PanaXSeries

The One to Watch for Constant Innovation-Making the Future Come Alive

MICROCOMPUTER MN101C

MN101C46F/F46F

LSI User's Manual

Pub.No.21446-0211E

Panasonic

“PanaXSeries” is a trademark of Matsushita Electric Inustrial Co., Ltd.
The other corporation names, logotype and product names written in this book are trademarks or registered trademarks of theri corresponding corporations.

Request for your special attention and precautions in using the technical informaition and semiconductors described in this book

(1) An export permit needs to be obtained from the competent authorities of the Japanese Government if any of the products or technologies described in this book and controlled under the "Foreign Exchange and Foreign Trade Law" is to be exported or taken out of Japan.

(2) The contents of this book are subject to change without notice in matters of improved function. When finalizing your design, therefore, ask for the most up-to-date version in advance in order to check for any changes.

(3) We are not liable for any damage arising out of the use of the contents of this book, or for any infringement of patents or any other rights owned by a third party.

(4) No part of this book may be reprinted or reproduced by any means without written permission from our company.

(5) This book deals with standard specification. Ask for the latest individual Product Standards or Specifications in advance for more detailsd infomation required for your design, purchasing and applications.

If you have any inquiries or questions about this book or our semiconductors, please contact one of our sales offices listed at the back of this book.

Contents

List of Tables

List of Figures

About This Manual

This manual is intended for assembly-language programming engineers. It describes the internal configuration and hardware functions of the MN101C46F and MN101CF46F microcontrollers. Except where specified, when this manual refers to MN101C46F, it implies both products.

Using This Manual

The chapters in this manual deal with the internal blocks of the MN101C46F. Chapters 1 to 5 provide an overview of the MN101C46F's general specifications, basic CPU functions, interrupts, I/O ports, and prescalar functions. Chapters 6 to 10 describe the 8-bit timers, on-screen display, analog-to-digital converter, watchdog timer, and closed-caption decoder. Chapter 11 describes the IR remote signal receiver. Chapter 12 describes the ROM correction feature. Chapter 13 describes the I^2C bus controller. Chapter 14 describes the pulse width modulator. The appendices provide register and instruction maps, instruction sets, and describe the flash EEPROM version.

Text Conventions

Where applicable, this manual provides special notes and warnings. Helpful or supplementary comments appear in the sidebar. In addition, the following symbols indicate key information and warnings:

Key information
These notes summarize key points relating to an operation.

Warning
Please read and follow these instructions to prevent damage or reduced performance.

Register Conventions

This manual presents 8-bit registers in the following format:

REGISTER: 8-Bit Register Name **x'00000'**

Bit:	7	6	5	4	3	2	1	0
	—	—	—	Bit Name	Bit Name	Bit Name	Bit Name	Bit Name
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R/W	R/W	R/W	R/W

The hexadecimal value (x'00000') indicates the register address. The top row of the register diagram holds the bit numbers. Bit 7 is the most significant bit (MSB). The second row holds the bit or field names. A dash (—) indicates a reserved bit. The third row shows the reset values, and the fourth row shows the accessibility. (R = read only, W = write only, and R/W = readable/writable.)

Related Documents

- *MN101C Series LSI User Manual*
  Describes the device architecture.

- *MN101C Series Instruction Manual*
  Describes the instruction set.

- *MN101C Series Cross-Assembler User Manual*
  Describes the assembler syntax and notation.

- *MN101C Series C Compiler User Manual: Usage Guide*
  Describes the installation, commands, and options for the C compiler.

- *MN101C Series C Compiler User Manual: Language Description*
  Describes the syntax for the C compiler.

- *MN101C Series C Compiler User Manual: Library Reference*
  Describes the standard libraries for the C compiler.

- *MN101C Series C Source Code Debugger User Manual*
  Describes the use of the C source code debugger.

- *MN101C Series PanaX Series Installation Manual*
  Describes the installation of the C compiler, cross-assembler, and C source code debugger and the procedures for using the in-circuit emulator.

Questions and Comments

We welcome your questions, comments, and suggestions. Please contact the semiconductor design center closest to you. See the last page of this manual for a list of addresses and telephone numbers. You can also find contact and product information on the World Wide Web at: **http://www.psdc.com/**

1 General Description

1.1 Overview

The MN101C series of 8-bit single-chip microcontrollers can be used in embedded applications that incorporate a wide array of peripheral features, such as cameras, VCRs, minidisc players, TVs, CD players, laser disc players, printers, telephones, home automation devices, pagers, air-conditioners, palmtop computers, remote controllers, fax machines, and electronic musical instruments.

The MN101C46F has a flexible and optimized hardware architecture and a simple, efficient instruction set. It has 96 Kbytes of ROM and 3 Kbytes of RAM on chip. It provides six external interrupts, ten internal interrupts (including the NMI), three timer/counters, an analog-to-digital converter, a watchdog timer, an on-screen display function, a closed-caption decoder, an I^2C interface, pulse width modulators, and an IR remote signal receiver. This structure makes it optimum for controlling a television tuner.

The machine cycle (minimum instruction execution time) in standard mode is 279.33 ns when the raw oscillation f_{OSC} is 14.32 MHz. The ICs use 42-pin SDIP packages.

1.2 Series Products

This manual describes the following MN101C46F derivatives. These two chips have the same functions.

Table 1-1 MN101C46F Derivatives

Product	ROM size	RAM size	Chip type
MN101C46F	96 Kbytes	3 Kbytes	Mask ROM
MN101CF46F	96 Kbytes	3 Kbytes	Flash ROM

1.3 0Hardware Functions

Table 1-2 Description of Hardware

Function	Description
CPU core	MN101C Core Load-store architecture (three-stage pipeline) Half-byte instruction set/handy addressing 256 Kbytes of memory space (shared by instructions and data) Machine cycle • NORMAL mode: 279.3 to 333.3 ns/3.0 to 3.58 MHz (3.0 to 3.6 V) • SLOW mode: 1.12 to 1.33 µs/0.75 to 0.895 MHz (3.0 to 3.6 V) Operating modes: NORMAL, SLOW, HALT, and STOP
ROM correction	Programming can be changed in up to 16 locations.
Internal memory	96-Kbyte ROM, 3-Kbyte RAM
Interrupt functions	Twelve internal interrupts Software fault interrupt (nonmaskable interrupt (NMI)) Timer interrupts • Timers 2 interrupt • Timer 3 interrupt • Timer 4 interrupt ADC interrupt I^2C interrupt OSD interrupt Closed-caption decoder interrupts • Closed-caption 0 interrupt • Closed-caption 1 interrupt • Closed-caption 0 VSYNC interrupt • Closed-caption 1 VSYNC interrupt IR remote receiver interrupt Six external interrupts IRQ0-5: Edge selectable
Timer/counters	Three 8-bit timers Timer 2 • Clock sources: f_{OSC}, $f_{OSC}/4$, $f_{OSC}/16$, $f_{OSC}/32$, $f_{OSC}/64$, $f_S/2$, $f_S/4$, f_X Timer 3 • 16-bit cascading function (connecting to timer 2) • Clock sources: f_{OSC}, $f_{OSC}/4$, $f_{OSC}/16$, $f_{OSC}/64$, $f_{OSC}/128$, $f_S/2$, $f_S/8$, f_X Timer 4 • Clock sources: f_{OSC}, $f_{OSC}/4$, $f_{OSC}/16$, $f_{OSC}/32$, $f_{OSC}/64$, $f_S/2$, $f_S/4$, f_X
Watchdog timer	• Timeout period selectable to $f_S/2^{16}$, $f_S/2^{18}$, or $f_S/2^{20}$ • Forced hardware reset within the IC when timeout period detected
A/D converter	5 bits and 8 channels
Remote signal reception	Household Electrical Appliance Manufacturers Association, auto 5-/6- bit detection, 1-bit interrupts
I^2C	Two multimaster circuits (1 internal)
PWM	Six 8-bit channels, one 14-bit channel

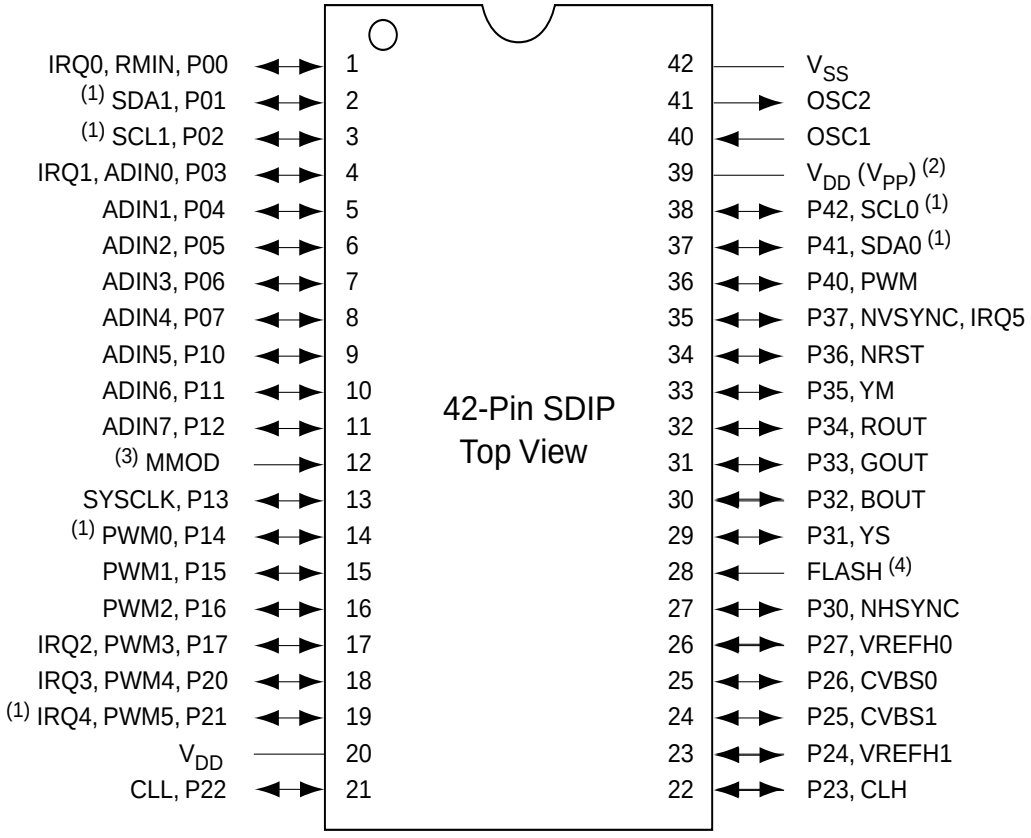
Table 1-2 Description of Hardware (Continued)

Function	Description
SYSCLK output	Clock output uses a period of f_S or $f_S/4096$
Closed caption decoders	• Two channels with a built-in sync separator. • Handles 12.0, 14.0, or 14.32 MHz
OSD functions	• 16- × 18- pixel (H × V) graphic tiles • 16 character tile sizes • 8 out of 27 colors displayable per tile on the graphics layer • 27 text colors and 27 text background colors displayable
Port functions	35 I/O ports • Six 5-V, N-channel, open-drain ports Four input ports
Low-power modes	STOP, HALT, and SLOW modes
Operating voltages	3.0 to 3.6 V
Package	42-pin SDIP 64-pin LQFP (1)
Package model number	SDIP042-P-0600 LQFP064-P-1414 (1)

Notes: 1. In case of flst package.

