the Read-Back command.)

The actual counter is labelled CE (for "Counting Element"). It is a 16-bit presettable synchronous down counter.

OL_M and OL_L are two 8-bit latches. OL stands for "Output Latch"; the subscripts M and L stand for "Most significant byte" and "Least significant byte"

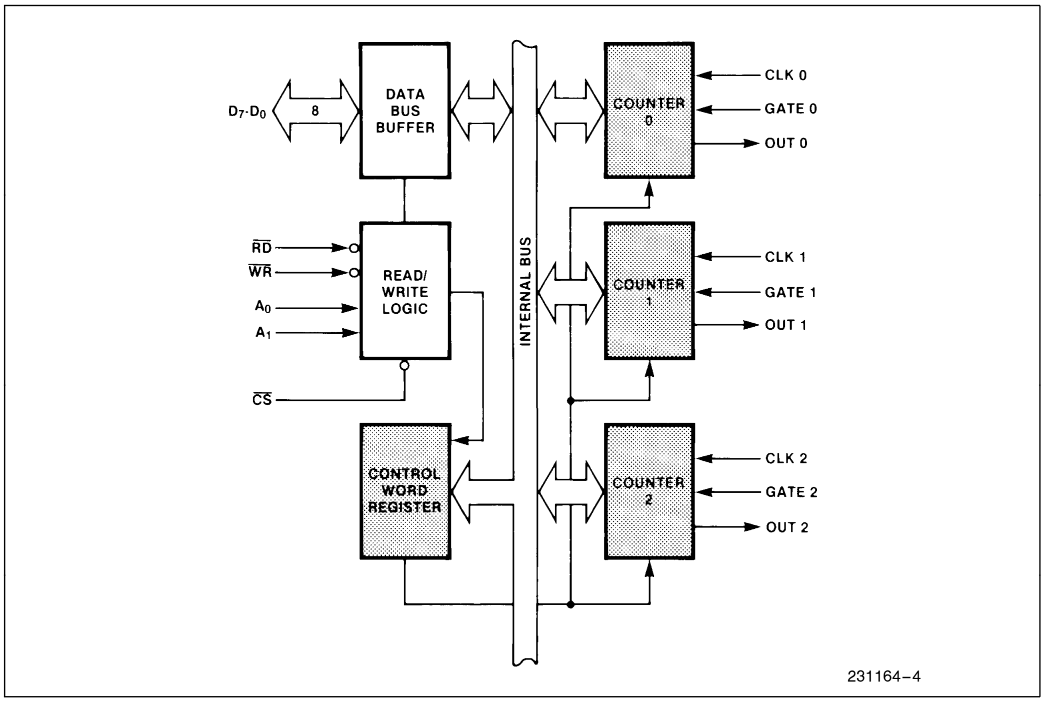


Figure 4. Block Diagram Showing Control Word Register and Counter Functions

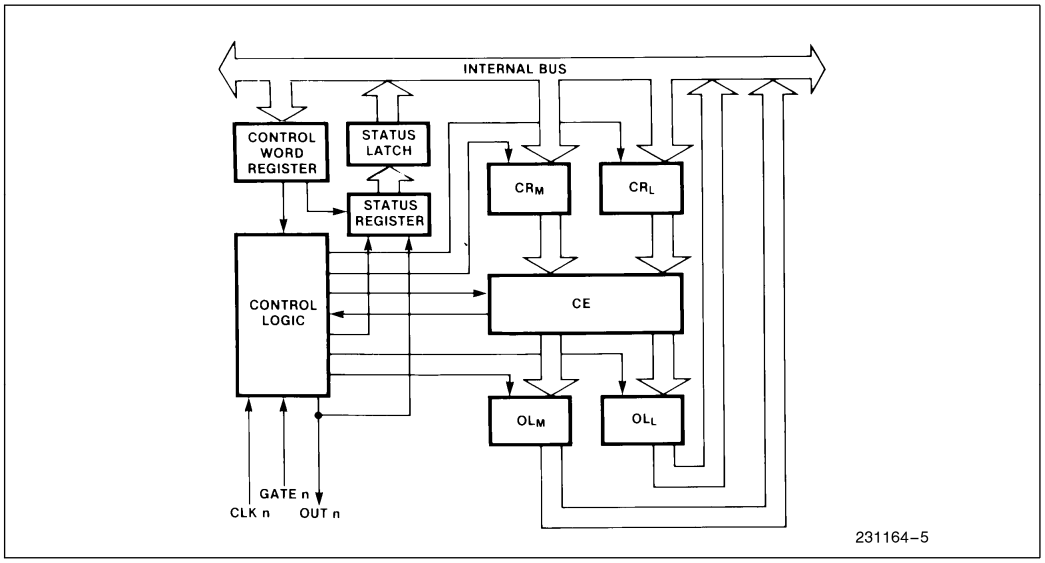


Figure 5. Internal Block Diagram of a Counter

respectively. Both are normally referred to as one unit and called just OL. These latches normally "follow" the CE, but if a suitable Counter Latch Command is sent to the 8254, the latches "latch" the present count until read by the CPU and then return to "following" the CE. One latch at a time is enabled by the counter's Control Logic to drive the internal bus. This is how the 16-bit Counter communicates over the 8-bit internal bus. Note that the CE itself cannot be read; whenever you read the count, it is the OL that is being read.

Similarly, there are two 8-bit registers called CR_M and CR_L (for "Count Register"). Both are normally referred to as one unit and called just CR. When a new count is written to the Counter, the count is stored in the CR and later transferred to the CE. The Control Logic allows one register at a time to be loaded from the internal bus. Both bytes are transferred to the CE simultaneously. CR_M and CR_L are cleared when the Counter is programmed. In this way, if the Counter has been programmed for one byte counts (either most significant byte only or least significant byte only) the other byte will be zero. Note that the CE cannot be written into; whenever a count is written, it is written into the CR.

The Control Logic is also shown in the diagram. CLK n, GATE n, and OUT n are all connected to the outside world through the Control Logic.

8254 SYSTEM INTERFACE

