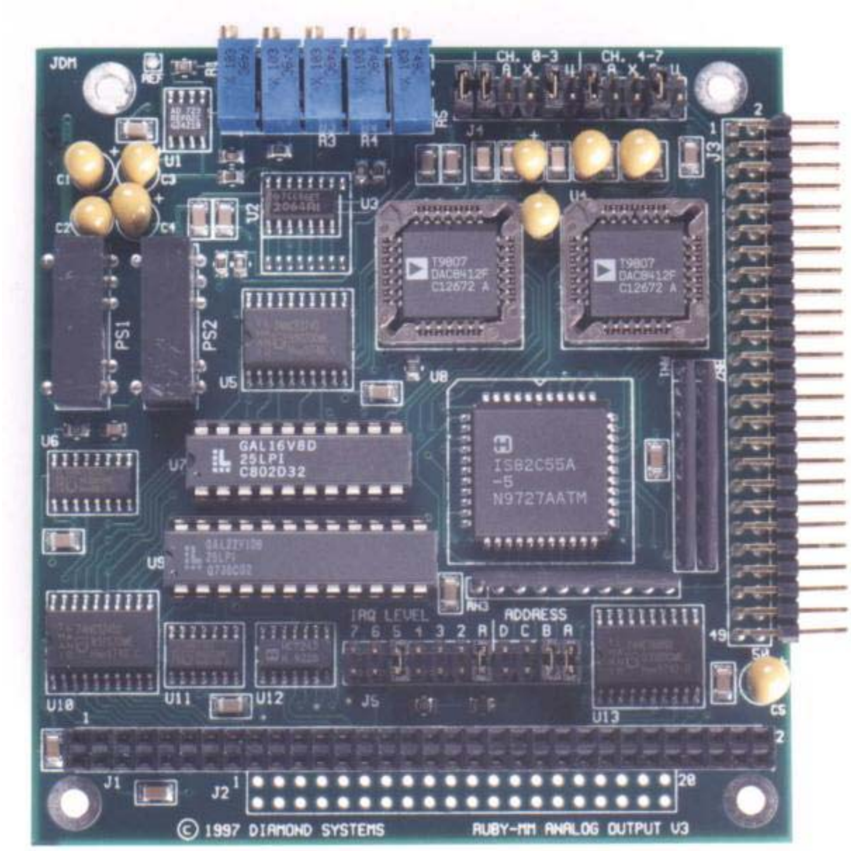
DIAMOND SYSTEMS CORPORATION

RUBY-MM

PC/104 4/8-Channel Analog Output Module

User Manual V3.21

Corresponds to Board Revision V3



Diamond Systems Corporation
8430-D Central Ave.
Newark, CA 94560 USA
Tel (510) 456-7800
Fax (510) 456-7878
techinfo@diamondsystems.com

Table of Contents

1. DESCRIPTION ... 3
2. I/O HEADER PINOUT ... 4
3. BOARD CONFIGURATION ... 5
   - Base Address ... 5
   - Interrupt Level Setting ... 5
4. ANALOG OUTPUT RANGE CONFIGURATION ... 6
   - On-Board Reference Full-Scale Voltage Selection ... 6
   - D/A Full-Scale Voltage ... 6
   - Unipolar / Bipolar Output Range ... 6
   - Adjustable Reference Voltage ... 6
5. RUBY-MM BOARD DRAWING ... 8
6. I/O REGISTER MAP ... 9
7. REGISTER DEFINITIONS ... 10
8. CALIBRATION PROCEDURE ... 12
   - +5.000V Reference Voltage Adjust ... 12
   - +10.00V Reference Voltage Adjust ... 12
   - Adjustable Reference Adjust ... 12
   - Negative Full-Scale Reference Adjust, Channels 0 - 3 ... 12
   - Negative Full-Scale Reference Adjust, Channels 4 - 7 ... 12
9. D/A CODE COMPUTATION ... 13
   - Straight Binary Coding (for unipolar output ranges) ... 13
   - Offset Binary Coding (for bipolar output ranges) ... 14
10. ANALOG OUTPUT RANGES AND RESOLUTION ... 15
11. HOW TO GENERATE AN ANALOG OUTPUT ... 16
    - Write the LSB to the board ... 16
    - Write the MSB to the board ... 17
    - Update the selected D/A channel ... 17
    - Examples ... 18
12. HARDWARE INTERRUPTS AND TRIGGERS ... 19
    - Interrupt Operation ... 19
    - DAC Update with External Trigger ... 19
13. DIGITAL I/O OPERATION ... 20
14. SPECIFICATIONS ... 21
15. 82C55 DIGITAL I/O CHIP DATASHEET ... 22

1. DESCRIPTION

Ruby-MM is a PC/104-format data acquisition board that provides analog outputs and digital I/O for process control and other applications. Below is a summary of key features:

Analog Outputs

Ruby-MM has 4 or 8 analog voltage outputs with 12-bit resolution (1 part in 4096).

⇒ **Note:** Analog output, D/A, and DAC are all used interchangeably in this manual.

Multiple Full-Scale Output Ranges

Six different preset ranges are available, including both bipolar and unipolar ranges.

Adjustable Full-Scale Output Range

One of the preset ranges (2.5V full-scale) can be adjusted by the user to any voltage between 0V and 5V.

External Reference Inputs

In addition to the preset ranges, two pins are provided on the user I/O header for additional custom ranges. One external reference pin is used for each bank of 4 D/A channels. Both unipolar and bipolar output ranges can be derived from each external reference input.

Simultaneous Update

Analog outputs are updated simultaneously in banks of 4 or all 8 at once. This prevents time skew errors which can result from updating outputs sequentially on a system which requires two or more control signals to change simultaneously.

External Trigger

An external trigger signal can be connected to the board. This trigger can be used to update the analog outputs, or it can be used to generate hardware interrupts on the PC/104 bus.

Digital I/O

An 82C55 chip is included to provide 24 lines of digital I/O. Each line has a 10K□ pull-up resistor. Each line is CMOS / TTL compatible and can supply up to ±6mA of current.

+5V Operation

Ruby-MM requires only +5VDC from the system power supply for operation. It generates its own ±15V supplies for the analog circuitry on board using two miniature DC/DC converters.

2. I/O HEADER PINOUT

Ruby-MM provides a 50-pin right-angle header labeled J3 for all user I/O. This header is located on the right side of the board. Pins 1, 2, 49, and 50 are marked to aid in proper orientation. A standard 50-pin cable-mount IDC (insulation displacement contact) connector will mate with this header.

J3
(Top of board)

Agnd	1	2	Vout 0
Agnd	3	4	Vout 1
Agnd	5	6	Vout 2
Agnd	7	8	Vout 3
Agnd	9	10	Vout 4*
Agnd	11	12	Vout 5*
Agnd	13	14	Vout 6*
Agnd	15	16	Vout 7*
External Ref. A	17	18	External Ref. B
Agnd	19	20	+15V
-15V	21	22	Agnd
Dgnd	23	24	External Trigger
A7	25	26	A6
A5	27	28	A4
A3	29	30	A2
A1	31	32	A0
C7	33	34	C6
C5	35	36	C4
C3	37	38	C2
C1	39	40	C0
B7	41	42	B6
B5	43	44	B4
B3	45	46	B2
B1	47	48	B0
+5V	49	50	Dgnd

Signal Name	Definition
Vout7 - 0	Analog output channels; * means present only on 8-channel version
Agnd	Analog ground
Dgnd	Digital ground
A7 - A0	Digital I/O port A
B7 - B0	Digital I/O port B
C7 - C0	Digital I/O port C
External Ref. A, B	External reference voltage inputs for custom D/A full-scale ranges; A is used for channels 0 - 3; B is used for channels 4 - 7
External Trigger	Input for external control of D/A updating and hardware interrupts
±15V	Analog power supply; maximum current draw 10mA per line
+5V	Connected to PC/104 bus power supply

3. BOARD CONFIGURATION

Refer to the Drawing of Ruby-MM on Page 8 for locations of headers described in Chapters 3 and 4.

Base Address

Each board in the system must have a different base address. Ruby-MM's base address is set with jumpers in the four rightmost positions of header **J5**, located in the lower section of the board near the PC/104 bus header. These four positions are labeled **D C B A**.

Each of these four locations corresponds to a different address bit in the base address; locations D - A correspond to address bits 9 - 6, respectively. A location where the jumper is **Out** is equal to a **1** for the corresponding address bit, and a location where the jumper is **In** is equal to a **0**. All address bits not selected (bits 5 - 0) are assumed to be 0 for the base address. Thus 16 different addresses can be selected. The table below lists recommended the 8 recommended base address settings for Ruby-MM (it is recommended to avoid addresses below 200 Hex). The default setting is 300 Hex.

Table 3.1: Base Address Configuration

Base Address		**Header J5 Position**			
Hex	**Decimal**	**D**	**C**	**B**	**A**
200	512	Out	In	In	In
240	576	Out	In	In	Out
280	640	Out	In	Out	In
2C0	704	Out	In	Out	Out
300	768 (Default)	Out	Out	In	In
340	832	Out	Out	In	Out
380	896	Out	Out	Out	In
3C0	960	Out	Out	Out	Out

Interrupt Level Setting

Ruby-MM can be programmed to generate interrupts in response to edges on the External Trigger signal. Interrupt levels 2 - 7 may be used. In addition, the user may connect the chosen interrupt level to a 1K□ pull-down resistor to enable interrupt sharing in conformance with the PC/104 standard. When an interrupt is pending, the selected interrupt line will be driven high by the board. When an interrupt is not pending, the selected interrupt line is tri-stated, and the pull-down resistor will drive the line low, while still allowing it to be driven high by a second board on the same line.

To select an interrupt level, install a jumper in locations **2 - 7** on header **J5**. Only one jumper can be installed in this group of locations. To connect the pull-down resistor, install a jumper in the **R** location of J5. The pull-down resistor must be installed unless another board in the system is using the same interrupt level and that board has the resistor connected.

4. ANALOG OUTPUT RANGE CONFIGURATION

Refer to the Drawing of Ruby-MM on Page 8 for locations of headers described in Sections 3 and 4.

Refer to Figure 4.1 on Page 7 for an explanation of the voltage reference circuitry. Also refer to Table 4.1 for a quick guide to output range configuration and jumper settings.

Header **J4** is used to configure the analog outputs. Four items are configurable: (1) On-board reference full-scale voltage, (2) D/A full-scale voltage, (3) unipolar / bipolar select, and (4) adjustable reference voltage. Items 2 and 3 in turn are configured separately for each bank of 4 analog output channels.