1.4 Pin Description

1.4.1 Pin Configuration



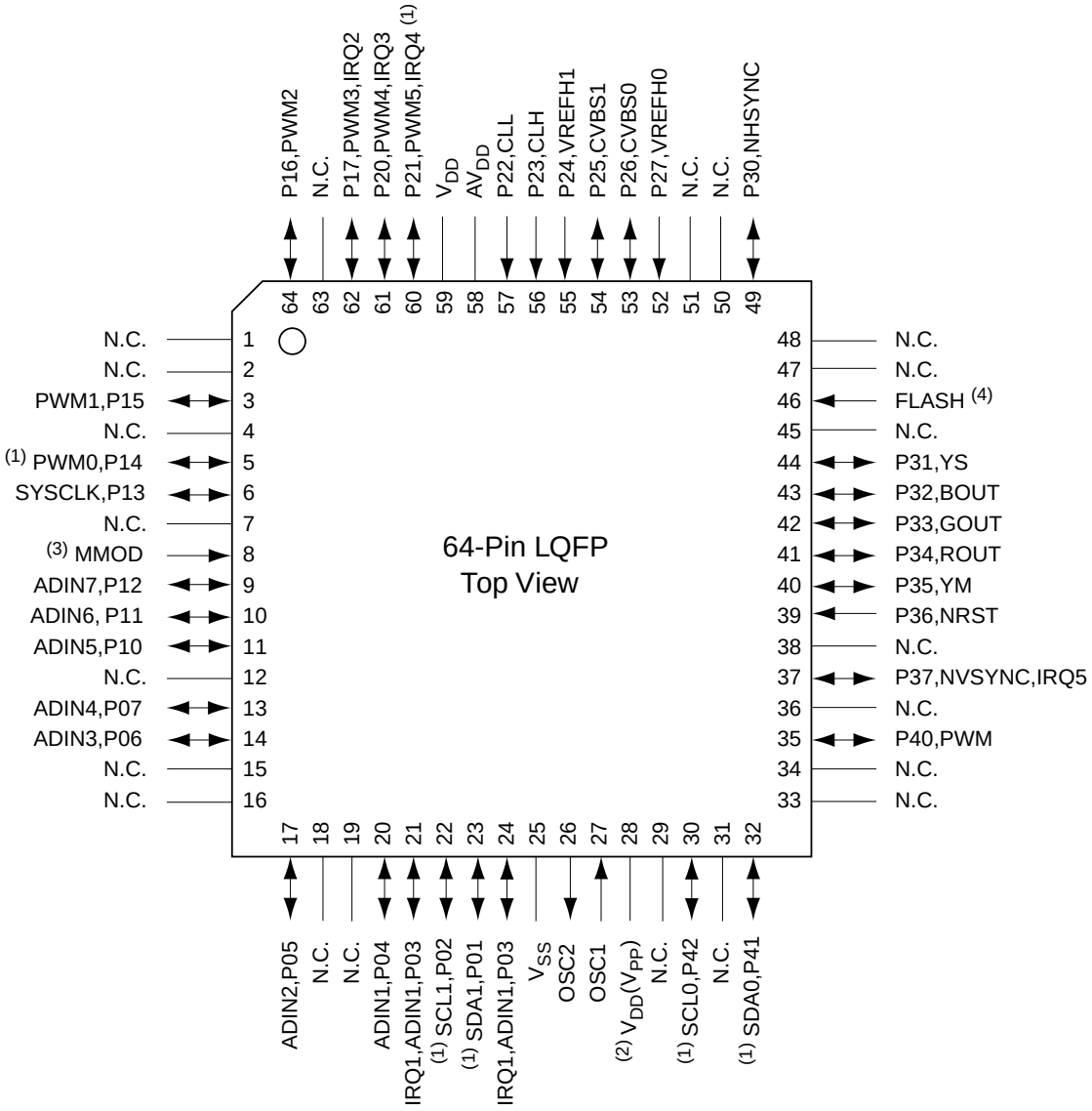
Notes: 1. P01, P02, P14, P21, P41, and P42 are 5-V, N-channel, open-drain pins.

2. V_{PP} port in flash ROM mode; V_{DD} port in mask ROM mode.

3. MMOD: tied high (sets test mode pin to normal mode)

4. FLASH: tied low (sets flash mode pin to normal mode)

Figure 1-1 MN101C46F Pin Configuration



Notes:
1. P01, P02, P14, P21, P41, and P42 are 5-V, N-channel, open-drain pins.
2. V_{PP} port in flash ROM mode; V_{DD} port in mask ROM mode.
3. MMOD: tied high (sets test mode pin to normal mode)
4. FLASH: tied low (sets flash mode pin to normal mode)

Figure 1-2 MN101C46F Pin Configuration

1.4.2 Pin Description

Table 1-3 Pin Description

Block	Pin Name	I/O	Pin Count	Description
Power	V_{DD}	I	1	Voltage supply (Apply 3.0 to 3.6 V.)
	V_{SS}	I	1	Ground reference (Connect directly to external ground.)
	V_{DD}/V_{PP}	I	1	Voltage supply: V_{DD} in mask ROM version and V_{PP} in EEPROM version
Clocks	SYSCLK	O	1	System clock output
	OSC1	I	1	Oscillator input connection
	OSC2	O	1	Oscillator output connection
Reset	NRST	I/O	1	Reset (alt. function: P36)
Interrupts (external)	IRQ0−IRQ5	I	6	External interrupt request to microcontroller (alt. functions: P00, P03, P17, P20, P21, P37)
OSD	NHSYNC	I	1	Horizontal sync signal input
	NVSYNC	I	1	Vertical sync signal input
	YS,YM	O	2	Video signal control
	ROUT, GOUT, BOUT	O	3	RGB screen output
I^2C interfaces (2)	SDA0/SDA1	I/O	2	I^2C data
	SCL0/SCL1	I/O	2	I^2C clock
IR remote signal receiver	RMIN	I	1	Remote signal input
PWM (8-bit, 6-channel)	PWM0−PWM5	O	6	8-bit pulse width modulator output
PWM (14-bit, 1-channel)	PWM	O	1	14-bit pulse width modulator output
I/O ports	P00−P07	I/O	8	General-purpose port 0 I/O
	P10−P17	I/O	8	General-purpose port 1 I/O
	P20−P27	I/O	8	General-purpose port 2 I/O
	P30−P37	I/O	8	General-purpose port 3 I/O
	P40−P42	I/O	3	General-purpose port 4 I/O
Closed-caption decoders (2)	CVBS0/CBVS1	I	2	Composite video signal input
	CLH	I	1	Clamp level high input
	CLL	I	1	Clamp level low input
	VREFHS	I	1	CCD reference voltage input
	VREFLS	I	1	CCD reference voltage input
ADC (5-bit, 8-channel)	ADIN0−ADIN7	I	8	Analog signal input
Flash	FLASH	I	1	Dedicated flash mode input (Connect to V_{SS}.)
Test	MMOD	I	1	Test pin (Connect to V_{DD}.)

1.5 Electrical Characteristics

Type: CMOS integrated circuit

Function: 16-bit microcontroller with graphic display circuit

Application: Television

Connection: See Figure 1-9, "Correct Connection Technique for the V_{DD} and V_{SS} Pins," on page 14.

Packaging: See Figure 1-1, "MN101C46F Pin Configuration," on page 4.

Table 1-4 Absolute Maximum Ratings (1)

V_{SS} = 0 V

No.	Parameter	Symbol	Rating	Unit
A1	Power supply voltage	V_{DD}	-0.3 to +4.6	V
A2	Input pin voltage (normal pins)	V_I	-0.3 to V_{DD} + 0.3	
	(5-V pins)	V_{I5}	-0.3 to +6.0	
A3	Output pin voltage (normal pins)	V_O	-0.3 to V_{DD} + 0.3	
	N-ch., open-drain pins when V_{DD} > 1.4 V	V_{O51}	-0.3 to +6.0	
	N-ch., open-drain pins when $V_{DD} \leq$ 1.4 V	V_{O52}	-0.3 to +4.6	
A4	Power dissipation	P_D	1000	mW
A5	Operating ambient temperature	T_{opr}	-20 to +70	°C
A6	Storage temperature MN101C46F	T	-55 to +125	
	MN101CF46F		-55 to +95	

Note: 1. Stresses beyond those listed under *Absolute Maximum Ratings* may cause permanent damage to the device. These are stress ratings only and functional operation of the device under these or any conditions other than those indicated in the operational sections of this specification is not implied.

■ Precautions

1. All of the V_{DD} and V_{SS} pins are external. Connect them directly to the power source and ground.

2. Thoroughly test your crystal oscillator in this device's oscillator cell before using.

3. If you install the product close to high-field emissions (under a cathode ray tube, for example), shield the package surface to ensure normal performance.

4. To improve noise and latch-up tolerance, connect bypass capacitors between V_{DD} and V_{SS} pins with a thick line by the shortest possible distance. Use electrolytic capacitors of at least 22 µF (6.3-V tolerance).

5. To ensure normal performance, you must connect V_{DD} power supply pins 20 and 39 to external sources of the same voltage (3.3 V). This ensures that no other element can supply power externally to either of the pins.