The 8254 is a component of the Intel Microcomputer Systems and interfaces in the same manner as all other peripherals of the family. It is treated by the system's software as an array of peripheral I/O ports; three are counters and the fourth is a control register for MODE programming.

Basically, the select inputs A_0,A_1 connect to the A_0, A_1 address bus signals of the CPU. The CS can be derived directly from the address bus using a linear select method. Or it can be connected to the output of a decoder, such as an Intel 8205 for larger systems.

OPERATIONAL DESCRIPTION

General

After power-up, the state of the 8254 is undefined. The Mode, count value, and output of all Counters are undefined.

How each Counter operates is determined when it is programmed. Each Counter must be programmed before it can be used. Unused counters need not be programmed.

Programming the 8254

Counters are programmed by writing a Control Word and then an initial count.

The Control Words are written into the Control Word Register, which is selected when A_1,A_0 = 11. The Control Word itself specifies which Counter is being programmed.

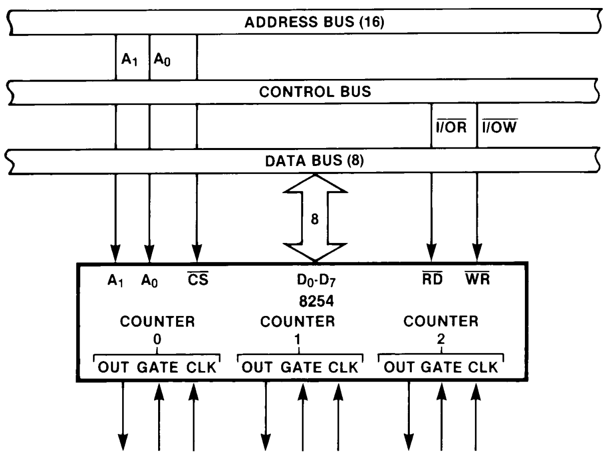


Figure 6. 8254 System Interface

Control Word Format

$A_1, A_0 = 11$ $\overline{CS} = 0$ $\overline{RD} = 1$ $\overline{WR} = 0$

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
SC1	SC0	RW1	RW0	M2	M1	M0	BCD

SC—Select Counter

SC1	SC0	
0	0	Select Counter 0
0	1	Select Counter 1
1	0	Select Counter 2
1	1	Read-Back Command (see Read Operations)

M—Mode

M2	M1	M0	
0	0	0	Mode 0
0	0	1	Mode 1
X	1	0	Mode 2
X	1	1	Mode 3
1	0	0	Mode 4
1	0	1	Mode 5

RW—Read/Write

RW1	RW0	
0	0	Counter Latch Command (see Read Operations)
0	1	Read/Write least significant byte only
1	0	Read/Write most significant byte only
1	1	Read/Write least significant byte first, then most significant byte

BCD

0	Binary Counter 16-bits
1	Binary Coded Decimal (BCD) Counter (4 Decades)

NOTE:
Don't care bits (X) should be 0 to insure compatibility with future Intel products.