On-Board Reference Full-Scale Voltage Selection

An on-board reference voltage generator provides a +5.000V full-scale voltage output. This voltage is used as the basis for all on-board full-scale output ranges. This +5 reference drives an operational amplifier, from which the fixed references are derived. The gain of this amplifier is normally set to 1, so that its output is also +5.000V. However, you can change the gain to 2 so that the output is +10.00V. For an output of +5V, install a jumper in location **5** in header J4. For an output of +10V, remove the jumper from this location. The output of this amplifier is used to generate the full-scale voltages for both bipolar and unipolar output ranges.

D/A Full-Scale Voltage

The full-scale voltage defines the full output range capability of the analog outputs. Locations **F A X** on header **J4** are used to select the full-scale voltage. Each bank of four channels has its own selection pins for full-scale voltage. Thus each bank of four channels may be configured differently. Install only one jumper in these locations for each bank of channels. Position **F** is for the Full-scale voltage (5V or 10V depending on the jumper in position 2, explained above). This is the default setting. Position **A** is for the Adjustable reference voltage (see section 4.4). Position **X** is for the External reference voltage. To use an external reference voltage, connect the voltage to the appropriate pin (A for channels 0 - 3, B for channels 4 - 7; see the I/O header pinout in Chapter 2 for pinout information), and install a jumper in the **X** location of J4.

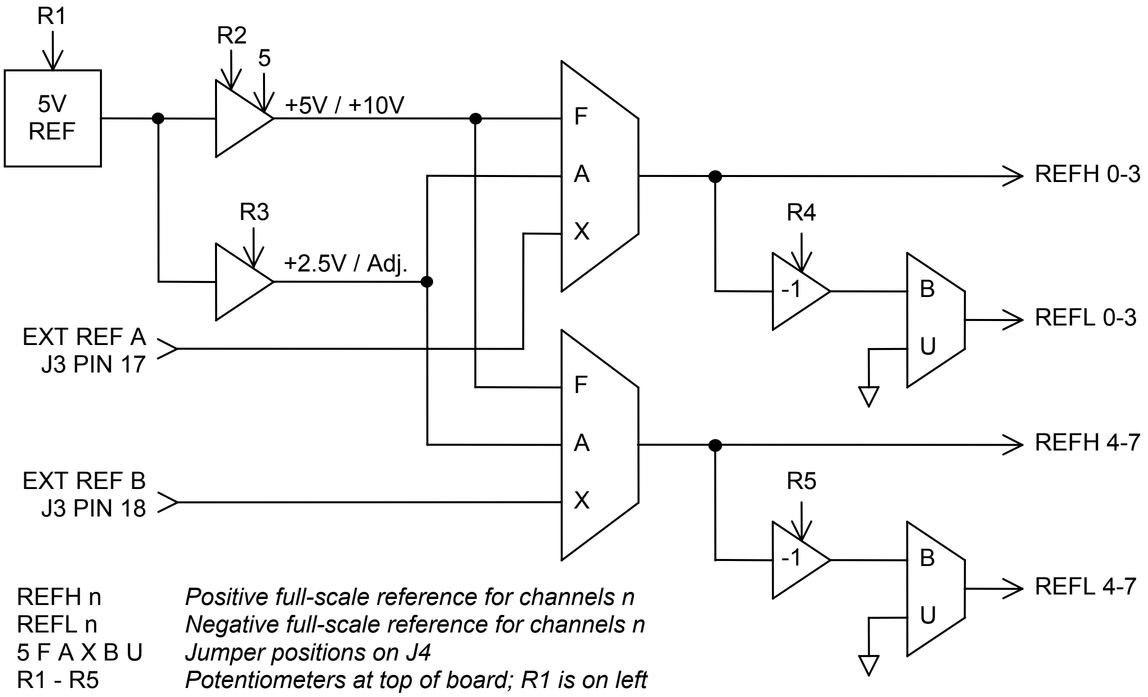
Unipolar / Bipolar Output Range

Unipolar output ranges are positive voltages only (for example 0 - 5V), while bipolar output ranges include both positive and negative voltages (for example ±5V). To select unipolar outputs, install a jumper in position **U** on J4. to select bipolar outputs, install a jumper in position **B**. Install only one jumper in these locations for each bank of channels.

Adjustable Reference Voltage

One full-scale voltage range is adjustable by the user. It is preset to 2.5V (for both 0-2.5V and ±2.5V ranges), but may be set anywhere between 0V and 5V. To adjust this voltage, apply a voltmeter to the top pin of header **J4** underneath either **A** mark and turn the screw on potentiometer **R4** (the fourth from the left / second from the right in the row of blue potentiometers at the top of the board) until the voltmeter reads the desired voltage.

Figure 4.1: Reference Voltage Circuit Block Diagram



REFH n	*Positive full-scale reference for channels n*
REFL n	*Negative full-scale reference for channels n*
5 F A X B U	*Jumper positions on J4*
R1 - R5	*Potentiometers at top of board; R1 is on left*

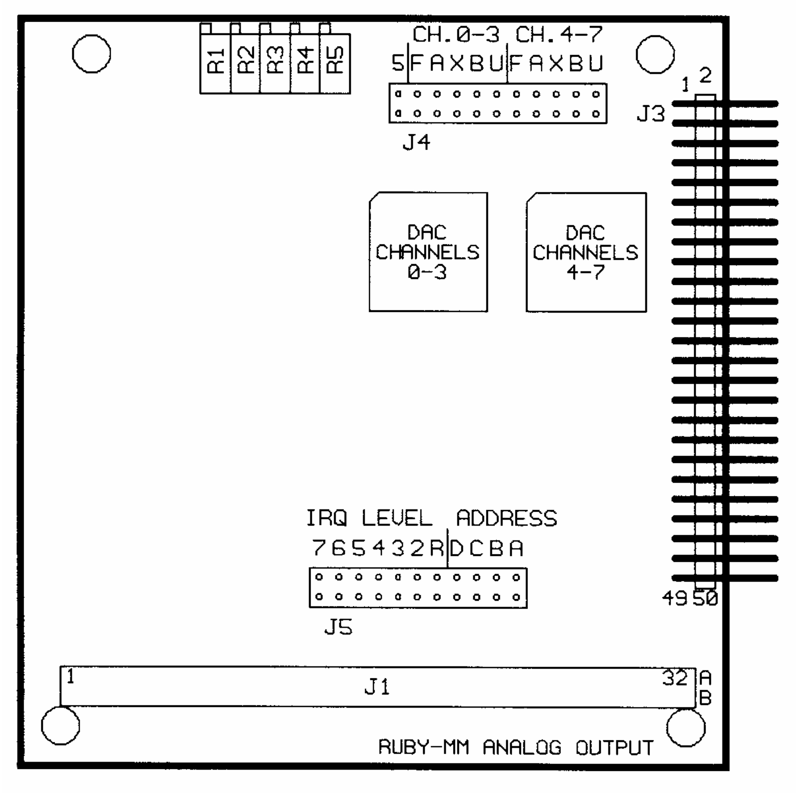
Table 4.1: Analog Output Configuration (Header J4)

Positions F A X B U are repeated
for each bank of output channels

Output Range	5	F	A	X	B	U	1 LSB is equal to
0 - 10V	Out	In	Out	Out	Out	In	2.44mV
0 - 5V	In	In	Out	Out	Out	In	1.22mV
Adjustable, Unipolar *(preset to 2.5V)*	In	Out	In	Out	Out	In	Full Scale / 4096
External Ref, Unipolar *(output range is 0-External Ref)*	In	Out	Out	In	Out	In	External Ref / 4096
±10V	Out	In	Out	Out	In	Out	4.88mV
±5V	In	In	Out	Out	In	Out	2.44mV
Adjustable, Bipolar *(preset to ±2.5V)*	In	Out	In	Out	In	Out	Full Scale / 2048
External Ref, Bipolar *(output range is ±External Ref)*	In	Out	Out	In	In	Out	External Ref / 2048

⇒ **Note:** Each bank of four channels (0 - 3 and 4 - 7) can have a different output range setting. However, all four channels within a bank will always have the same output range.

5. RUBY-MM BOARD DRAWING



J1:	PC/104 Bus Connection
J3:	I/O Header (2x25 pins)
J4:	Output Range Configuration Header
J5:	Interrupt and Base Address Configuration Header
R1 – R5:	Calibration Potentiometers

6. I/O REGISTER MAP

Ruby-MM occupies 16 consecutive 8-bit locations in I/O space. For example, the default base address is 300 Hex (768 Decimal); in this case the board occupies addresses 300 - 30F (768 - 783).

The first 8 locations are used individually for each analog output channel. Since analog output data is 12 bits wide, it is broken into two bytes. The first byte contains the 8 least significant bits (called the LSB) of the D/A data, and the 4 lowest bits of the second byte contain the 4 most significant bits (called the MSB) of the D/A data. The 4 highest bits of the second byte are not used. The LSB is always written to the same location on the board, Base + 8, regardless of which channel it is intended for. When the MSB is written to the appropriate address, the LSB is directed to that channel's register automatically.

The DACs can be updated all at once or in groups of 4 channels, depending on which location is read from. See Read Function, Base + 8 through Base + 10, in the table below. The value read from these locations is not predictable and not meaningful. Only the act of reading from the board is required to perform the indicated operation.

Reading from or writing to any location on the board resets the interrupt request flip flop.

Base +	Write Function	Read Function
0	Channel 0 MSB	N/A
1	Channel 1 MSB	N/A
2	Channel 2 MSB	N/A
3	Channel 3 MSB	N/A
4	Channel 4 MSB	N/A
5	Channel 5 MSB	N/A
6	Channel 6 MSB	N/A
7	Channel 7 MSB	N/A
8	D/A LSB register	Update all channels 0 - 7
9	Board reset	Update channels 0 - 3
10	Configuration register	Update channels 4 - 7
11	N/A	N/A
12	Digital I/O port A data	Digital I/O port A data
13	Digital I/O port B data	Digital I/O port B data
14	Digital I/O port C data	Digital I/O port C data
15	Digital I/O control register	Digital I/O control register

Reset information:

Writing to or reading from any address on the board resets the interrupt request flip flop.

A system hardware reset will also reset the board.

During a reset, the following occurs:

- All analog outputs are set to mid-scale (0V for bipolar ranges and 1/2 full-scale for unipolar ranges).
- The control register is set to 0, disabling all external trigger and interrupt functions.
- All digital I/O ports are set to input mode.

The next chapter describes all registers on the board. You should familiarize yourself with these registers in order to get a complete understanding of the board's operation.