6. P01, P02, P14, P21, P41, and P42 are N-channel, open-drain pins.

Table 1-5 Recommended Operating Conditions V_{SS} = 0 V

No.	Parameter	Symbol	Conditions	Min	Typ	Max	Unit
B1	Power supply voltage 1	V_{DD}	f_{OSC} = 12 to 14.32 MHz	3.0	3.3	3.6	V
B2	Ambient temperature	T_a	V_{DD} = 3.0 V to 3.6 V	-20		70	°C
B3	Instruction execution speed	t_c	V_{DD} = 3.0 V to 3.6 V	279.3		333.3	ns
B4	Oscillator frequency (See figure 1-3.)	f_{OSC1}	V_{DD} = 3.0 V to 3.6 V	12.00		14.32	MHz

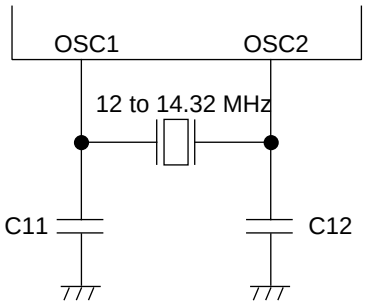


Figure 1-3 OSC1 and OSC2 Oscillator Circuits

Table 1-6 Electrical Characteristics T_a = -20 to +70°C, V_{DD} = 3.3 V, V_{SS} = 0 V

No.	Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Power supply current							
C1	Operating supply current	I_{DD0}	f_{OSC} = 14.32 MHz Input pins tied to V_{DD} or V_{SS} Output pins open OSD operating	10		50	mA
OSC2: Oscillator circuit							
C2	Internal feedback resistance	R_{fbOSC2}	OSC1 = V_{DD} or OSC1 =V_{SS}	313	940	2820	kΩ

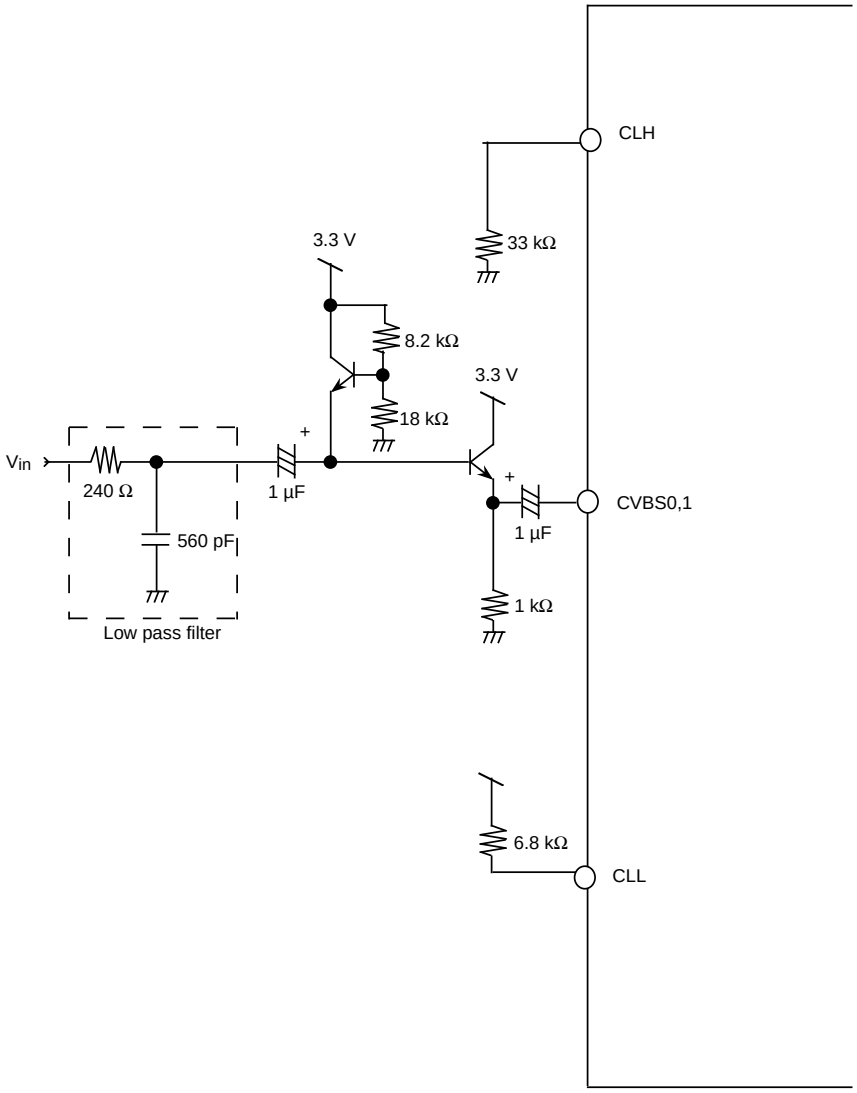
Table 1-6 Electrical Characteristics (Continued) T_a = -20 to +70°C, V_{DD} = 3.3 V, V_{SS} = 0 V

No.	Parameter	Symbol	Conditions	Min	Typ	Max	Unit
MMOD and FLASH: Input pin with CMOS input level							
C3	Input high voltage	V_{IH}	V_{DD} = 3.0 V to 3.6 V	0.7V_{DD}		V_{DD}	V
C4	Input low voltage	V_{IL}	V_{DD} = 3.0 V to 3.6 V	0		0.3V_{DD}	
C5	Input leakage current	I_{LI}	V_{IN} = 0.0 V to 3.6 V			±5	µA
P07, P10–P13, P15–P16, P22–P27, P31–P35, P40: I/O pins with CMOS input level							
C6	Input high voltage	V_{IH}	V_{DD} = 3.3 V to 3.6 V	0.7V_{DD}		V_{DD}	V
C7	Input low voltage	V_{IL}	V_{DD} = 3.3 V to 3.6 V	0		0.3V_{DD}	
C8	Output high voltage	V_{OH}	I_{OH} = -1 mA	V_{DD} - 0.6			
C9	Output low voltage	V_{OL}	I_{OL} = 1.8 mA			0.4	
C10	Output leakage current	I_{LO}	Output = Hi-Z V_{IN} = 0.0 V to 3.6 V			±5	µA
C11	Pullup resistance	R_{IH}	V_{IN} = 0.0 V	10	30	90	kΩ
P00, P03–P06, P17, P20, P30, P37: I/O pins with CMOS input level and Schmidt trigger							
C12	Input high voltage	V_{IH}	V_{DD} = 3.3 V to 3.6 V	0.7V_{DD}		V_{DD}	V
C13	Input low voltage	V_{IL}	V_{DD} = 3.3 V to 3.6 V	0		0.3V_{DD}	
C14	Output high voltage	V_{OH}	I_{OH} = -1 mA	V_{DD} - 0.6			
C15	Output low voltage	V_{OL}	I_{OL} = 1.8 mA			0.4	
C16	Output leakage current	I_{LO}	Output = Hi-Z V_{IN} = 0.0 V to 3.6 V			±5	µA
C17	Pullup resistance	R_{IH}	V_{IN} = 0.0 V	10	30	90	kΩ
P36 (NRST): I/O pin with CMOS input level, Schmidt trigger, and N-channel open-drain							
C18	Input high voltage	V_{IH}	V_{DD} = 3.3 V to 3.6 V	0.7V_{DD}		V_{DD}	V
C19	Input low voltage	V_{IL}	V_{DD} = 3.3 V to 3.6 V	0		0.3V_{DD}	
C21	Output leakage current	I_{LO}	Output = Hi-Z V_{IN} = 3.6 V			±5	µA
C22	Pullup resistance	R_{IH}	V_{IN} = 0.0 V	10	30	90	kΩ
P01–P02, P14, P21, P41-42: I/O pins with TTL input level, Schmidt trigger, and N-channel open-drain							
C23	Input high voltage	V_{IH}	V_{DD} = 3.3 V	2.2		V_{DD}	V
C24	Input low voltage	V_{IL}	V_{DD} = 3.3 V	0		0.6	
C25	Output low voltage	V_{OL}	I_{OL} = 4 mA			0.4	
C26	Output leakage current	I_{LO}	Output = Hi-Z V_{IN} = 0.0 V to 5.25 V			±10	µA
A/D Converter Characteristics							
C27	Resolution	RES				5	bits
C28	Conversion time	t_{AD}	f_{OSC} = 14.32 MHz	13.4			µs
C29	Analog input voltage	V_{IA}		V_{SS}		V_{DD}	V
C30	Conversion relativity accuracy	LE				±2	LSB

Table 1-6 Electrical Characteristics (Continued) T_a = -20 to +70°C, V_{DD} = 3.3 V, V_{SS} = 0 V

No.	Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Closed-Caption Decoder Characteristics (5, 6)							
(These pins also function as port pins. The characteristics below apply when the port function is disabled.)							
CVBS0, CVBS1							
C31	Low clamping current for input high	I_{CLH1}	V_{DD} = 3.3 V	5		25	µA
C32	Medium clamping current for input high	I_{CLH2}	V_{IN} = 1.6 V	50		180	
C33	Very high clamping current for input high	I_{CLH3}	With the following connections:	415		615	
C34	High clamping current for input high	I_{CLH4}	33 kΩ between CLH and GND	185		335	
C35	Low clamping current for input low	I_{CLL1}	6.8 kΩ between CLL and V_{DD}	-18		-3	
C36	Medium clamping current for input low	I_{CLL2}		-70		-20	
C37	Very high clamping current for input low	I_{CLL3}		-345		-145	
C38	High clamping current for input low	I_{CLL4}		-205		-55	

Note: 2. See figure 1-4 for the recommended pin connections.
3. See figure 1-5 for the input waveform specification for the composite video signal.