Figure 7. Control Word Format

By contrast, initial counts are written into the Counters, not the Control Word Register. The A_1, A_0 inputs are used to select the Counter to be written into. The format of the initial count is determined by the Control Word used.

Write Operations

The programming procedure for the 8254 is very flexible. Only two conventions need to be remembered:

1) For each Counter, the Control Word must be written before the initial count is written.
2) The initial count must follow the count format specified in the Control Word (least significant byte only, most significant byte only, or least significant byte and then most significant byte).

Since the Control Word Register and the three Counters have separate addresses (selected by the A_1, A_0 inputs), and each Control Word specifies the Counter it applies to (SC0,SC1 bits), no special instruction sequence is required. Any programming sequence that follows the conventions in Figure 7 is acceptable.

A new initial count may be written to a Counter at any time without affecting the Counter's programmed Mode in any way. Counting will be affected as described in the Mode definitions. The new count must follow the programmed count format.

If a Counter is programmed to read/write two-byte counts, the following precaution applies: A program must not transfer control between writing the first and second byte to another routine which also writes into that same Counter. Otherwise, the Counter will be loaded with an incorrect count.

	A_1	A_0		A_1	A_0
Control Word—Counter 0	1	1	Control Word—Counter 2	1	1
LSB of count—Counter 0	0	0	Control Word—Counter 1	1	1
MSB of count—Counter 0	0	0	Control Word—Counter 0	1	1
Control Word—Counter 1	1	1	LSB of count—Counter 2	1	0
LSB of count—Counter 1	0	1	MSB of count—Counter 2	1	0
MSB of count—Counter 1	0	1	LSB of count—Counter 1	0	1
Control Word—Counter 2	1	1	MSB of count—Counter 1	0	1
LSB of count—Counter 2	1	0	LSB of count—Counter 0	0	0
MSB of count—Counter 2	1	0	MSB of count—Counter 0	0	0

	A_1	A_0		A_1	A_0
Control Word—Counter 0	1	1	Control Word—Counter 1	1	1
Control Word—Counter 1	1	1	Control Word—Counter 0	1	1
Control Word—Counter 2	1	1	LSB of count—Counter 1	0	1
LSB of count—Counter 2	1	0	Control Word—Counter 2	1	1
LSB of count—Counter 1	0	1	LSB of count—Counter 0	0	0
LSB of count—Counter 0	0	0	MSB of count—Counter 1	0	1
MSB of count—Counter 0	0	0	LSB of count—Counter 2	1	0
MSB of count—Counter 1	0	1	MSB of count—Counter 0	0	0
MSB of count—Counter 2	1	0	MSB of count—Counter 2	1	0

NOTE:
In all four examples, all Counters are programmed to read/write two-byte counts. These are only four of many possible programming sequences.

Figure 8. A Few Possible Programming Sequences

Read Operations

It is often desirable to read the value of a Counter without disturbing the count in progress. This is easily done in the 8254.

There are three possible methods for reading the counters: a simple read operation, the Counter Latch Command, and the Read-Back Command. Each is explained below. The first method is to perform a simple read operation. To read the Counter, which is selected with the A1, A0 inputs, the CLK input of the selected Counter must be inhibited by using either the GATE input or external logic. Otherwise, the count may be in the process of changing when it is read, giving an undefined result.

COUNTER LATCH COMMAND

The second method uses the "Counter Latch Command". Like a Control Word, this command is written to the Control Word Register, which is selected when A_1,A_0 = 11. Also like a Control Word, the SC0, SC1 bits select one of the three Counters, but two other bits, D5 and D4, distinguish this command from a Control Word.

A_1,A_0 = 11; CS = 0; RD = 1; WR = 0

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
SC1	SC0	0	0	X	X	X	X

SC1,SC0—specify counter to be latched

SC1	SC0	Counter
0	0	0
0	1	1
1	0	2
1	1	Read-Back Command

D5,D4—00 designates Counter Latch Command

X—don't care

NOTE:
Don't care bits (X) should be 0 to insure compatibility with future Intel products.