7. REGISTER DEFINITIONS

Base + 0 through Base + 7 Write DAC 0 - 7 MSB

Bit No.	7	6	5	4	3	2	1	0
Name	X	X	X	X	DA11	DA10	DA9	DA8

X	Unused bits
DA11 - 8	D/A bits 11 - 8; DA11 is the most significant bit; data is an unsigned 12-bit value.

1. Write the LSB to the LSB register at Base + 8 (see below).
2. Write the MSB to the individual channel's register, Base + 0 through Base + 7.
3. Perform an update, either through software or via a hardware trigger, thus setting the channel's output to the desired voltage.

⇒ **Note:** The LSB (see below) **must** be written to the board **before** the MSB, since writing the MSB causes the contents of the LSB register to be loaded into the DAC along with the MSB.

Base + 8 Write DAC LSB

An 8-bit register at Base + 8 is used to store the low 8 bits (LSB or least significant byte) of the D/A code for any channel. Writing to any channel is a three-step process:

Bit No.	7	6	5	4	3	2	1	0
Name	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0

Definitions:

DA7-0 D/A data bits 7 - 0; DA0 is the least significant bit; data is an unsigned 12-bit value.

Base + 9 Write Board Reset

This address does not contain a data register. It is simply used to reset the board. Any value written to this address will reset the board, causing the following to occur:

- All analog outputs are set to mid-scale (0V for bipolar ranges and 1/2 full-scale for unipolar ranges).
- The control register is set to 0, disabling all external trigger and interrupt functions.
- All digital I/O ports are set to input mode.

The same reset operation will occur during system reset.

Base + 10 Write Configuration Register

A configuration register at Base + 10 is used to control external trigger enabling, trigger source, and interrupt operation. Bit 0 determines whether interrupts are enabled onto the PC bus in response to active trigger edges. Active trigger edges are rising edges. Bit 1 selects the source for the trigger. Bits 2-3 determine whether the D/A channels are updated when the active trigger edge occurs. This provides complete control over external trigger operations and interrupts.

Bit No.	7	6	5	4	3	2	1	0
Name	X	X	X	X	TRIGL47	TRIGL03	TRIGSEL	INTE

Definitions:

X Unused bits

TRIGL47 Trigger latch enable for channels 4 - 7: 0 = disabled, 1 = enabled

TRIGL03 Trigger latch enable for channels 0 - 3: 0 = disabled, 1 = enabled

These two lines determine whether the D/A channels may be updated upon an active trigger edge (rising edge of the trigger selected with TRIGSEL).

TRIGSEL Interrupt / trigger source select:

0 = external trigger (pin 24 on the I/O header)

1 = digital I/O bit C3 (pin 37 on the I/O header or controlled in software at base+14 bit 3)

⇒ **Note:** If an external signal is attached to pin C3, then the lower half of port C must be in input mode (see the 82C55 datasheet at the back of this manual or the table on Page 20 for information on port direction programming).

INTE Interrupt enable: 0 = disabled, 1 = enabled

Interrupt operations use the trigger source selected with TRIGSEL.

Base + 11 N/A N/A

This address is not used on Ruby-MM. The only result of writing to or reading from this address is that the interrupt request flip flop will be reset.

Base + 12 through Base + 15 Read/Write 82C55 Digital I/O Registers

These registers map directly to the 82C55. The definitions of these registers can be found in the 82C55 datasheet appended to the back of this manual.

8. CALIBRATION PROCEDURE

Calibration requires a voltmeter (at least 4 1/2 digits of precision is preferred) and a miniature screwdriver to turn the potentiometer screws. The common lead of the voltmeter must be connected to analog ground (not digital ground). The best source for this connection is any of the analog ground pins on the user I/O header J3.

⇒ **Note:** All steps should be completed in the sequence shown, since each step affects the following steps. (Steps 4 and 5 may be interchanged since they do not depend on each other.)

During calibration, each bank of output channels should have jumpers installed in a valid configuration. In For each bank, one jumper should be installed in the F position, and one jumper should be installed in either the B or U position. If you are using the adjustable (A) reference, you can switch to that range after calibration.

+5.000V Reference Voltage Adjust

Install a jumper in position **"5"** on J4. Connect the high side lead of the voltmeter to the jumper that is in the F position. Adjust **R1** so that the voltmeter reads +5.000V.

+10.00V Reference Voltage Adjust

Keep the voltmeter connected as above. Remove the jumper in position **"5"** on J4 and adjust **R3** so that the voltmeter reads +10.000V.

Adjustable Reference Adjust

This step can be skipped if you are not using the adjustable reference.

Connect the voltmeter to the pin immediately below either **"A"** on J4 (the pin in the top row of the header). Adjust **R2** so that the voltmeter reads the desired full-scale voltage range. This voltage is factory-preset to 2.500V.

Negative Full-Scale Reference Adjust, Channels 0 - 3

Install jumpers in positions **"5"** and the *leftmost* **"F"** on J4. Connect the voltmeter to the pin under the *leftmost* **"B"** on J4 (the pin in the top row of the header). Adjust **R4** so that the voltmeter reads -5.000V.

Negative Full-Scale Reference Adjust, Channels 4 - 7

Install jumpers in positions **"5"** and the *rightmost* **"F"** on J4. Connect the voltmeter to the pin under the *rightmost* **"B"** on J4 (the pin in the top row of the header). Adjust **R5** so that the voltmeter reads -5.000V.

9. D/A CODE COMPUTATION

Two different methods are used to compute the 12-bit D/A code used for analog output operations.

For *unipolar* output ranges (positive voltages only), *straight binary coding* is used.

For *bipolar* output ranges (both positive and negative voltages), *offset binary coding* is used.

For any output range, the resolution is equal to the maximum possible range of output voltages divided by the maximum number of possible steps. For a 12-bit D/A converter as is used on Ruby-MM, the maximum number of steps is 2^{12} = 4096 (the actual output codes range from 0 to 4095, which is the full range of possible 12-bit binary numbers). Thus the resolution is equal to 1/4096 times the full-scale range. This is the smallest possible change in the output and corresponds to a change of 1 in the output code. Because of this fact the resolution is often referred to as the value of **1 LSB**, or 1 least significant bit.

Straight Binary Coding (for unipolar output ranges)

This is the simplest form of binary coding. The output voltage is given by:

Output Voltage = (Output Code / 4096) x Full-Scale Voltage

Example: Output code = 1024, full-scale voltage = 5V

Output voltage = (1024 / 4096) x 5 = .25 x 5 = **1.250V**

Conversely, the output code for a desired output voltage is given by:

Output Code = (Desired Output Voltage / Full-Scale Voltage) * 4096

Example: Desired output voltage = 0.485V, Full-scale voltage = 2.5V

Output Code = (0.485 / 2.5) x 4096 = 0.194 x 4096 = **795** (rounded up)

The relationship between D/A resolution and Full-scale voltage is:

1 LSB = 1/4096 x Full-Scale Voltage

Example: Full-scale voltage = 5V; 1 LSB = 5V / 4096 = 1.22mV

Here is a brief overview of the relationship between output code and output voltage:

Output Code	Output Voltage	Example Output for 0-5V Range
0	0V	0V
1	1 LSB	.0024V (2.44mV)
2048	1/2 positive full scale	2.5V
4095	Positive full scale - 1 LSB	4.9988V

⇒ **Note:** In order to generate an output voltage of positive full scale, you would have to output a code of 4096 (4096 / 4096 x full-scale = full-scale). However, 4096 is a 13-bit number which cannot be reproduced on a 12-bit D/A converter. The highest number that can be output is 4095, which is 4096 - 1. This results in a maximum output voltage of full scale minus 1 LSB for any analog output range.

Offset Binary Coding (for bipolar output ranges)

This method takes into account the fact that the lowest output voltage is not zero but a negative value. The output voltage is given by:

Output Voltage = (Output Code / 2048) x Full-Scale Voltage - Full-Scale Voltage

Example: Output code = 1024, full-scale voltage = 5V

Output voltage = (1024 / 2048) x 5 - 5 = (0.5 x 5) - 5 = **-2.500V**

Note the difference between this output voltage to the output voltage using straight binary coding shown above using the same output code.

Conversely, the output code for a desired output voltage is given by:

Output Code = (Desired Output Voltage / Full-Scale Voltage) * 2048 + 2048

Example: Desired output voltage = 0.485V, Full-scale voltage = 2.5V

Output Code = (0.485 / 2.5) x 2048 + 2048 = 0.194 x 2048 + 2048 = **2445** (rounded down)

The relationship between D/A resolution and Full-scale voltage is:

1 LSB = 1/2048 x Full-Scale Voltage

Example: Full-scale voltage = 5V; 1 LSB = 5V / 2048 = 2.44mV

The reason that 1 LSB for a bipolar range is twice the magnitude of 1 LSB for a unipolar range with the same full-scale voltage is that for the bipolar range, the full voltage span is twice the magnitude. For example, a unipolar range with a full-scale voltage of 5V has a range of 0V to 5V, for a total span of 5V. However, a bipolar range with a full-scale voltage of 5V has a range of ±5V, for a total span of 10V.

Here is a brief overview of the relationship between output code and output voltage:

Output Code	Output Voltage	Example Output for ±5V Range
0	Negative full scale	-5V
1	Negative full scale + 1 LSB	-4.9976V
2047	-1 LSB	-.0024V (-2.44mV)
2048	0V	0V
2049	+1 LSB	+.0024V (+2.44mV)
4095	Positive full scale - 1 LSB	+4.9976V

⇒ **Note:** Again, an output code of 4096 would be required to generate the positive-full-scale output voltage, but since that is impossible, the maximum output voltage is 1 LSB less then positive full scale.

10. ANALOG OUTPUT RANGES AND RESOLUTION

The table below lists the available fixed full-scale output ranges and their corresponding actual full-scale voltage ranges and resolution.

For any output range, the resolution is equal to the maximum possible range of output voltages divided by the maximum number of possible steps. For a 12-bit D/A converter as is used on Ruby-MM, the maximum number of steps is 2^{12} = 4096 (the actual output codes range from 0 to 4095, which is the full range of possible 12-bit binary numbers). Thus the resolution is equal to 1/4096 times the full-scale range. This is the smallest possible change in the output and corresponds to a change of 1 in the output code. Because of this fact the resolution is often referred to as the value of **1 LSB**, or 1 least significant bit.