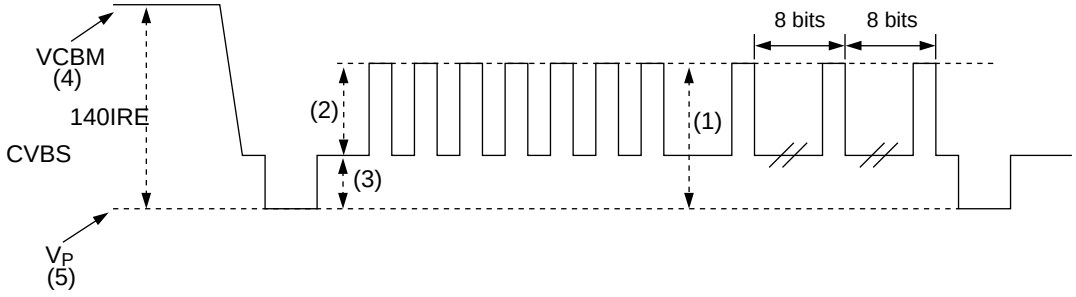


Note: The constants shown in this diagram are recommended values only. Operation at these values is not guaranteed.

Figure 1-4 External Connection Example for Closed-Caption Decoder Pins

Table 1-6 Electrical Characteristics (Continued) T_a = -20 to +70°C, V_{DD} = 3.3 V, V_{SS} = 0 V

No.	Parameter	Symbol	Conditions	Min	Typ	Max	Unit



Note: V_{DD} = 3.3 V, V_{SS} = 0 V, T_a = 25°C

Figure 1-5 Composite Video Signal Specification

Composite Video Signal Specification

(1)	Signal input amplitude	V_{CI}	The phase difference is in reference to the sync signal input to CVBS.	720		2000	mV
(2)	Caption data amplitude	V_{CDH}		360			
(3)	Sync signal amplitude	V_{SH}		360			
(4)	Maximum CVBS input potential	V_{CBM}				V_{DD}	V
(5)	Minimum CVBS input potential	V_P		V_{SS}			

1.5.1 I²C Interface Timing

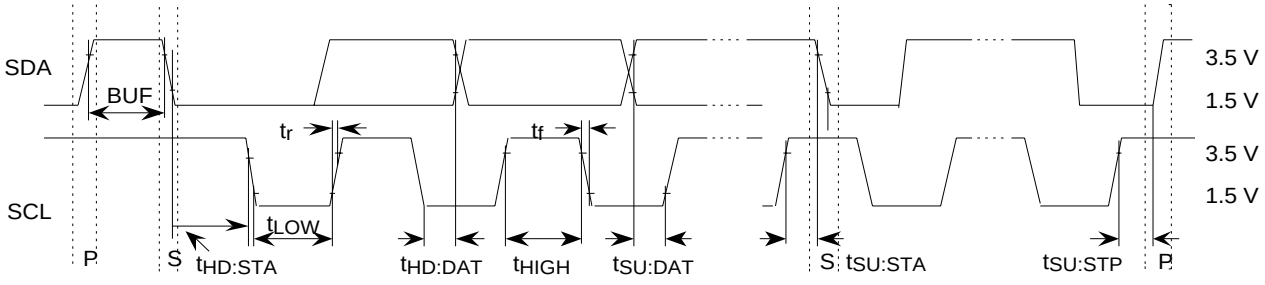


Figure 1-6 I²C Interface Timing

Table 1-7 I²C Timing for Master Transmission (SCL, SDA), Master Reception (SCL), and Slave Transmission (SDA)

No.	Parameter	Symbol	Conditions	Min		Max	Unit
	SCL clock frequency (1)	f_{SCL}	f_{OSC} = 12 to 14.32 MHz			100	kHz
	Bus free time	t_{BUF}	f_{SCL} = 100 kHz	20			µs
	SDA and SCL rise time	t_r				1	
	SDA and SCL fall time	t_f				300	ns

Notes: 1. See section 13 for information on the I²C clock select. All other parameters adhere to the specifications shown above.

2. figure 1-7 shows the software method of attaining the bus free time indicated above. The microcontroller requires 80 machine cycles from the time the stop condition occurs to the time the next start condition occurs.

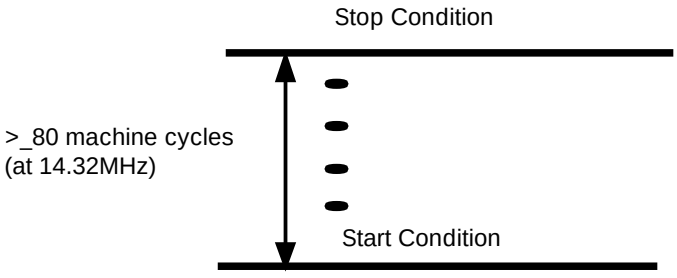


Figure 1-7 Start and Stop Conditions

Table 1-8 I²C Timing for Slave Reception (SCL, SDA), Slave Transmission (SCL), and Master Reception (SDA)

No.	Parameter	Symbol	Conditions	Min		Max	Unit
	SCL clock frequency	f_{SCL}	f_{OSC} = 12 to 14.32 MHz			100	kHz
	Bus free time	t_{BUF}		7.7			µs
	Hold time to start condition	$t_{HD:STA}$		4.7			
	Clock low pulse width	t_{LOW}		4.7			
	Clock high pulse width	t_{HIGH}		4			
	Setup time for start condition	$t_{SU:STA}$		4.7			
	Data hold time	$t_{HD:DAT}$		0			
	Data setup time	$t_{SU:DAT}$		250			ns
	SDA and SCL rise time	t_r				1	µs
	SDA and SCL fall time	t_f				300	ns
	Setup time for stop condition	$t_{SU:STP}$		4.7			µs
	Noise sampling time	t_{NOISE}				1	

1.5.2 HSYNC and VSYNC Input Conditions

Use the flyback H and V for onscreen displays and to identify onscreen fields. Adhere to the input conditions described below in your design to ensure accurate displays.

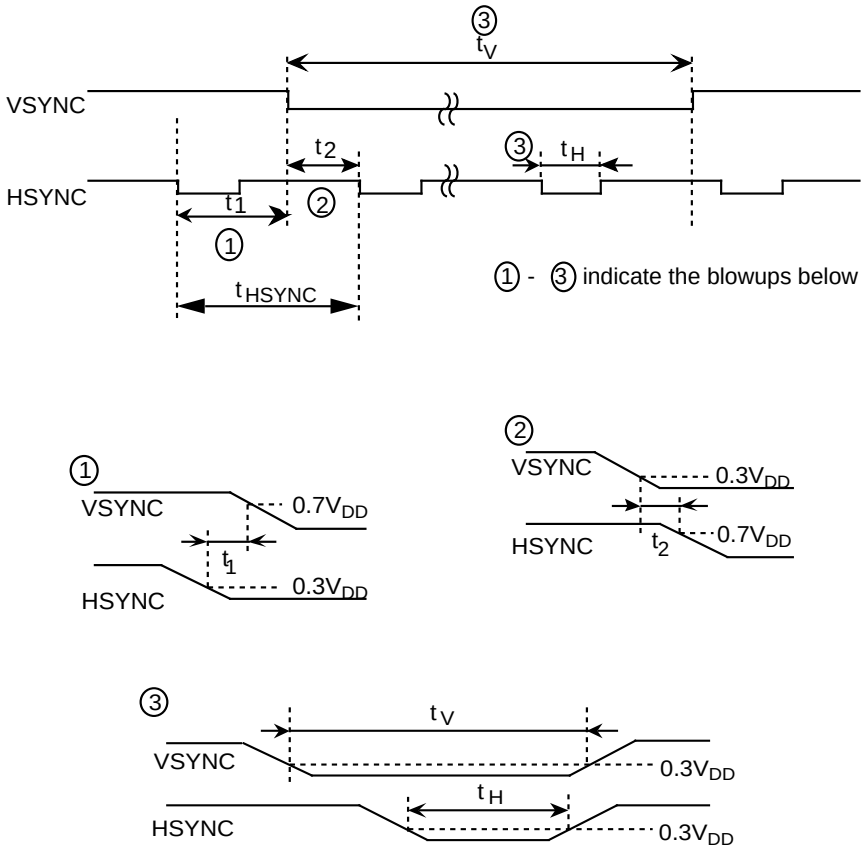


Figure 1-8 HSYNC and VSYNC Input Conditions

Table 1-9 HSYNC and VSYNC Input Conditions

Symbol	Description	Min	Typ	Max	Unit
t_1	Gap between VSYNC leading edge and leading edge of previous HSYNC	0.4		T_{HSYNC} - 0.4	µs
t_2	Gap between VSYNC leading edge and leading edge of following HSYNC	0.4		T_{HSYNC} - 0.4	
t_V	VSYNC pulse width	4.0			
t_H	HSYNC pulse width	4.0		T_{HSYNC} - 4.0	

1.6 Circuit Design Considerations

1.6.1 Considerations when Using the IC

1.6.1.1 Connecting the V_{DD} and V_{SS} Pins

Directly connect all V_{DD} and V_{SS} pins separately to an external power source and GND. Thoroughly double check the pin positions of the IC package and install it on a PCB. The figures below show examples of correct and incorrect connection techniques. Incorrect connection runs the risk of damaging the device by melting the metallization with large currents.

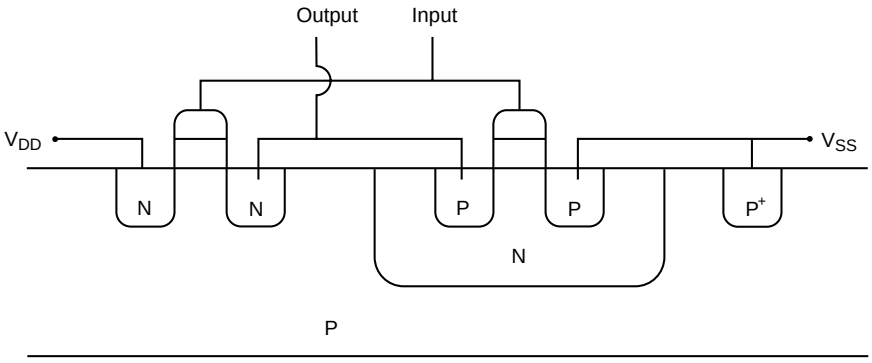


Figure 1-9 Correct Connection Technique for the V_{DD} and V_{SS} Pins

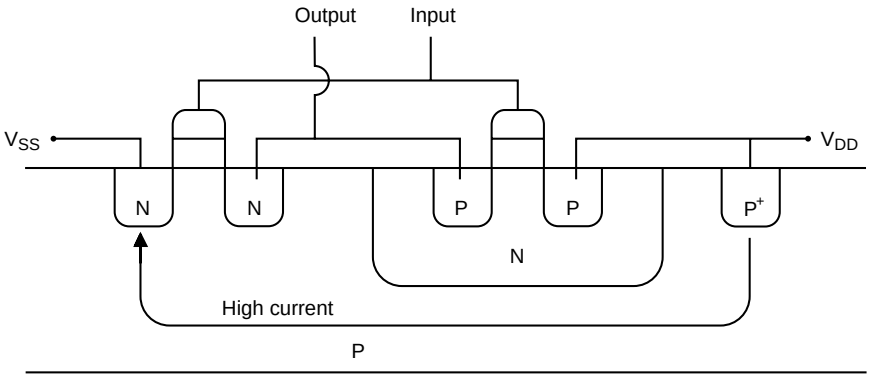


Figure 1-10 Incorrect Connection Technique for the V_{DD} and V_{SS} Pins

1.6.1.2 Operation Considerations

1. To ensure that operation is correct, shield the surface of the package if you are using it where it will be subject to strong electrical fields, such as under a CRT.