Figure 9. Counter Latching Command Format

The selected Counter's output latch (OL) latches the count at the time the Counter Latch Command is received. This count is held in the latch until it is read by the CPU (or until the Counter is reprogrammed). The count is then unlatched automatically and the OL returns to "following" the counting element (CE). This allows reading the contents of the Counters "on the fly" without affecting counting in progress. Multiple Counter Latch Commands may be used to latch more than one Counter. Each latched Counter's OL holds its count until it is read. Counter Latch Commands do not affect the programmed Mode of the Counter in any way.

If a Counter is latched and then, some time later, latched again before the count is read, the second Counter Latch Command is ignored. The count read will be the count at the time the first Counter Latch Command was issued.

With either method, the count must be read according to the programmed format; specifically, if the Counter is programmed for two byte counts, two bytes must be read. The two bytes do not have to be read one right after the other; read or write or programming operations of other Counters may be inserted between them.

Another feature of the 8254 is that reads and writes of the same Counter may be interleaved; for example, if the Counter is programmed for two byte counts, the following sequence is valid.

1) Read least significant byte.
2) Write new least significant byte.
3) Read most significant byte.
4) Write new most significant byte.

If a Counter is programmed to read/write two-byte counts, the following precaution applies: A program must not transfer control between reading the first and second byte to another routine which also reads from that same Counter. Otherwise, an incorrect count will be read.

READ-BACK COMMAND

The third method uses the Read-Back Command. This command allows the user to check the count value, programmed Mode, and current states of the OUT pin and Null Count flag of the selected counter(s).

The command is written into the Control Word Register and has the format shown in Figure 10. The command applies to the counters selected by setting their corresponding bits D3, D2, D1 = 1.

A0, A1 = 11 $\overline{\text{CS}}$ = 0 $\overline{\text{RD}}$ = 1 $\overline{\text{WR}}$ = 0

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	1	$\overline{\text{COUNT}}$	$\overline{\text{STATUS}}$	CNT 2	CNT 1	CNT 0	0

D_5: 0 = Latch count of selected counter(s)
D_4: 0 = Latch status of selected counters(s)
D_3: 1 = Select Counter 2
D_2: 1 = Select Counter 1
D_1: 1 = Select Counter 0
D_0: Reserved for future expansion; Must be 0

Figure 10. Read-Back Command Format

The read-back command may be used to latch multiple counter output latches (OL) by setting the $\overline{\text{COUNT}}$ bit D5 = 0 and selecting the desired counter(s). This single command is functionally equivalent to several counter latch commands, one for each counter latched. Each counter's latched count is held until it is read (or the counter is reprogrammed). The counter is automatically unlatched when read, but other counters remain latched until they are read. If multiple count read-back commands are issued to the same counter without reading the count, all but the first are ignored; i.e., the count which will be read is the count at the time the first read-back command was issued.

The read-back command may also be used to latch status information of selected counter(s) by setting $\overline{\text{STATUS}}$ bit D4 = 0. Status must be latched to be read; status of a counter is accessed by a read from that counter.

The counter status format is shown in Figure 11. Bits D5 through D0 contain the counter's programmed Mode exactly as written in the last Mode Control Word. OUTPUT bit D7 contains the current state of the OUT pin. This allows the user to monitor the counter's output via software, possibly eliminating some hardware from a system.

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
Output	Null Count	RW1	RW0	M2	M1	M0	BCD

D_7	1 = OUT Pin is 1 0 = OUT Pin is 0
D_6	1 = Null Count 0 = Count available for reading
D_5–D_0	Counter programmed mode (see Figure 7)

Figure 11. Status Byte

NULL COUNT bit D6 indicates when the last count written to the counter register (CR) has been loaded into the counting element (CE). The exact time this happens depends on the Mode of the counter and is described in the Mode Definitions, but until the count is loaded into the counting element (CE), it can't be read from the counter. If the count is latched or read before this time, the count value will not reflect the new count just written. The operation of Null Count is shown in Figure 12.

This Action	Causes
A. Write to the control word register;(1)	Null Count = 1
B. Write to the count register (CR);(2)	Null Count = 1
C. New Count is loaded into CE (CR $\longrightarrow$ CE);	Null Count = 0

NOTE:
1. Only the counter specified by the control word will have its Null Count set to 1. Null count bits of other counters are unaffected.
2. If the counter is programmed for two-byte counts (least significant byte then most significant byte) Null Count goes to 1 when the second byte is written.