Table 7.1: Analog Output Ranges and Resolution

Full-Scale Voltage	Unipolar or Bipolar	Range Name	Negative Full Scale	Positive Full Scale	Resolution (1LSB)
10V	Unipolar	0-10V	0V	+9.9976V	2.44mV
5V	Unipolar	0-5V	0V	+4.9988V	1.22mV
2.5V	Unipolar	0-2.5V	0V	+2.4994V	0.61mV
10V	Bipolar	±10V	-10V	+9.9951V	4.88mV
5V	Bipolar	±5V	-5V	+4.9963V	2.44mV
2.5V	Bipolar	±2.5V	-2.5V	+2.4988V	1.22mV

In the table above, *negative full scale* refers to the output voltage for a code of 0, and *positive full scale* refers to the output voltage for a code of 4095.

11. HOW TO GENERATE AN ANALOG OUTPUT

This chapter describes how to generate an analog output directly (without the use of the driver software).

Ruby-MM has 12-bit resolution analog outputs. However, data is written to the board in 8-bit bytes. Therefore two bytes must be written to the board to generate a single analog output. In addition, many applications require several channels to be updated simultaneously. In order to provide this ability, the update operation is separate from the data write operation.

Thus there are three steps required to generate an analog output. Each step is described in detail. The steps must be completed in the sequence shown below.

To generate an analog output on a single channel:

1. Write the LSB (least significant byte) to the board
2. Write the MSB (most significant byte) to the board
3. Update the bank of channels containing the selected D/A channel

To generate analog outputs on several channels at once:

To write to several channels at once, the update can be performed just once at the end:

1. Write the LSB (least significant byte) for the first channel to the board
2. Write the MSB (most significant byte) for the first channel to the board
3. Write the LSB (least significant byte) for the second channel to the board
4. Write the MSB (most significant byte) for the second channel to the board
5. Update the bank(s) of channels containing the selected D/A channels

Write the LSB to the board

One register is provided to hold the LSB data for all analog outputs. This is done in order to conserve I/O space. If individual LSB and MSB registers were provided for each channel, Ruby-MM would require a total of 16 addresses for the D/A + additional data and control registers, resulting in a total I/O space requirement of 32 bytes. In order to reduce the I/O footprint of the board, a single LSB register is provided for all channels. Each channel is still completely independent since the register is written with new data each time a channel is updated.

The LSB register is located at address **base + 8**. See page 10 for a description of this register.

To calculate the LSB, use the following formula:

LSB = Data AND 255

This strips off the highest 4 bits and keeps only the lowest 8 bits for output to the LSB register. On Ruby-MM this operation isn’t actually necessary, since the board has only an 8-bit data bus, and all I/O commands are 8 bits wide. This means that you can write the entire 12-bit D/A code to the LSB register, and the board will automatically strip off the 4 high bits and keep only the 8 low bits.

Write the MSB to the board

Each D/A channel has its own MSB register, located at address **base + (channel no.)**. Only the four least significant bits of the register are used to hold the 4 most significant bits of the D/A code. See page 10 for a description of these registers.

Writing to this register causes the 4 most significant bits plus the contents of the MSB register (total 12 bits) to be written to the selected D/A channel. However, the D/A is not updated and will maintain its current output voltage. A separate update command must be given as described below.

To calculate the MSB, use the following formula:

MSB = (Data AND 3840) / 256

This strips off the low 8 bits, keeps only the 4 highest bits, and shifts them into the low 4 bits for output to the board.

Update the selected D/A channel

Once the LSB and MSB are written, the board has stored the data. However an update command must be given to actually set the output voltage to the given value. An update command can given in either software or hardware.

Software Update Command

A software update command can be given either to a group of four channels simultaneously (0 - 3 or 4 - 7), or to all 8 channels simultaneously.

To update all 8 channels simultaneously, read from the address **base + 8**.

To update channels 0 - 3 simultaneously, read from the address **base + 9**.

To update channels 4 - 7 simultaneously, read from the address **base + 10**.

In all these instances, the value read from the board is not predictable and is not meaningful. Only the act of reading from the indicated location is required to perform the selected operation.

Hardware Update Command

A hardware update command can occur in one of two ways:

- A rising edge on the external trigger, pin 24 on J3 (the user I/O header).
- A rising edge on digital I/O bit C3 (pin 37 on J3), in either input or output mode.

To use hardware updating, or triggering, you must program the control register at base + 10. This register defines which channels may be updated with the external trigger as well as the source of the trigger. See Chapter 3 for details.

⇒ **Note:** When a channel is updated, its output will change only if new data has been written to it since the last update. For example, if you do a simultaneous update on channels 0 - 3 but you only wrote data to channel 0, then only channel 0 will change, and channels 1 - 3 will stay the same.

⇒ **Note:** If hardware updating is enabled, software updating will still work.

Examples

Single channel output

Assume channels 0 - 3 are configured for 0-5V. To set channel 0 to 3V, do the following:

D/A code is 3V / 5V x 4096 = 2458 (value is rounded to nearest integer)

LSB = 2458 AND 255 = 154

MSB = (2458 AND 3840) / 256 = 9

Step 1. Write **154** to base + 8 (LSB register).

Step 2. Write **9** to base + 0 (Channel 0 MSB register). The value 2458 is written to DAC 0.

Step 3. Read from base + 0 (Channel 0 update address). DAC 0 now outputs 3.000V.

Two channel output

Assume channels 0 - 3 are configured for 0-5V. To set channel 0 to 3.8V and channel 1 to 1.5V, do the following:

D/A code for channel 0 = 3.8 / 5 x 4096 = 3113

LSB = 3113 AND 255 = 41

MSB = (3113 AND 3840) / 256 = 12

D/A code for channel 1 = 1.5 / 5 x 4096 = 1229

LSB = 1229 AND 255 = 205

MSB = (1229 AND 3840) / 256 = 4

Step 1. Write **41** to base + 8 (LSB register).

Step 2. Write **12** to base + 0 (Channel 0 MSB register). The value 3113 is written to DAC 0.

Step 3. Write **205** to base + 8 (LSB register).

Step 4. Write **4** to base + 1 (Channel 1 MSB register). The value 1229 is written to DAC 1.

Step 5. Read from base + 9 (Channels 0 - 3 simultaneous update). DAC 0 and DAC3 are both updated to their new output voltages. All other channels remain at their existing output voltages.

12. HARDWARE INTERRUPTS AND TRIGGERS

Interrupt Operation

Ruby-MM supports interrupt operation for analog output and digital I/O. With interrupts, you can cause analog outputs to change or digital I/O data to be transferred under control of an external signal or one of the digital I/O lines.

The control register at **base + 10** is used to configure interrupts. To enable interrupts, bit 0, INTE, must be set to 1. Then bit 1, TRIGSEL, selects the trigger / interrupt source. If it is 0, then pin 24 on the I/O header is used as the trigger. If it is 1, then bit C3 of the digital I/O, pin 37 on the I/O header, is used as the trigger. Note that if an external signal is attached to this pin, then the lower half of port C must be in input mode. See the 82C55 datasheet at the back of this manual or the table in the next chapter for information on port direction programming.

A rising edge on the selected trigger source acts as the trigger. Both Ext. Trigger (pin 24) and C3 (pin 37) have 10K□ pull-up resistors on them, so they are high by default.

The Ruby-MM driver software includes an interrupt routine for both digital I/O and analog output. Using this interrupt routine you can program Ruby-MM to transfer digital data into or out of your PC/104 system with an external trigger, or you can output analog voltages under external control. Refer to Functions 22, 47, and 48 in the Software Function Descriptions below.

DAC Update with External Trigger

The DACs can be updated using an external hardware trigger as well as in software. This can be useful for applications where the time of the update needs to be synchronized to an external event.

To enable DAC updating via hardware trigger, use the control register at base + 10. Bits 3 and 2 select whether channels 0-3 and 4-7 respectively can be updated using a hardware trigger. A 1 in the corresponding bit enables hardware updating for that bank of 4 channels. Bit 1 is used to select the trigger source. A 1 selects digital I/O bit C3 (pin 37 on the I/O header) as the trigger source, and a 0 selects the dedicated external trigger line on pin 24.

Note that if a software update command is issued to a channel, then hardware updating will not affect that channel again until new data is written to it. The same is true in reverse. Only one update command is effective for any data value written to the DAC; subsequent update commands have no effect on the DAC until new data is written to it.

⇒ **Note:** When hardware triggering is enabled, software updating will still work. Software updating is always enabled on Ruby-MM.

13. DIGITAL I/O OPERATION

Ruby-MM contains an 82C55 chip for digital I/O. This chip provides 3 8-bit ports, called A, B, and C, for a total of 24 I/O lines. The chip contains four registers, 1 each for A, B, and C and 1 for control. These four registers are mapped to Ruby-MM's I/O space at base + 12 through base + 15. The 24 I/O lines are brought out to pins 25 - 48 on the user I/O header J3. Each line has a 10K□ pull-up resistor so that when the port is configured for input mode, the line is pulled up to a logic 1 level.

The I/O lines are standard CMOS logic with ±2.5mA output current capability. If more output current is required, use a buffer chip such as 74F244.

Each port's direction is determined through a control register. In normal "Mode 0" operation (the most common operating mode), ports A and B can be independently configured for input or output, and each half of port C can be independently configured for input or output. Output data is latched in the chip, but input data is not latched. When reading a port that is in input mode, the current logic levels of the port at the time of the read operation will be returned. To latch data into the port, you must use Modes 1 or Mode 2. In "Mode 1" and "Mode 2" operation, A and B are again configurable for input/output, but port C is reconfigured to provide control signals for transfer requests and data latching. A complete 82C55 datasheet is included at the end of this manual; please refer to it for programming details.

At power up, hardware reset, or board reset (write to base + 9), the 82C55 is set to all input and the data registers for ports A, B, and C are set to 0. When this happens, all I/O lines will have a logic 1 level because of the pull-up resistors.

When a port's direction is set to output, its data register is cleared to 0.