2. Double check the operating temperatures before use. Different products have different operating temperature ranges. If, for example, you are using a product guaranteed for +70°C at temperatures higher than this, it may malfunction because it has no operating margin.

3. Double check the operating voltages before use. Different products have different operating voltage ranges.

   - Panasonic cannot guarantee reliability at voltages higher than the guaranteed voltage. (Service life of transistors, for example, may vary with age.)

   - Using a product at a voltage below the guaranteed voltage may cause malfunction since there is no operating margin.

1.6.2 Handling Unused Pins

1.6.2.1 Handling Unused Functions
If you are not using a function, set it so its operation is halted.

1.6.2.2 Handling Unused Pins (Dedicated Output Pins)
Leave any unused pins that are dedicated for output open.

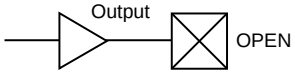


Figure 1-11 Handling Unused Pins (Dedicated for Output)

1.6.2.3 Handling Unused Pins (Dedicated Input Pins)
Pull unused pins that are dedicated for input either up or down by inserting a resistor of at least 10 kΩ. If an unstable input causes both the P channel and N channel transistors of the input inverter to engage, a through-current will flow into the input circuit, causing an increase in current consumption or noise in the chip's internal power supply.

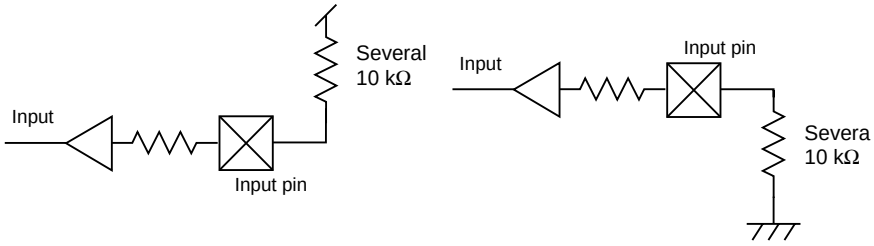
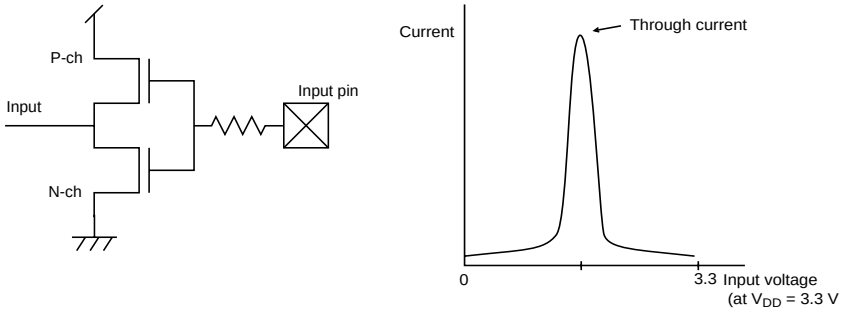


Figure 1-12 Handling Unused Pins (Dedicated for Input)



A. Structure **B. Characteristics**

Figure 1-13 Input Inverter Structure and Characteristics

1.6.2.4 Handling Unused Pins (I/O Pins)

First check the reset value of any unused I/O pins. If the pin output is supposed to be high impedance after a reset (output is off for both P channel and N channel transistors), insert a resistor of at least 10 kΩ to pull the pin up or down so that input is not unstable. If the pin output is on after a reset, leave the pin open.

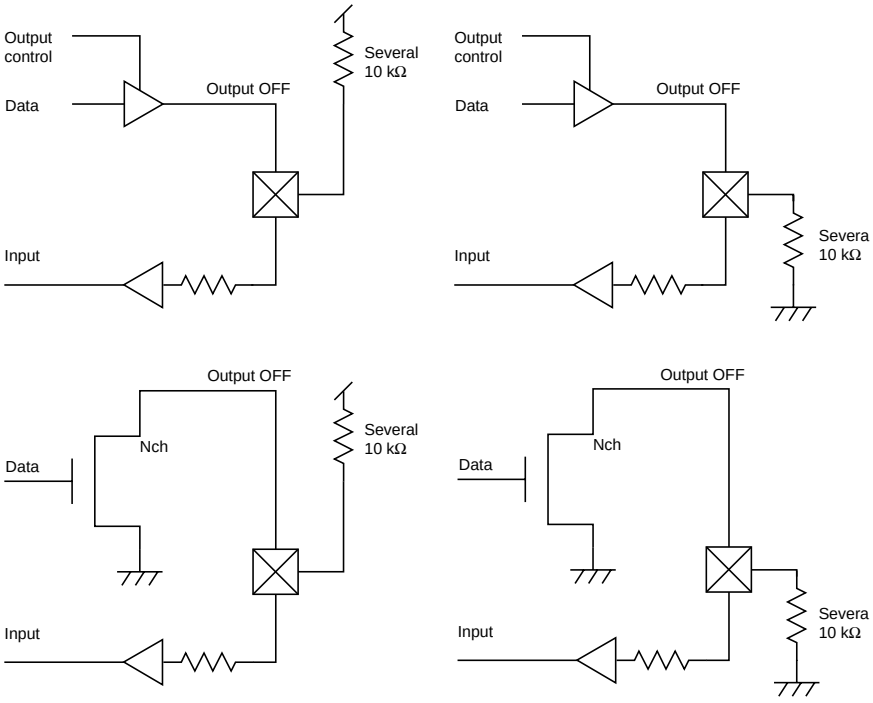


Figure 1-14 Handling Unused I/O Pins (Output Is Hi-Z at Reset)

1.6.3 Turning on Power

1.6.3.1 Power and Input Pin Voltage

Design circuits so that voltage is supplied to input pins after the microcontroller has powered up. Reversing this order may cause latch-ups in the microcontroller which can lead to damaging high currents.

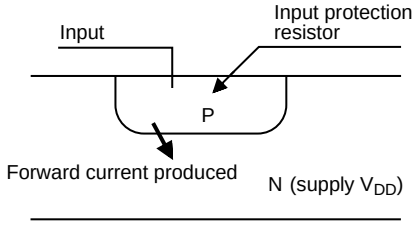


Figure 1-15 Power and Input Pin Voltage

1.6.3.2 Power and Reset Input Voltage

Design in sufficient time after the supply voltage powers up for the microcontroller to recognize the reset pin voltage as a reset signal.

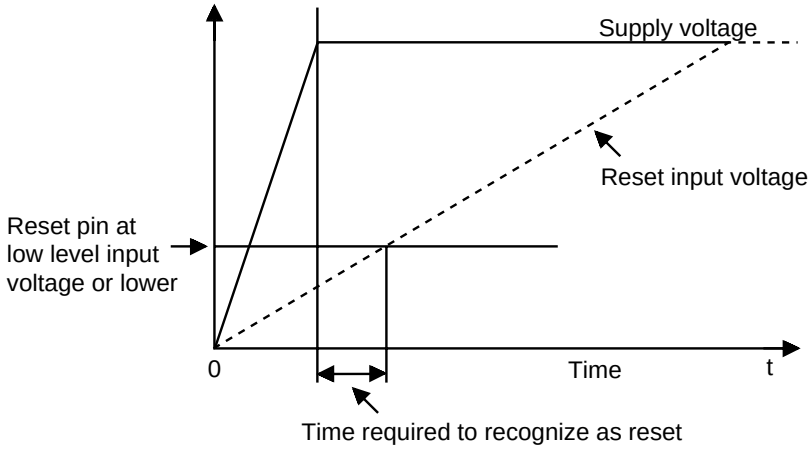


Figure 1-16 Power and Reset Input Voltage

1.6.4 Power Circuit

1.6.4.1 Design Considerations for the Power Circuit

MOS logic devices such as microcontrollers use fast, large-scale integration designs, so use a power circuit that provides a sufficient margin. Consider a power system like that shown in Figure 1-18, making sure you first evaluate AC line noise and check the ripple when driving LEDs and the like.

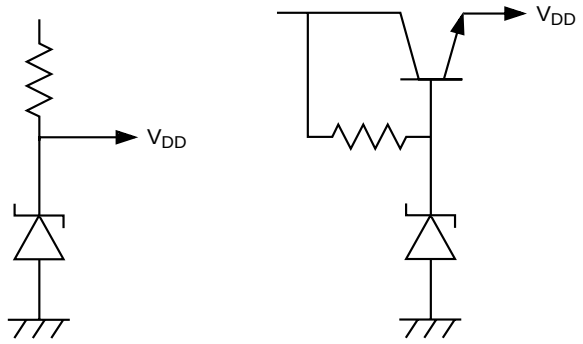


Figure 1-17 Design Considerations for the Power Circuit

1.6.4.2 Sample Power Circuit (Emitter-Follower Type)

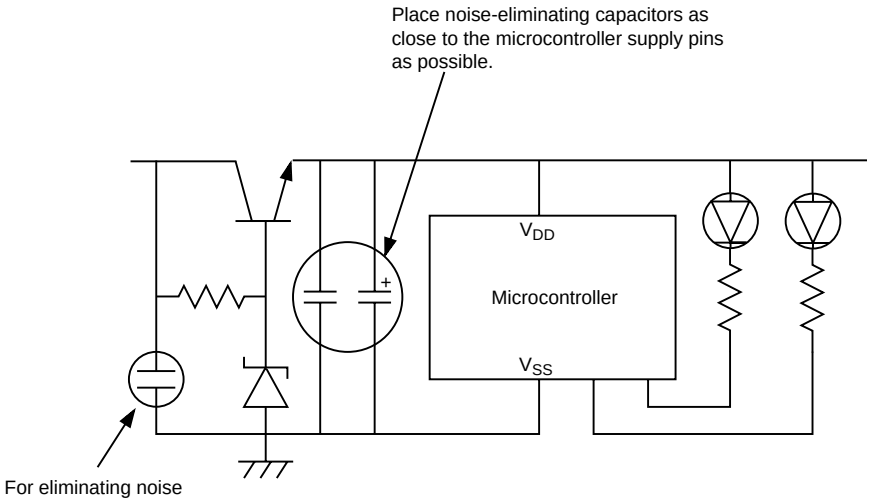


Figure 1-18 Sample Power Circuit (Emitter-Follower Type)

2 Basic CPU Functions

2.1 Description

The MN101C series of microcontrollers are designed with a flexible, optimized hardware architecture for embedded applications in electronic equipment. Its simple, efficient instruction set helps make it both economical and fast. It offers the following features:

- A 4-bit instruction word length minimizing instruction code size.
  Code is compressed by adopting a basic instruction word length of 1 byte and allowing instruction length to vary in 4-bit units. This prevents code expansion even though the simple instruction set is limited to load/stores of data transfers to memory.