Figure 12. Null Count Operation

If multiple status latch operations of the counter(s) are performed without reading the status, all but the first are ignored; i.e., the status that will be read is the status of the counter at the time the first status read-back command was issued.

Both count and status of the selected counter(s) may be latched simultaneously by setting both $\overline{\text{COUNT}}$ and $\overline{\text{STATUS}}$ bits D5,D4 = 0. This is functionally the same as issuing two separate read-back commands at once, and the above discussions apply here also. Specifically, if multiple count and/or status read-back commands are issued to the same counter(s) without any intervening reads, all but the first are ignored. This is illustrated in Figure 13.

If both count and status of a counter are latched, the first read operation of that counter will return latched status, regardless of which was latched first. The next one or two reads (depending on whether the counter is programmed for one or two type counts) return latched count. Subsequent reads return unlatched count.

$\overline{CS}$	$\overline{RD}$	$\overline{WR}$	A_1	A_0	
0	1	0	0	0	Write into Counter 0
0	1	0	0	1	Write into Counter 1
0	1	0	1	0	Write into Counter 2
0	1	0	1	1	Write Control Word
0	0	1	0	0	Read from Counter 0
0	0	1	0	1	Read from Counter 1
0	0	1	1	0	Read from Counter 2
0	0	1	1	1	No-Operation (3-State)
1	X	X	X	X	No-Operation (3-State)
0	1	1	X	X	No-Operation (3-State)

Figure 14. Read/Write Operations Summary

Command								Description	Result
D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0		
1	1	0	0	0	0	1	0	Read back count and status of Counter 0	Count and status latched for Counter 0
1	1	1	0	0	1	0	0	Read back status of Counter 1	Status latched for Counter 1
1	1	1	0	1	1	0	0	Read back status of Counters 2, 1	Status latched for Counter 2, but not Counter 1
1	1	0	1	1	0	0	0	Read back count of Counter 2	Count latched for Counter 2
1	1	0	0	0	1	0	0	Read back count and status of Counter 1	Count latched for Counter 1, but not status
1	1	1	0	0	0	1	0	Read back status of Counter 1	Command ignored, status already latched for Counter 1

Figure 13. Read-Back Command Example

Mode Definitions

The following are defined for use in describing the operation of the 8254.

CLK Pulse:	a rising edge, then a falling edge, in that order, of a Counter's CLK input.
Trigger:	a rising edge of a Counter's GATE input.
Counter loading:	the transfer of a count from the CR to the CE (refer to the "Functional Description")

MODE 0: INTERRUPT ON TERMINAL COUNT

Mode 0 is typically used for event counting. After the Control Word is written, OUT is initially low, and will remain low until the Counter reaches zero. OUT then goes high and remains high until a new count or a new Mode 0 Control Word is written into the Counter.

GATE = 1 enables counting; GATE = 0 disables counting. GATE has no effect on OUT.

After the Control Word and initial count are written to a Counter, the initial count will be loaded on the next CLK pulse. This CLK pulse does not decrement the count, so for an initial count of N, OUT does not go high until N + 1 CLK pulses after the initial count is written.

If a new count is written to the Counter, it will be loaded on the next CLK pulse and counting will continue from the new count. If a two-byte count is written, the following happens:

1) Writing the first byte disables counting. OUT is set low immediately (no clock pulse required)
2) Writing the second byte allows the new count to be loaded on the next CLK pulse.

This allows the counting sequence to be synchronized by software. Again, OUT does not go high until N+1 CLK pulses after the new count of N is written.

If an initial count is written while GATE = 0, it will still be loaded on the next CLK pulse. When GATE goes high, OUT will go high N CLK pulses later; no CLK pulse is needed to load the Counter as this has already been done.

MODE 1: HARDWARE RETRIGGERABLE ONE-SHOT

OUT will be initially high. OUT will go low on the CLK pulse following a trigger to begin the one-shot pulse, and will remain low until the Counter reaches zero. OUT will then go high and remain high until the CLK pulse after the next trigger.

After writing the Control Word and initial count, the Counter is armed. A trigger results in loading the Counter and setting OUT low on the next CLK pulse, thus starting the one-shot pulse. An initial count of N will result in a one-shot pulse N CLK cycles in duration. The one-shot is retriggerable, hence OUT will remain low for N CLK pulses after any trigger. The one-shot pulse can be repeated without rewriting the same count into the counter. GATE has no effect on OUT.