Here is a quick list of configuration bytes for some common Mode 0 I/O configurations:

Configuration Byte				
Hex	**Decimal**	**Port A**	**Port B**	**Port C (both halves)**
9B	155	Input	Input	Input
92	146	Input	Input	Output
99	153	Input	Output	Input
90	144	Input	Output	Output
8B	139	Output	Input	Input
82	130	Output	Input	Output
89	137	Output	Output	Input
80	128	Output	Output	Output

14. SPECIFICATIONS

Analog Outputs

No. of outputs	4 or 8, voltage output
Resolution	12 bits (1 part in 4096)
Fixed output ranges	0 - 5V, 0 - 10V unipolar, ±5V, ±10V bipolar
Adjustable output range	Preset to 2.5V for 0 - 2.5V, ±2.5V output ranges Can be adjusted anywhere between 0V and 5V
External reference	0V min, 10V max
Settling time	6□s max to ±.01%
Accuracy	±1LSB
Integral nonlinearity	±1LSB max
Differential nonlinearity	-1LSB max, guaranteed monotonic
Output current	±5mA max per channel
Minimum output load	2K□
Update method	Simultaneous update in groups of 4 (Channels 0-3 and 4-7) or all 8
Reset	All DACs reset to mid-scale (0V for bipolar ranges, 1/2 full-scale for unipolar ranges)

Digital I/O

No. of lines	24
Compatibility	CMOS / TTL
Input voltage	Logic 0: -0.5V min, 0.8V max Logic 1: 2.0V min, 5.5V max
Output voltage	Logic 0: 0.0V min, 0.4V max Logic 1: 3.0V min, Vcc - 0.4V max
Output current	±2.5mA max per line
Pull-up resistor	10K□ resistor on each I/O line
External trigger	TTL / CMOS compatible, 10K□ pull-up resistor, active high edge
Reset	All digital I/O lines are set to input and all data registers are set to 0

Miscellaneous

Power supply (Vcc)	+5VDC ±10%
Current requirement	4-channel board: 220mA typical, all outputs open 8-channel board: 290mA typical, all outputs open
Operating temperature	0 to 70^{o}C standard; -40 to +85^{o}C optional
Operating humidity	5 to 95% non-condensing
Size	3.55” x 3.775”
Data bus	8 bits (16-bit header can be installed for pass-through function but is not used on board)

HARRIS SEMICONDUCTOR

82C54

March 1997

CMOS Programmable Interval Timer

Features

- **8MHz to 12MHz Clock Input Frequency**
- **Compatible with NMOS 8254**
  - **Enhanced Version of NMOS 8253**
- **Three Independent 16-Bit Counters**
- **Six Programmable Counter Modes**
- **Status Read Back Command**
- **Binary or BCD Counting**
- **Fully TTL Compatible**
- **Single 5V Power Supply**
- **Low Power**
  - **ICCSB 10µA**
  - **ICCOP . . . . .10mA at 8MHz**
- **Operating Temperature Ranges**
  - **C82C54 . . . . .0°C to +70°C**
  - **I82C54 . . . . -40°C to +85°C**
  - **M82C54 . . . . -55°C to +125°C**

Description

The Harris 82C54 is a high performance CMOS Programmable Interval Timer manufactured using an advanced 2 micron CMOS process.

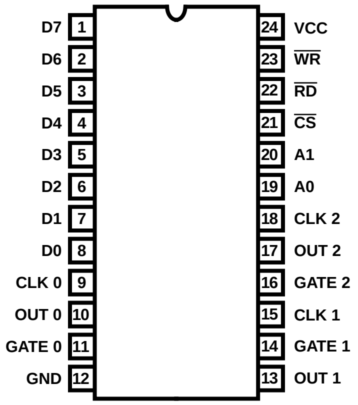
The 82C54 has three independently programmable and functional 16-bit counters, each capable of handling clock input frequencies of up to 8MHz (82C54) or 10MHz (82C54-10) or 12MHz (82C54-12).

The high speed and industry standard configuration of the 82C54 make it compatible with the Harris 80C86, 80C88, and 80C286 CMOS microprocessors along with many other industry standard processors. Six programmable timer modes allow the 82C54 to be used as an event counter, elapsed time indicator, programmable one-shot, and many other applications. Static CMOS circuit design insures low power operation.

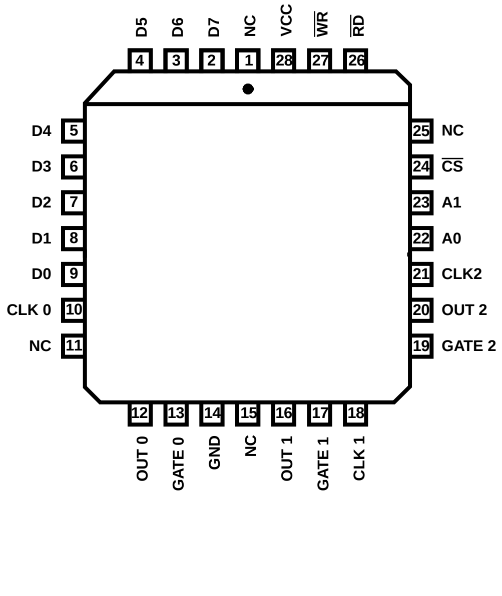
The Harris advanced CMOS process results in a significant reduction in power with performance equal to or greater than existing equivalent products.

Pinouts

82C54 (PDIP, CERDIP, SOIC)
TOP VIEW



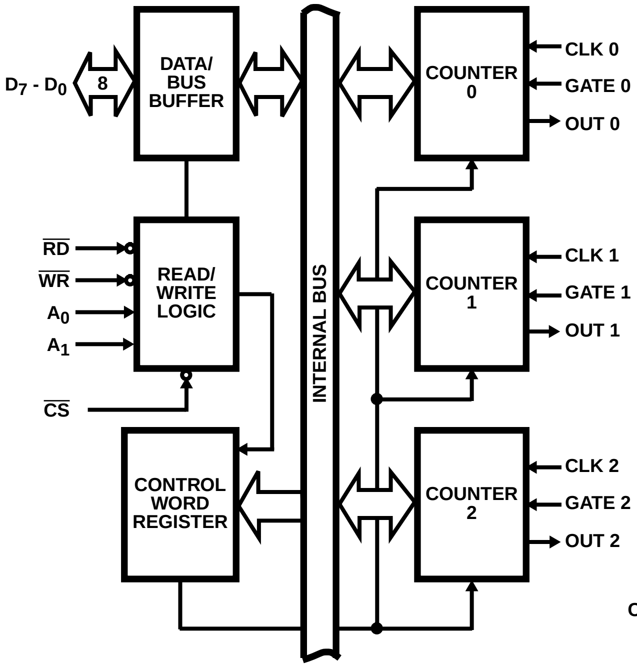
82C54 (PLCC/CLCC)
TOP VIEW

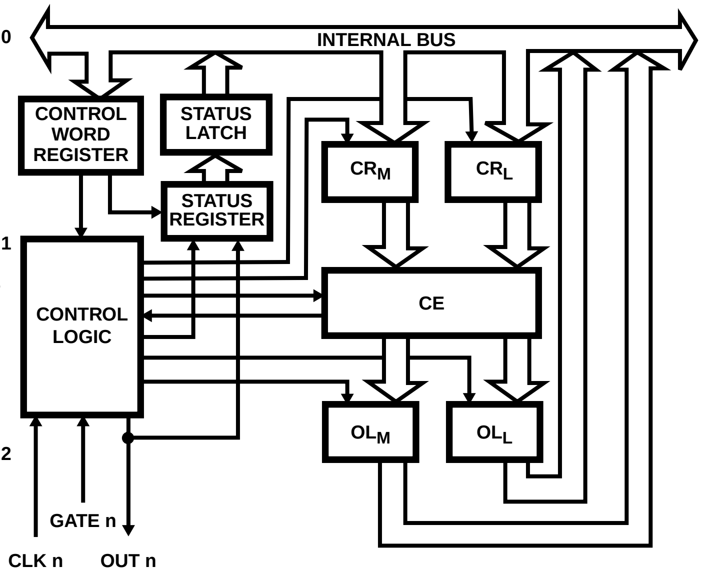


Ordering Information

PART NUMBERS			TEMPERATURE RANGE	PACKAGE	PKG. NO.
8MHz	10MHz	12MHz			
CP82C54	CP82C54-10	CP82C54-12	0°C to +70°C	24 Lead PDIP	E24.6
IP82C54	IP82C54-10	IP82C54-12	-40°C to +85°C	24 Lead PDIP	E24.6
CS82C54	CS82C54-10	CS82C54-12	0°C to +70°C	28 Lead PLCC	N28.45
IS82C54	IS82C54-10	IS82C54-12	-40°C to +85°C	28 Lead PLCC	N28.45
CD82C54	CD82C54-10	CD82C54-12	0°C to +70°C	24 Lead CERDIP	F24.6
ID82C54	ID82C54-10	ID82C54-12	-40°C to +85°C	24 Lead CERDIP	F24.6
MD82C54/B	MD82C54-10/B	MD82C54-12/B	-55°C to +125°C	24 Lead CERDIP	F24.6
MR82C54/B	MR82C54-10/B	MR82C54-12/B	-55°C to +125°C	28 Lead CLCC	J28.A
SMD # 8406501JA	-	8406502JA	-55°C to +125°C	24 Lead CERDIP	F24.6
SMD# 84065013A	-	84065023A	-55°C to +125°C	28 Lead CLCC	J28.A
CM82C54	CM82C54-10	CM82C54-12	0°C to +70°C	24 Lead SOIC	M24.3

Functional Diagram





COUNTER INTERNAL BLOCK DIAGRAM

Pin Description

SYMBOL	DIP PIN NUMBER	TYPE	DEFINITION
D7 - D0	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	OUT 0: Output of Counter 0.
GATE 0	11	I	GATE 0: Gate input of Counter 0.
GND	12		GROUND: Power supply connection.
OUT 1	13	O	OUT 1: Output of Counter 1.
GATE 1	14	I	GATE 1: Gate input of Counter 1.
CLK 1	15	I	CLOCK 1: Clock input of Counter 1.
GATE 2	16	I	GATE 2: Gate input of Counter 2.
OUT 2	17	O	OUT 2: Output of Counter 2.

Pin Description (Continued)

<table>
<tr><th>SYMBOL</th><th>DIP PIN NUMBER</th><th>TYPE</th><th>DEFINITION</th></tr>
<tr><td>CLK 2</td><td>18</td><td>I</td><td>CLOCK 2: Clock input of Counter 2.</td></tr>
<tr><td>A0, A1</td><td>19 - 20</td><td>I</td><td>ADDRESS: Select inputs for one of the three counters or Control Word Register for read/write operations. Normally connected to the system address bus.
<table>
<tr><th>A1</th><th>A0</th><th>SELECTS</th></tr>
<tr><td>0</td><td>0</td><td>Counter 0</td></tr>
<tr><td>0</td><td>1</td><td>Counter 1</td></tr>
<tr><td>1</td><td>0</td><td>Counter 2</td></tr>
<tr><td>1</td><td>1</td><td>Control Word Register</td></tr>
</table></td></tr>
<tr><td>$\overline{CS}$</td><td>21</td><td>I</td><td>CHIP SELECT: A low on this input enables the 82C54 to respond to $\overline{RD}$ and $\overline{WR}$ signals. $\overline{RD}$ and $\overline{WR}$ are ignored otherwise.</td></tr>
<tr><td>$\overline{RD}$</td><td>22</td><td>I</td><td>READ: This input is low during CPU read operations.</td></tr>
<tr><td>$\overline{WR}$</td><td>23</td><td>I</td><td>WRITE: This input is low during CPU write operations.</td></tr>
<tr><td>V_{CC}</td><td>24</td><td></td><td>V_{CC}: The +5V power supply pin. A 0.1µF capacitor between pins VCC and GND is recommended for decoupling.</td></tr>
</table>

Functional Description

General

The 82C54 is a programmable interval timer/counter designed for use with 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 82C54 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 82C54 to match his requirements and programs one of the counters for the desired delay. After the desired delay, the 82C54 will interrupt the CPU. Software overhead is minimal and variable length delays can easily be accommodated.