- A minimum instruction execution time of 1 cycle (279.3 ns).

- A C-compatible register set minimized for the simplest architecture.
  The instruction set analyzes code generated by the C compiler and code from assembler programming. The set represents a trade-off of hardware size against performance. The instruction set is thus the smallest C-oriented set available, and is notable for its simplicity.

See the *MN101C Series LSI User Manual* for more information.

2.2 Features

Table 2-1 Basic CPU Features

Parameter	Description	
Architecture	Load/store architecture	
	Six registers	Four 8-bit data registers and two 16-bit address registers
	Miscellaneous	19-bit PC, 8-bit PSW, and 16-bit SP
Instructions	Number of instructions	37
	Addressing modes	9
	Instruction word length	Basic part: 1 byte (minimum) Extensions: 0.5 bytes × n ($0 \leq n \leq 9$)
Basic performance	Internal operating frequency (max)	3.58 MHz
	Fastest instruction execution	1 cycle
	Fastest operation between registers	2 cycles
	Fastest load/store	2 cycles
	Conditional jumps	2-3 cycles
Pipeline	Three stages (instruction fetch, decoding, and execution)	
Address space	256 Kbytes (maximum 64 Kbytes for data)	
	Space shared by instructions and data	
External bus	Addresses	18 bits (max)
	Data	8 bits
	Shortest bus cycle	1 clock (279.3 ns)
Interrupts	Vector interrupt system	3 levels
Power down modes	STOP and HALT modes	

2.3 Block Functions

2.3.1 Block Diagram

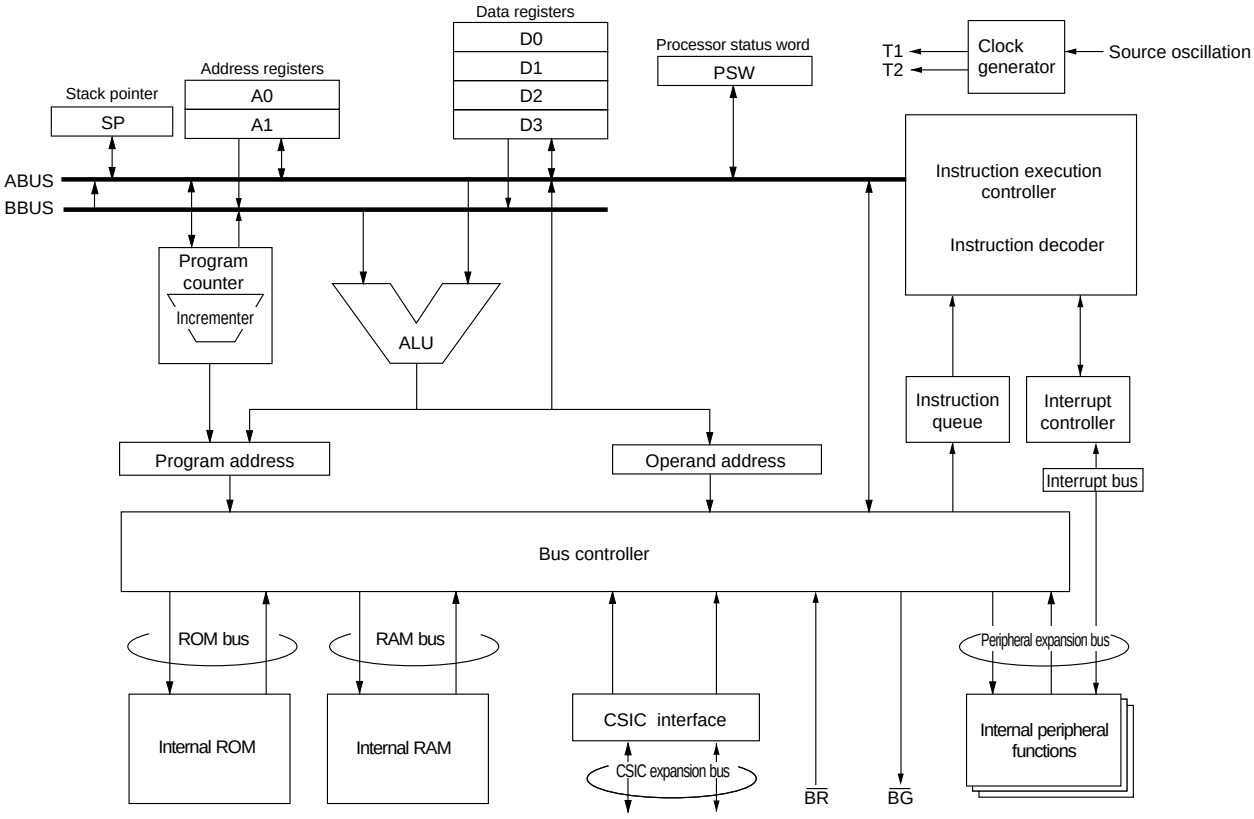


Figure 2-1 Block Structure and Functions

2.3.2 Block Description

Table 2-2 Block Description

Block	Description
Clock Generator	An oscillation circuit connected to a quartz or ceramic oscillator supplies the clock to all blocks within the CPU.
Program counter	The program counter generates addresses for queued instructions. Normally it increments based on the sequencer indications, but for branch instructions it is set to the branch head address. For interrupt servicing, it is set to the result of the ALU operation.
Instruction queue	This block contains up to two bytes of prefetched instructions.
Instruction decoder	The instruction decoder decodes the contents of the instruction queue, generates (in the proper sequence) the control signals necessary for executing the instruction, and controls every chip block involved in instruction execution.
Instruction execution controller	This block controls the operation of all blocks within the CPU using the results from the instruction decoder and interrupt requests.
ALU	Arithmetic and logic unit. This block calculates operand addresses for data arithmetic operations, logic operations, shift operations, and relative indirect register addressing.
Internal ROM and RAM	These memory blocks contain the program, data, and stack areas.
Address registers	The address registers store the memory addresses that will be accessed during data transfers. In relative indirect register addressing mode, they store the base address.
Data registers	These registers store data used for operation. You can link two 8-bit data registers and use them as a 16-bit register.
Interrupt controller	This block detects interrupt requests from peripheral function blocks and requests that the CPU service the interrupt.
Bus controller	This block controls the connection between the CPU's internal and external buses. It also contains a bus arbitration function.
Internal peripheral functions	MN101C series devices contain a wide range of internal peripheral devices, such as timers and ADCs.

2.4 CPU Control Registers

The MN101C46F uses memory-mapped I/Os. The registers of its peripheral circuits are placed in memory space (x'03D00'-x'03FFF'). The CPU's control registers are also mapped to this memory space.

Table 2-3 CPU Control Registers

Address	Register	R/W	Description
x'03F00'	CPUM	R/W*	CPU mode control register
x'03F01'	MEMCTR	R/W	Memory control register
x'03F0D'	OSCMD	R/W	Oscillation frequency control register
x'03FE0'	Reserved	R/W	(For debugging)
x'03FE1'	NMICR	R/W	Nonmaskable interrupt control register (See section 3, "Interrupts," on page 59.)
x'03FE2'–x'03FFE'	xxxICR	R/W	Maskable interrupt control registers (See section 3, "Interrupts," on page 59.)
x'03FFF'	Reserved	(Use to read interrupt vector information in hardware servicing of interrupts.)	

Note: Some CPUM bits are read-only.

2.5 Organization of Instruction Execution Controller

The instruction execution controller is comprised of four blocks: memory, the instruction queue, the instruction register, and the instruction decoder. Instructions are fetched in bytes, and temporarily stored in the 2-byte instruction queue. They are transferred from the instruction queue in bytes or half bytes to the instruction register, where they are decoded by the instruction decoder.

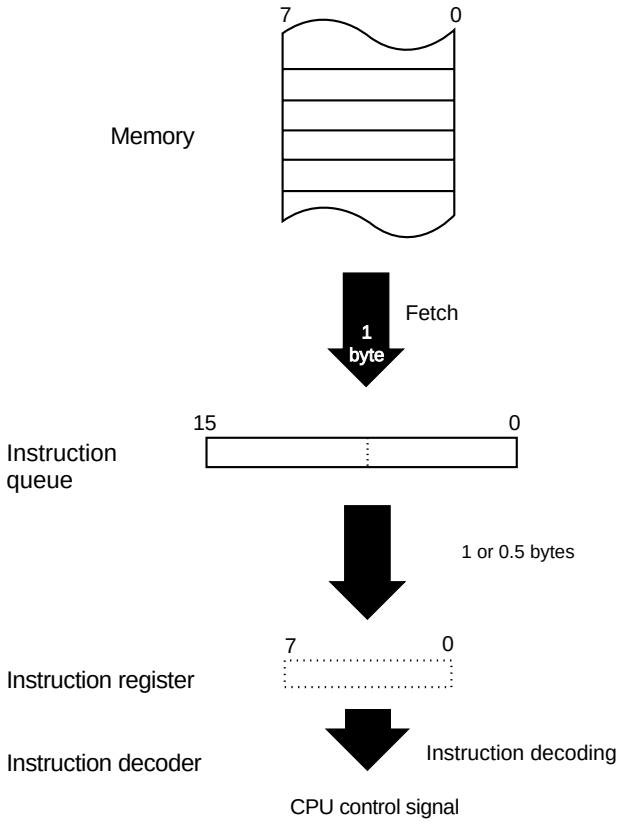


Figure 2-2 Organization of the Instruction Execution Controller

2.6 Pipeline Processing

Pipeline processing refers to the reading and decoding of instructions simultaneous with instruction execution so that the next instruction is ready for processing as soon as execution of the first instruction ends. Pipeline processing allows instructions to execute continuously, increasing speed of execution. It is performed by the instruction queue and instruction decoder.