If a new count is written to the Counter during a one-shot pulse, the current one-shot is not affected unless the counter is retriggered. In that case, the Counter is loaded with the new count and the one-shot pulse continues until the new count expires.

MODE 2: RATE GENERATOR

This Mode functions like a divide-by-N counter. It is typically used to generate a Real Time Clock interrupt. OUT will initially be high. When the initial count has decremented to 1, OUT goes low for one CLK pulse. OUT then goes high again, the Counter reloads the initial count and the process is repeated. Mode 2 is periodic; the same sequence is repeated indefinitely. For an initial count of N, the sequence repeats every N CLK cycles.

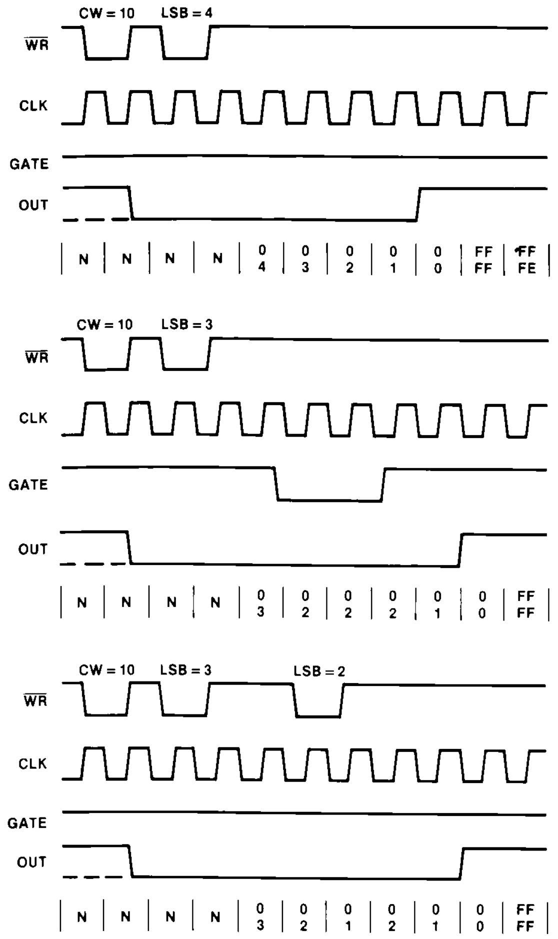
GATE = 1 enables counting; GATE = 0 disables counting. If GATE goes low during an output pulse, OUT is set high immediately. A trigger reloads the Counter with the initial count on the next CLK pulse; OUT goes low N CLK pulses after the trigger. Thus the GATE input can be used to synchronize the Counter.

After writing a Control Word and initial count, the Counter will be loaded on the next CLK pulse. OUT goes low N CLK Pulses after the initial count is written. This allows the Counter to be synchronized by software also.

Writing a new count while counting does not affect the current counting sequence. If a trigger is received after writing a new count but before the end of the current period, the Counter will be loaded with the new count on the next CLK pulse and counting will continue from the new count. Otherwise, the new count will be loaded at the end of the current counting cycle. In mode 2, a COUNT of 1 is illegal.

MODE 3: SQUARE WAVE MODE

Mode 3 is typically used for Baud rate generation. Mode 3 is similar to Mode 2 except for the duty cycle of OUT. OUT will initially be high. When half the


CW = 10
LSB = 4
WR
CLK
GATE
OUT
N N N N 0/4 0/3 0/2 0/1 0/0 FF/FF FF/FE
CW = 10
LSB = 3
WR
CLK
GATE
OUT
N N N N 0/3 0/2 0/2 0/2 0/1 0/0 FF/FF
CW = 10
LSB = 3
LSB = 2
WR
CLK
GATE
OUT
N N N N 0/3 0/2 0/1 0/2 0/1 0/0 FF/FF

231164–7

NOTE:
The following conventions apply to all mode timing diagrams:
1. Counters are programmed for binary (not BCD) counting and for reading/writing least significant byte (LSB) only.
2. The counter is always selected ($\overline{CS}$ always low).
3. CW stands for "Control Word"; CW = 10 means a control word of 10 HEX is written to the counter.
4. LSB stands for "Least Significant Byte" of count.
5. Numbers below diagrams are count values. The lower number is the least significant byte. The upper number is the most significant byte. Since the counter is programmed to read/write LSB only, the most significant byte cannot be read.
N stands for an undefined count.
Vertical lines show transitions between count values.

Figure 15. Mode 0