Some of the other computer/timer functions common to microcomputers which can be implemented with the 82C54 are:

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

Data Bus Buffer

This three-state, bi-directional, 8-bit buffer is used to interface the 82C54 to the system bus (see Figure 1).

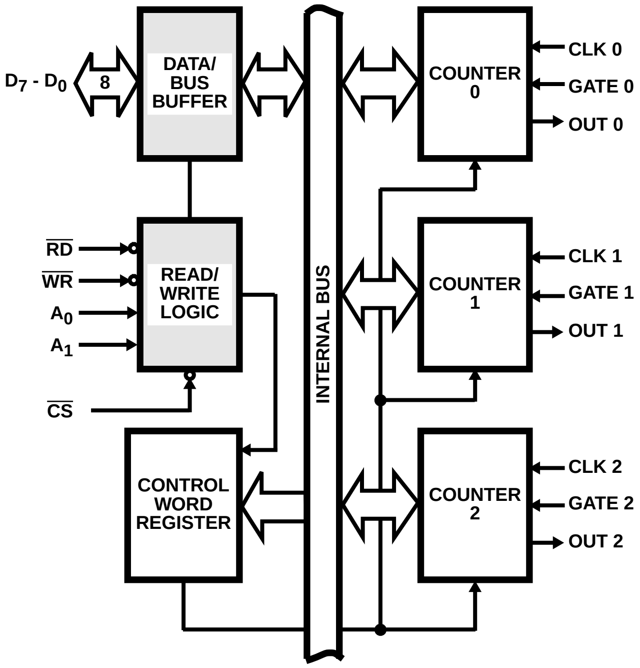


FIGURE 1. DATA BUS BUFFER AND READ/WRITE LOGIC FUNCTIONS

Read/Write Logic

The Read/Write Logic accepts inputs from the system bus and generates control signals for the other functional blocks of the 82C54. A1 and A0 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 82C54 that the CPU is reading one of the counters. A "low" on the $\overline{WR}$ input tells the 82C54 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 82C54 has been selected by holding $\overline{CS}$ low.

Control Word Register

The Control Word Register (Figure 2) is selected by the Read/Write Logic when A1, A0 = 11. If the CPU then does a write operation to the 82C54, the data is stored in the Control Word Register and is interpreted as a Control Word used to define the Counter operation.

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

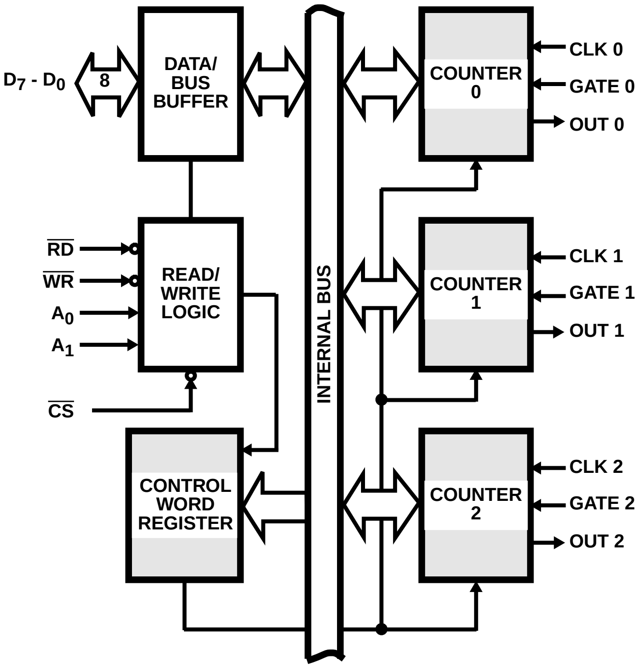


FIGURE 2. CONTROL WORD REGISTER AND COUNTER FUNCTIONS

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 signal counter is shown in Figure 3. 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 the figure, 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 labeled CE (for Counting Element). It is a 16-bit presettable synchronous down counter.

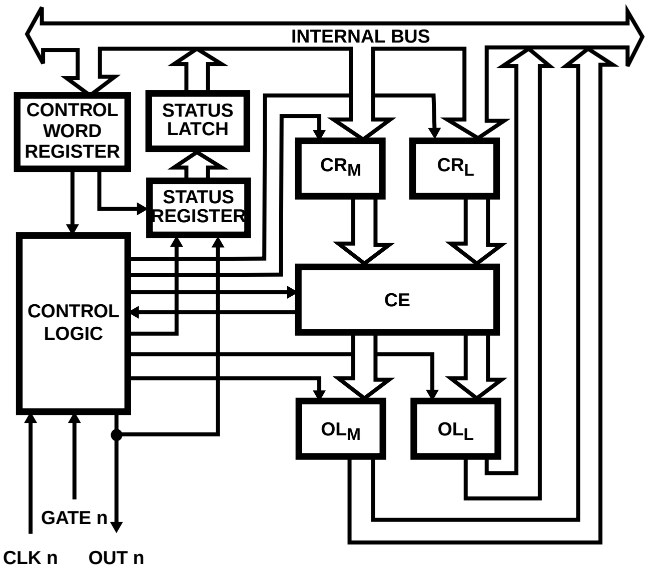


FIGURE 3. COUNTER INTERNAL BLOCK DIAGRAM

OLM and OLL are two 8-bit latches. OL stands for “Output Latch”; the subscripts M and L for “Most significant byte” and “Least significant byte”, 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 82C54, 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 CRM and CRL (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. CRM and CRL are cleared when the Counter is 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.

82C54 System Interface

The 82C54 is treated by the system 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 A0, A1 connect to the A0, A1 address bus signals of the CPU. The $\overline{CS}$ can be derived directly from the address bus using a linear select method or it can be connected to the output of a decoder.

Operational Description

General

After power-up, the state of the 82C54 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 82C54

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

All Control Words are written into the Control Word Register, which is selected when A1, A0 = 11. The Control Word specifies which Counter is being programmed.

By contrast, initial counts are written into the Counters, not the Control Word Register. The A1, A0 inputs are used to select the Counter to be written into. The format of the initial count is determined by the Control Word used.

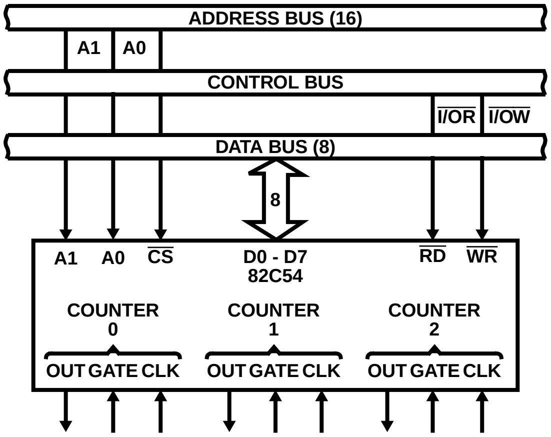


FIGURE 4. 82C54 SYSTEM INTERFACE

Write Operations

The programming procedure for the 82C54 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 A1, A0 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 above is acceptable.

Control Word Format

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

D7	D6	D5	D4	D3	D2	D1	D0
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)

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.

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

BCD - Binary Coded Decimal

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

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

Possible Programming Sequence

	A1	A0
Control Word - Counter 0	1	1
LSB of Count - Counter 0	0	0
MSB of Count - Counter 0	0	0
Control Word - Counter 1	1	1
LSB of Count - Counter 1	0	1
MSB of Count - Counter 1	0	1
Control Word - Counter 2	1	1
LSB of Count - Counter 2	1	0
MSB of Count - Counter 2	1	0

Possible Programming Sequence

	A1	A0
Control Word - Counter 0	1	1
Control Word - Counter 1	1	1
Control Word - Counter 2	1	1
LSB of Count - Counter 2	1	0

Possible Programming Sequence (Continued)

	A1	A0
LSB of Count - Counter 1	0	1
LSB of Count - Counter 0	0	0
MSB of Count - Counter 0	0	0
MSB of Count - Counter 1	0	1
MSB of Count - Counter 2	1	0

Possible Programming Sequence

	A1	A0
Control Word - Counter 2	1	1
Control Word - Counter 1	1	1
Control Word - Counter 0	1	1
LSB of Count - Counter 2	1	0
MSB of Count - Counter 2	1	0
LSB of Count - Counter 1	0	1
MSB of Count - Counter 1	0	1
LSB of Count - Counter 0	0	0
MSB of Count - Counter 0	0	0

Possible Programming Sequence

	A1	A0
Control Word - Counter 1	1	1
Control Word - Counter 0	1	1
LSB of Count - Counter 1	0	1
Control Word - Counter 2	1	1
LSB of Count - Counter 0	0	0
MSB of Count - Counter 1	0	1
LSB of Count - Counter 2	1	0
MSB of Count - Counter 0	0	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 programming sequences.

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.

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 82C54.

There are three possible methods for reading the Counters. The first is through the Read-Back command, which is explained later. The second is a simple read operation of the Counter, which is selected with the A1, A0 inputs. The only requirement is that 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 process of changing when it is read, giving an undefined result.

Counter Latch Command

The other method for reading the Counters involves a special software command called the "Counter Latch Command". Like a Control Word, this command is written to the Control Word Register, which is selected when A1, A0 = 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.

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

D7	D6	D5	D4	D3	D2	D1	D0
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 products.

The selected Counter's output latch (OL) latches the count when 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 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 82C54 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 or 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 read-back command allows the user to check the count value, programmed Mode, and current state 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 5. The command applies to the counters selected by setting their corresponding bits D3, D2, D1 = 1.