The instruction queue is a two-stage instruction pre-fetch buffer. It is controlled so that whenever the queue is empty during a cycle of instruction execution, the next instruction is fetched. The first word of the instruction to be executed (operation code) is stored in instruction register during the last cycle of instruction execution. At this time, the next operand or operation code is fetched to the instruction queue, so execution can begin immediately if a direct address (da) or immediate data (imm) is in the first cycle of the next instruction to be executed. In some instructions, such as jump instructions, the instruction queue may be empty when the operation code to be executed next is stored in the instruction register during the last cycle. The queue may therefore produce a wait of one machine cycle if the instruction queue is empty and a direct address (da) or immediate data (imm) is needed in the first cycle of the instruction to be executed.

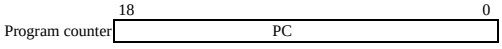
Hardware automatically controls the instruction queue, so you do not need to pay any particular attention to it when writing a program. When you are calculating instruction execution time, however, be aware of how the instruction queue operates. The instruction decoder will generate control signals in each instruction execution cycle according to microprogram control. The instruction decoder decodes the contents of the instruction queue in the cycle before the control signal is required.

2.7 Address Registers

The address registers consist of the program counter (PC), the address registers (A0, A1), and the stack pointer (SP).

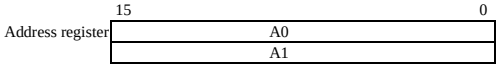
2.7.1 Program Counter (PC)

This register indicates the address of the instruction being executed. Instructions are separated by half bytes (4 bits), so the program counter requires 19 bits to express the 256-Kbyte instruction space. The LSB of the program counter indicates the half byte. The program counter resets to the value stored in the vector table at address x'04000'.



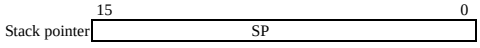
2.7.2 Address Registers (A0, A1)

These registers are used as address pointers. They can be used for operation instructions that calculate addresses (addition, subtraction, or compare). The address registers contain pointers (2 bytes of data), so transfers to memory are normally in 16-bit units. You can transfer with either odd or even addresses. They are undefined at reset.



2.7.3 Stack Pointer (SP)

This register specifies the top address of the stack area. It is decremented during saves and incremented when restored. It is undefined at reset.

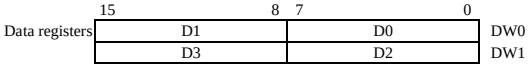


2.8 Operations Registers

The operations registers consist of four data registers (D0, D1, D2, and D3).

2.8.1 Data Registers (D0, D1, D2, and D3)

The data registers are all 8-bit general-purpose registers. They can be used for arithmetic, logical, or shift operations, or for transfers of data to memory. D0 and D1 can be paired and handled as a 16-bit registers, as can D2 and D3. The data registers are undefined at reset.



2.8.2 Processor Status Word

The processor status word (PSW) is an 8-bit register that stores an operation result flag, an interrupt mask level, and a maskable interrupt enable flag. The PSW is automatically saved to the stack when an interrupt occurs, and automatically restored from the stack after the interrupt is recovered.

PSW: Processor Status Word

Bit:	7	6	5	4	3	2	1	0
	Reserved	MIE	IM1	IM0	VF	NF	CF	ZF
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Reserved: Always set to 0.

MIE: Maskable interrupt enable

0: Disables all maskable interrupts

1: Enables interrupts individually (xxxLVn, xxxIE).

IM[1:0]: Interrupt mask level

Control acceptance of maskable interrupts.

VF: Overflow flag

0: No overflow occurred.

1: An overflow occurred.

NF: Negative flag

0: The MSB of the operation result is a 0.

1: The MSB of the operation result is a 1.

CF: Carry flag

0: The MSB did not produce a carry up or down.

1: The MSB produced a carry up or down.

ZF: 0 flag

0: The operation result is not all zeros.

1: The operation result is all zeros.

■ **Maskable Interrupt Enable Flag (MIE)**

This flag enables maskable interrupts. Maskable interrupts are accepted when MIE is 1. When MIE is 0, all maskable interrupts are disabled regardless of the interrupt mask level (IM[1:0]) within the PSW. Interrupts do not change MIE.

■ **Interrupt Mask Level Field (IM[1:0])**

The interrupt mask level field (IM[1:0]) controls acceptance of maskable interrupts according to the interrupt level of the interrupt source. The 2-bit field defines levels 0 through 3. Level 0 is the highest interrupt mask level. Interrupt requests are accepted only when the level set in the interrupt level flag (xxxLVn) of the interrupt control register (xxxICR) is higher than the interrupt mask level set here (when the setting is smaller). When an interrupt is accepted, its interrupt level is set in IM1 and IM0, so interrupts of that level or lower are not accepted until servicing of the received interrupt is completed.

Table 2-4 Interrupt Mask Levels and Interrupt Acceptance

	Interrupt Mask Level			
	IM1	IM0	Mask Priority	Interrupt Levels Accepted
Mask level 0	0	0	High ⇑ ⇓ Low	Only the non-maskable interrupt (NMI)
Mask level 1	0	1		NMIs and level 0
Mask level 2	1	0		NMIs and levels 0-1
Mask level 3	1	1		NMIs and level 0-2

■ **Overflow Flag (VF)**

VF is set to 1 when the result of an arithmetic operation causes an overflow of a signed number. If there is no overflow, VF is 0. Use VF when working with signed data.

■ **Negative Flag (NF)**

NF is set to 1 when the MSB of an operation result is 11; when the MSB is 0, NF is 0 also. Use NF when working with signed data.

■ **Carry Flag (CF)**

CF is set to 1 when the MSB produces a carry up or down as a result of an operation; if there is no carry, CF is 0.

■ **Zero Flag (ZF)**

ZF is set to 1 when all bits of the operation result are 0; otherwise ZF is 0.

2.9 Addressing Modes

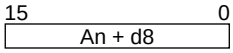
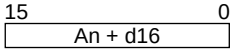
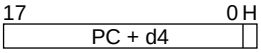
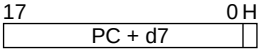
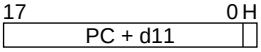
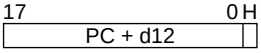
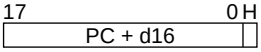
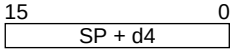
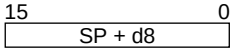
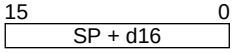
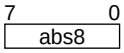
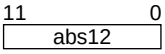
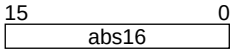
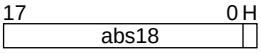
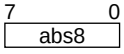
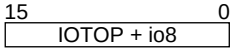
The MN101C46F has nine addressing modes. The usable modes are preset for each instruction.

- Register direct
- Immediate value
- Register indirect
- Register relative indirect
- Stack relative indirect
- Absolute
- RAM short
- I/O short
- Handy addressing

The MN101C46F provides the addressing modes most frequently used by C compilers. Data transfer instructions can use any of the addressing modes. Since relative values can be specified in half bytes (4 bits), instruction code can be shortened. Handy addressing reuses addresses that have accessed memory, so it can only be used with store instructions. It can be combined with absolute addressing to shorten code. There are seven addressing modes that can be used to transfer data to memory: register indirect, register relative indirect, stack relative indirect, absolute, RAM short, I/O short, and handy. Two addressing modes can be used with operation instructions: register direct and immediate value. See the MN101C series instruction manual for details.

The MN101C46F uses a basic 8-bit data access. You can use either an odd address or an even address to specify addresses for a 16-bit data access.

Table 2-5 Address Space

Addressing Mode		Valid Addresses	Description
Register direct	Dn/DWn An/SP PSW	—	Specifies the register directly. Only internal registers can be specified.
Immediate values	imm4/imm8 imm16	—	Directly specifies an operand value, mask value, or the like to be appended to an instruction code.
Register indirect	(An)		Specifies addresses using address registers.
Register relative indirect	(d8, An)		Specifies addresses using address registers and an 8-bit displacement.
	(d16, An)		Specifies addresses using address registers and a 16-bit displacement.
	(d4, PC) (Jump instructions only)		Specifies addresses using the program counter, a 4-bit displacement, and the H bit.
	(d7, PC) (Jump instructions only)		Specifies addresses using the program counter, a 7-bit displacement, and the H bit.
	(d11, PC) (Jump instructions only)		Specifies addresses using the program counter, a 11-bit displacement, and the H bit.
	(d12, PC) (Jump instructions only)		Specifies addresses using the program counter, a 12-bit displacement, and the H bit.
	(d16, PC) (Jump instructions only)		Specifies addresses using the program counter, a 16-bit displacement, and the H bit.
Stack relative indirect	(d4, SP)		Specifies addresses using the stack pointer and a 4-bit displacement.
	(d8, SP)		Specifies addresses using the stack pointer and an 8-bit displacement.
	(d16, SP)		Specifies addresses using the stack pointer and a 16-bit displacement.
Absolute	(abs8)		Specifies addresses using operand values appended to instruction code. You can specify the optimum operand length for the address to be specified.
	(abs12)		
	(abs16)		
	(abs18) (Jump instructions only)		
RAM short	(abs8)		You can specify addresses with an 8-bit offset from address x'00000'.
I/O short	(io8)		You can specify addresses using an 8-bit offset from the top address (x'03F00') of the special register area.
Handy	(HA)	—	This type of addressing reuses an address that has accessed memory. It can only be used with MOV and MOVW instructions. Combine it with absolute addressing to keep code size small.

Note: H: Hybrid bit.