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

D7	D6	D5	D4	D3	D2	D1	D0
1	1	$\overline{\text{COUNT}}$	$\overline{\text{STATUS}}$	CNT 2	CNT 1	CNT 0	0

D5: 0 = Latch count of selected Counter (s)
D4: 0 = Latch status of selected Counter(s)
D3: 1 = Select Counter 2
D2: 1 = Select Counter 1
D1: 1 = Select Counter 0
D0: Reserved for future expansion; Must be 0

FIGURE 5. READ-BACK COMMAND FORMAT

The read-back command may be used to latch multiple counter output latches (OL) by setting the COUNT bit D5 = 0 and selecting the desired counter(s). This signal 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). That 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 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 6. 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.

D7	D6	D5	D4	D3	D2	D1	D0
OUTPUT	NULL COUNT	RW1	RW0	M2	M1	M0	BCD

D7: 1 = Out pin is 1
0 = Out pin is 0
D6: 1 = Null count
0 = Count available for reading
D5 - D0 = Counter programmed mode (See Control Word Formats)

FIGURE 6. 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 counter 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 below.

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 - CE)	Null Count = 0

(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.

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.

COMMANDS								DESCRIPTION	RESULT
D7	D6	D5	D4	D3	D2	D1	D0		
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 7. READ-BACK COMMAND EXAMPLE

Both count and status of the selected counter(s) may be latched simultaneously by setting both COUNT and 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 7.

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.

$\overline{\text{CS}}$	$\overline{\text{RD}}$	$\overline{\text{WR}}$	A1	A0	
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 (Three-State)
1	X	X	X	X	No-Operation (Three-State)
0	1	1	X	X	No-Operation (Three-State)

FIGURE 8. READ/WRITE OPERATIONS SUMMARY

Mode Definitions

The following are defined for use in describing the operation of the 82C54.

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 (See "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 to 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.

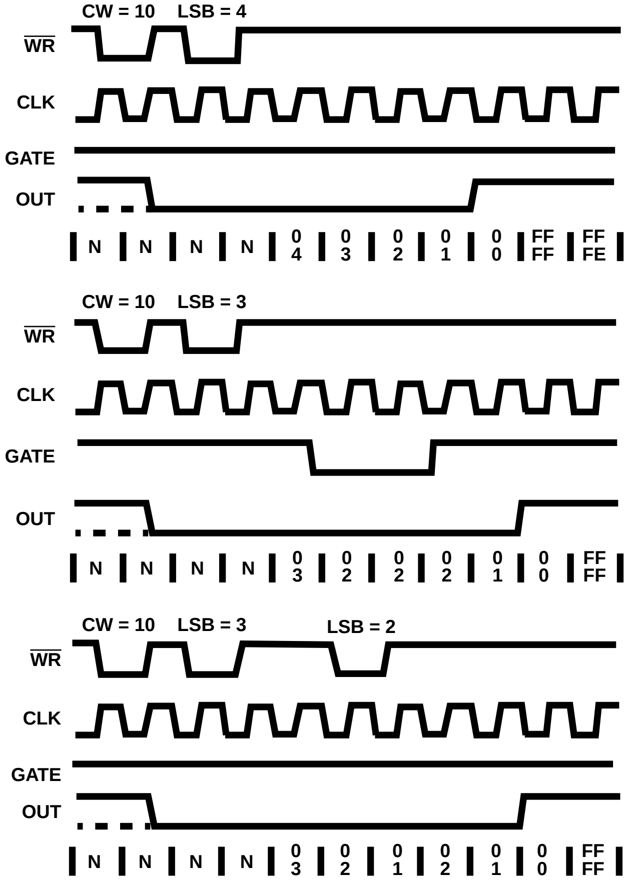

CW = 10 LSB = 4
WR
CLK
GATE
OUT
N N N N 0 4 0 3 0 2 0 1 0 0 FF FF FF FE
CW = 10 LSB = 3
WR
CLK
GATE
OUT
N N N N 0 3 0 2 0 2 0 2 0 1 0 0 FF FF
CW = 10 LSB = 3
LSB = 2
WR
CLK
GATE
OUT
N N N N 0 3 0 2 0 1 0 2 0 1 0 0 FF FF

FIGURE 9. MODE 0

NOTES: 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 ($\overline{\text{CS}}$ 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.
6. N stands for an undefined count.
7. Vertical lines show transitions between count values.

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 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 retriggerable. In that case, the Counter is loaded with the new count and the one-shot pulse continues until the new count expires.

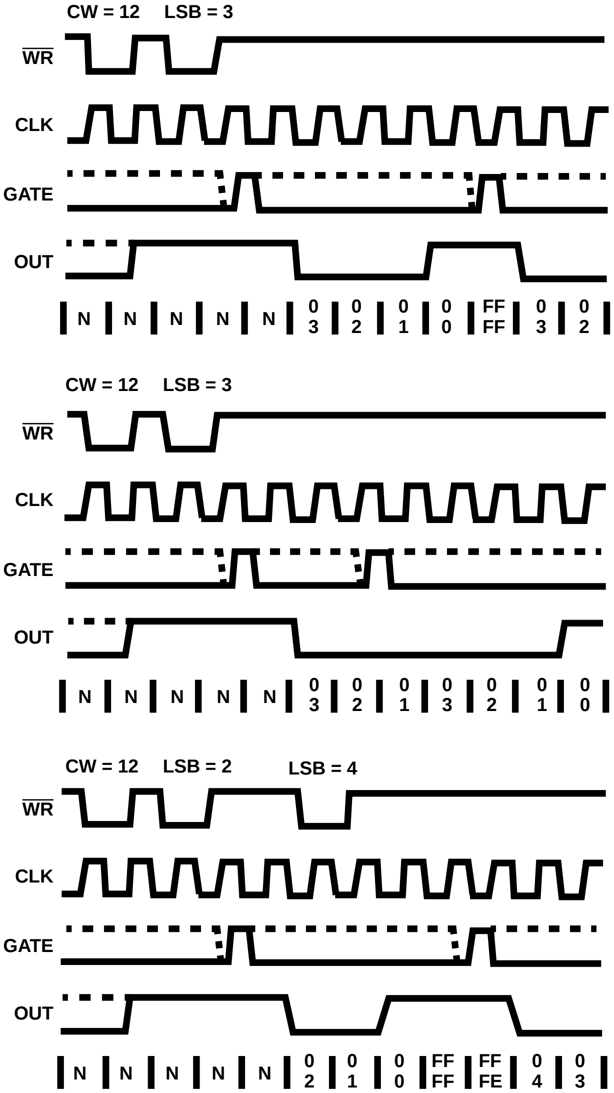


FIGURE 10. MODE 1

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 end of the current counting cycle.

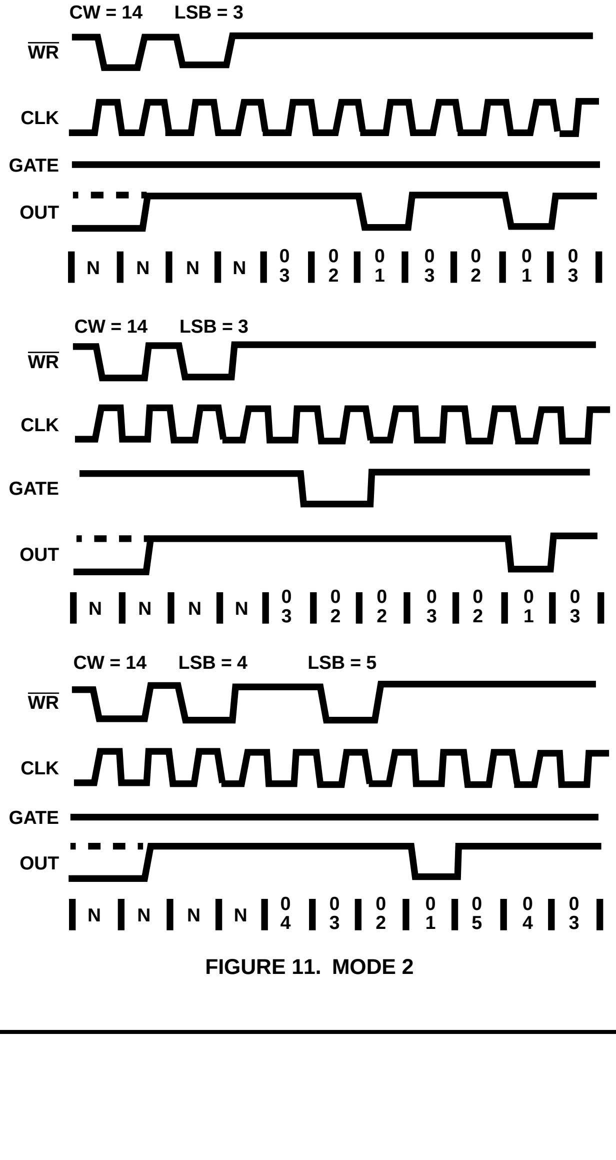


FIGURE 11. MODE 2

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 initial count has expired, OUT goes low for the remainder of the count. Mode 3 is periodic; the sequence above is repeated indefinitely. An initial count of N results in a square wave with a period of N CLK cycles.

GATE = 1 enables counting; GATE = 0 disables counting. If GATE goes low while OUT is low, OUT is set high immediately; no CLK pulse is required. A trigger reloads the Counter with the initial count on the next CLK pulse. 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. 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 half-cycle of the square wave, 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 half-cycle.

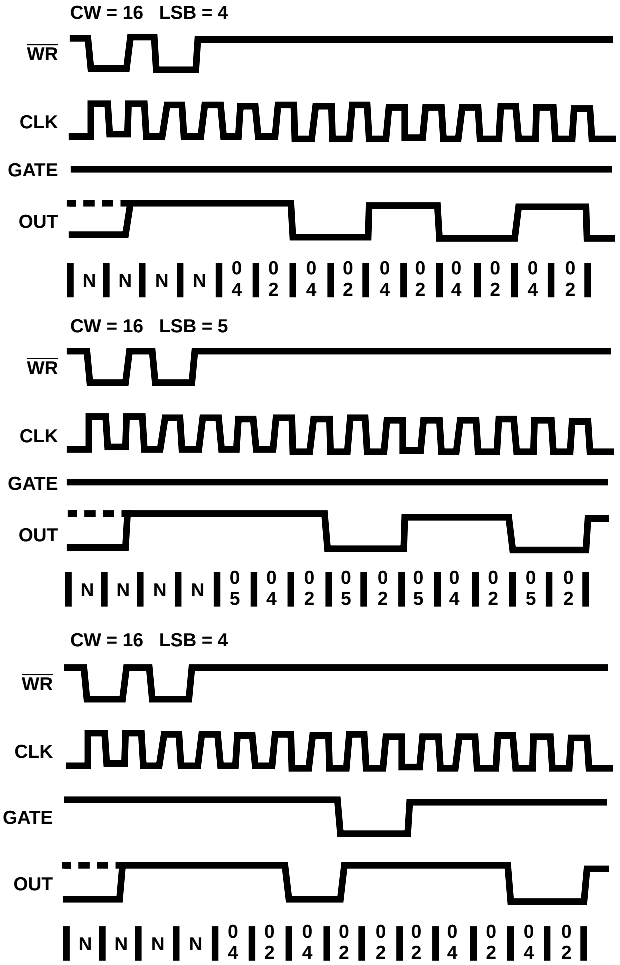

CW = 16 LSB = 4
WR
CLK
GATE
OUT
N N N N 0 4 0 2 0 4 0 2 0 4 0 2 0 4 0 2 0 4 0 2
CW = 16 LSB = 5
WR
CLK
GATE
OUT
N N N N 0 5 0 4 0 2 0 5 0 2 0 5 0 4 0 2 0 5 0 2
CW = 16 LSB = 4
WR
CLK
GATE
OUT
N N N N 0 4 0 2 0 4 0 2 0 2 0 2 0 4 0 2 0 4 0 2

FIGURE 12. MODE 3

Mode 3 is Implemented as Follows:

EVEN COUNTS: OUT is initially high. The initial count is loaded on one CLK pulse and then is decremented by two on succeeding CLK pulses. When the count expires, OUT changes value and the Counter is reloaded with the initial count. The above process is repeated indefinitely.

ODD COUNTS: OUT is initially high. The initial count is loaded on one CLK pulse, decremented by one on the next CLK pulse, and then decremented by two on succeeding CLK pulses. When the count expires, OUT goes low and the Counter is reloaded with the initial count. The count is decremented by three on the next CLK pulse, and then by two on succeeding CLK pulses. When the count expires, OUT goes high again and the Counter is reloaded with the initial count. The above process is repeated indefinitely. So for odd counts, OUT will be high for (N + 1)/2 counts and low for (N - 1)/2 counts.

Mode 4: Software Triggered Mode

OUT will be initially high. When the initial count expires, OUT will go low for one CLK pulse then go high again. The counting sequence is “Triggered” by writing the initial count.

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

After writing a Control Word and initial count, the Counter 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 strobe low until N + 1 CLK pulses after the initial count is written.

If a new count is written during counting, 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 has no effect on counting.

(2) Writing the second byte allows the new count to be loaded on the next CLK pulse.

This allows the sequence to be “retriggered” by software. OUT strobes low N + 1 CLK pulses after the new count of N is written.

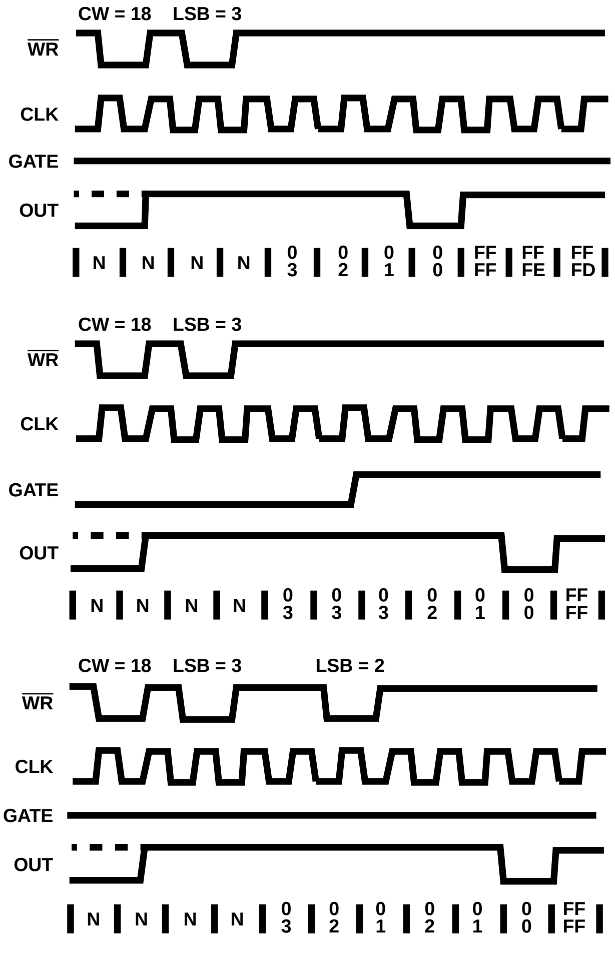

CW = 18 LSB = 3
WR
CLK
GATE
OUT
N N N N 0 3 0 2 0 1 0 0 FF FF FF FE FF FD
CW = 18 LSB = 3
WR
CLK
GATE
OUT
N N N N 0 3 0 3 0 3 0 2 0 1 0 0 FF FF
CW = 18 LSB = 3 LSB = 2
WR
CLK
GATE
OUT
N N N N 0 3 0 2 0 1 0 2 0 1 0 0 FF FF

FIGURE 13. MODE 4

Mode 5: Hardware Triggered Strobe (Retriggerable)

OUT will initially be high. Counting is triggered by a rising edge of GATE. When the initial count has expired, OUT will go low for one CLK pulse and then go high again.

After writing the Control Word and initial count, the counter will not be loaded until the CLK pulse after a trigger. This CLK pulse does not decrement the count, so for an initial count of N, OUT does not strobe low until N + 1 CLK pulses after trigger.

A trigger results in the Counter being loaded with the initial count on the next CLK pulse. The counting sequence is triggerable. OUT will not strobe low for N + 1 CLK pulses after any trigger GATE has no effect on OUT.

If a new count is written during counting, the current counting sequence will not be affected. If a trigger occurs after the new count is written but before the current count expires, the Counter will be loaded with new count on the next CLK pulse and counting will continue from there.

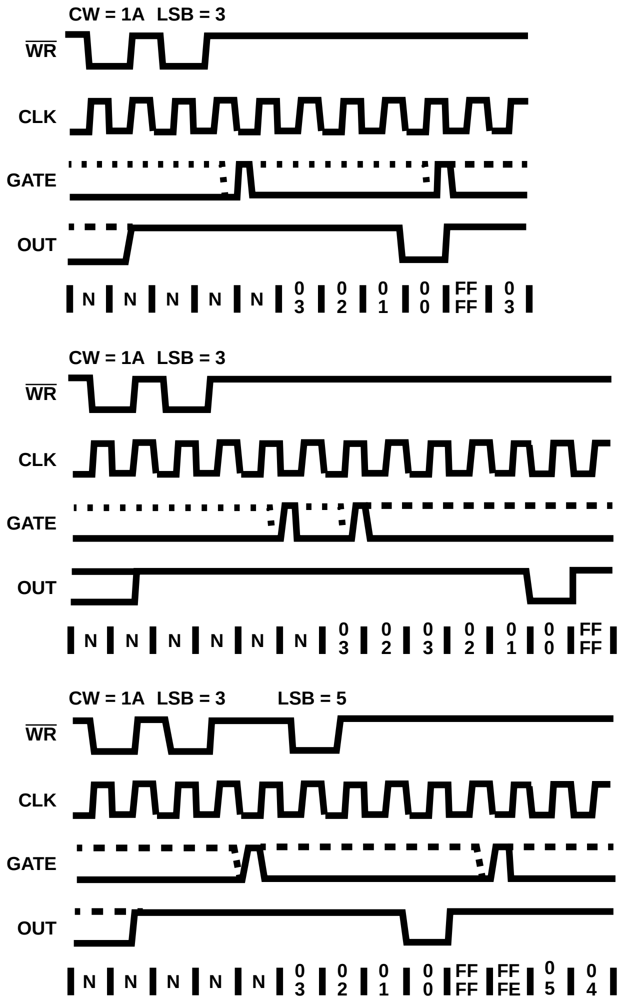

CW = 1A LSB = 3
WR
CLK
GATE
OUT
N N N N N 0 3 0 2 0 1 0 0 FF FF 0 3
CW = 1A LSB = 3
WR
CLK
GATE
OUT
N N N N N N 0 3 0 2 0 3 0 2 0 1 0 0 FF FF
CW = 1A LSB = 3 LSB = 5
WR
CLK
GATE
OUT
N N N N N 0 3 0 2 0 1 0 0 FF FF FF FE 0 5 0 4

FIGURE 14. MODE 5

Operation Common to All Modes

Programming

When a Control Word is written to a Counter, all Control Logic, is immediately reset and OUT goes to a known initial state; no CLK pulses are required for this.

Gate

The GATE input is always sampled on the rising edge of CLK. In Modes 0, 2, 3 and 4 the GATE input is level sensitive, and logic level is sampled on the rising edge of CLK. In modes 1, 2, 3 and 5 the GATE input is rising-edge sensitive. In these Modes, a rising edge of Gate (trigger) sets an edge-sensitive flip-flop in the Counter. This flip-flop is then sampled on the next rising edge of CLK. The flip-flop is reset immediately after it is sampled. In this way, a trigger will be detected no matter when it occurs - a high logic level does not have to be maintained until the next rising edge of CLK. Note that in Modes 2 and 3, the GATE input is both edge- and level-sensitive.

Counter

New counts are loaded and Counters are decremented on the falling edge of CLK.

The largest possible initial count is 0; this is equivalent to 2^{16} for binary counting and 10^{4} for BCD counting.

The counter does not stop when it reaches zero. In Modes 0, 1, 4, and 5 the Counter "wraps around" to the highest count, either FFFF hex for binary counting or 9999 for BCD counting, and continues counting. Modes 2 and 3 are periodic; the Counter reloads itself with the initial count and continues counting from there.

SIGNAL STATUS MODES	LOW OR GOING LOW	RISING	HIGH
0	Disables Counting	-	Enables Counting
1	-	1) Initiates Counting 2) Resets output after next clock	-
2	1) Disables counting 2) Sets output immediately high	Initiates Counting	Enables Counting
3	1) Disables counting 2) Sets output immediately high	Initiates Counting	Enables Counting
4	1) Disables Counting	-	Enables Counting
5	-	Initiates Counting	-

FIGURE 15. GATE PIN OPERATIONS SUMMARY

MODE	MIN COUNT	MAX COUNT
0	1	0
1	1	0
2	2	0
3	2	0
4	1	0
5	1	0

NOTE: 0 is equivalent to 2^{16} for binary counting and 10^{4} for BCD counting.

FIGURE 16. MINIMUM AND MAXIMUM INITIAL COUNTS