2.10 Memory Space

2.10.1 Memory Modes

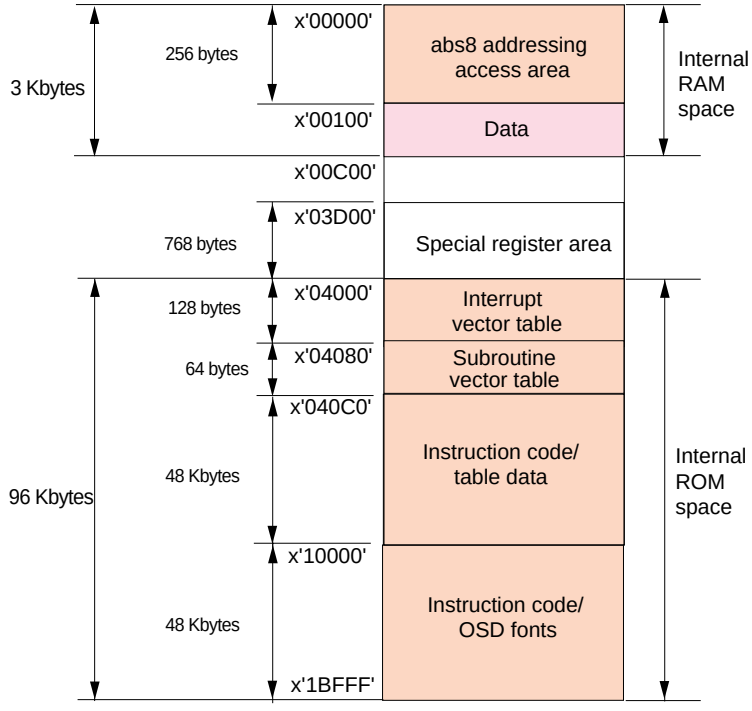
Memory space includes the ROM area (the program area for instructions), the RAM area (where data can be read or written), and a memory-mapped special register area. The MN101C46F supports a single chip mode. Table 2-6 shows settings for this mode.

Table 2-6 Memory Mode Settings

Memory Mode	MMOD Pin	EXMEM flag (MEMCTR register)
Single chip mode	L	0
Disabled	L	1
Disabled	H	—

2.10.2 Single Chip Mode

Single chip mode is a memory mode in which the system is constructed entirely of internal memory. It is an optimized memory model that allows you to build the highest performance system architecture. In single chip mode, both ROM and RAM are internal memory. The MN101C46F has 3 Kbytes of RAM space and 96 Kbytes of ROM space.



Note: MMOD=L

Figure 2-3 Single Chip Mode

2.10.3 Special Function Registers

The MN101C46F assigns x'03D00'-x'03FFF' in memory space as a special function register area (I/O space). Its special registers are placed as follows.

Table 2-7 Register Map: x'03D00'-x'03EFF'

16 MSBs	4 LSBs																Description
	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	
x'03D00'																	
x'03D10'																	
x'03D20'																	
x'03D30'																	
x'03D40'																	
x'03D50'																	
x'03D60'																	
x'03D70'					SLCNT2	SLCNT1											VBI (1)
x'03D80'																	
x'03D90'																	
x'03DA0'																	
x'03DB0'																	
x'03DC0'																	
x'03DD0'																	
x'03DE0'																	
x'03DF0'					SLCNT2W	SLCNT1W											VBI (2)
x'03E00'	CRI4FQW	CRI3FQW	CRI2FQW	CRI1FQW	CAPDATAH	CAPDATA	ACQ1H	ACQ1	VBIIRQ	HNUM	SLSF	SLHD	MAX	MIN	SLCNT	FC	VBI 1 registers
x'03E10'	FCPNUMH	FCPNUM	STAPH	STAP	DATAEH	DATAE	DATASH	DATAS	CRI2EH	CRI2E	CRI2SH	CRI2S	CRI1EH	CRI1E	CRI1SH	CRI1S	
x'03E20'	CRI4FQWW	CRI3FQWW	CRI2FQWW	CRI1FQWW	CAPDATAWH	CAPDATAW	ACQ1WH	ACQ1W	VBIIRQW	HNUMW	SLSFW	SLHDW	MAXW	MINW	SLCNTW	FCW	VBI 2 registers
x'03E30'	FCPNUMWH	FCPNUMW	STAPWH	STAPW	DATAEWH	DATAEW	DATASWH	DATASW	CRI2EWH	CRI2EW	CRI2SWH	CRI2SW	CRI1EWH	CRI1EW	CRI1SWH	CRI1SW	
x'03E40'	HSEP1H	HSEP1	CLAMPH	CLAMP	SPLVH	SPLV	BPLV	SYNCMIN	BPPSTH	BPPST	SCMINGH	SCMING	FQSELH	FQSEL	NFSELH	NFSEL	Sync separator 1 registers
x'03E50'			CLPCND1H	CLPCND1		HVCOND	VCNTH	VCNT	HDISTWH	HDISTW	HLOCKLVH	HLOCKLV	FIELDH	FIELD	HSEP2H	HSEP2	
x'03E60'	HSEP1WH	HSEP1W	CLAMPWH	CLAMPW	SPLVWH	SPLVW	BPLVW	SYNCMINW	BPPSTWH	BPPSTW	SCMINGWH	SCMINGW	FQSELWH	FQSELW	NFSELWH	NFSELW	Sync separator 2 registers
x'03E70'			CLPCND1WH	CLPCND1W		HVCONDW	VCNTWH	VCNTW	HDISTWWH	HDISTWW	HLOCKLVWH	HLOCKLVW	FIELDWH	FIELDW	HSEP2WH	HSEP2W	

Table 2-7 Register Map: x'03D00'-x'03EFF' (Continued)

16 MSBs	4 LSBs																Description
	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	
x'03E80'						I2C BSTS		I2C BRST		I2C CLK		I2C MYAD	I2C DRECH	I2C DREC	I2C DTRMH	I2C DTRM	I^2C interface registers
x'03E90'	TDCHR	TDCLR		TDCC		PWM5		PWM4		PWM3		PWM2		PWM1		PWM0	PWM registers
x'03EA0'				RMLD		RMTR		RMSR		RMCS		RMTC		RMIR		RMIS	Remote signal receiver registers
x'03EB0'	EVODH	EVOD	HCOUNTH	HCOUNT		OSD3		OSD2		OSD1		RAMEND		GROMEND		CROMEND	OSD control registers
x'03EC0'	IAPH	IAP	IVPH	IVP	IHPH	IHP		SHTC	HSHT1H	HSHT1	HSHT0H	HSHT0	VSHT1H	VSHT1	VSHT0H	VSHT0	
x'03ED0'										WBSHD		BBSHD		FRAME		COLB	
x'03EE0'	PLT17	PLT16	PLT15	PLT14	PLT13	PLT12	PLT11	PLT10	PLT07	PLT06	PLT05	PLT04	PLT03	PLT02	PLT01	PLT00	
x'03EF0'	PLT37	PLT36	PLT35	PLT34	PLT33	PLT32	PLT31	PLT30	PLT27	PLT26	PLT25	PLT24	PLT23	PLT22	PLT21	PLT20	
x'03F00'			OSCMD							ACTMD			DLYCTR	WDCTR	MEMCTR	CPUM	CPU mode and memory control
x'03F10'												P4OUT	P3OUT	P2OUT	P1OUT	P0OUT	I/O port output
x'03F20'				P4MD	P3MD	P2MD	P1MD	P0MD				P4IN	P3IN	P2IN	P1IN	P0IN	I/O port input
x'03F30'	ROMCENH	ROMCEN										P4DIR	P3DIR	P2DIR	P1DIR	P0DIR	I/O port I/O mode control
x'03F40'		PCNT2				PCNT0						P4PLU	P3PLU	P2PLU	P1PLU	P0PLU	I/O port pullup resistor control
x'03F50'	CK3MD	CK2MD	TM3MD	TM2MD	TM3OC	TM2OC	TM3BC	TM2BC									Timer control
x'03F60'	PSCMD									CK4MD		TM4MD		TM4OC		TM4BC	
x'03F70'	AMCHIM7	AMCHIL7	AMCHIM6	AMCHIL6	AMCHIM5	AMCHIL5	AMCHIM4	AMCHIL4	AMCHIM3	AMCHIL3	AMCHIM2	AMCHIL2	AMCHIM1	AMCHIL1	AMCHIM0	AMCHIL0	ROM correction control 1 (correction address)
x'03F80'	AMCHIMF	AMCHILF	AMCHIME	AMCHILE	AMCHIMD	AMCHILD	AMCHIMC	AMCHILC	AMCHIMB	AMCHILB	AMCHIMA	AMCHILA	AMCHIM9	AMCHIL9	AMCHIM8	AMCHIL8	
x'03F90'	AMCHIHF	AMCHIHE	AMCHIHD	AMCHIHC	AMCHIHB	AMCHIHA	AMCHIH9	AMCHIH8	AMCHIH7	AMCHIH6	AMCHIH5	AMCHIH4	AMCHIH3	AMCHIH2	AMCHIH1	AMCHIH0	
x'03FA0'																	Serial interface control
x'03FB0'												ANBUF1		ANCTR2	ANCTR1	ANCTR0	Analog interface control
x'03FC0'																	
x'03FD0'	CHDATF	CHDATE	CHDATD	CHDATC	CHDATB	CHDATA	CHDAT9	CHDAT8	CHDAT7	CHDAT6	CHDAT5	CHDAT4	CHDAT3	CHDAT2	CHDAT1	CHDAT0	ROM correction control 2 (correction data)
x'03FE0'			TM4ICR	TM3ICR	TM2ICR				IRQ5ICR	IRQ4ICR	IRQ3ICR	IRQ2ICR	IRQ1ICR	IRQ0ICR	NMICR		Interrupt control
x'03FF0'					RMCICR	ADICR		VBIV1ICR	VBIV0ICR	I2CICR	OSDICR	VBI1ICR	VBI0ICR				

2.11 Bus Interface

2.11.1 Bus Controller

The MN101C series limits the effects of bus line loads and speeds up operation by separating the buses to which internal memory and internal peripheral functions are connected.

Bus control uses four types of buses: a ROM bus, a RAM bus, a peripheral expansion bus (I/O bus), and an external expansion bus. These buses are connected respectively to internal ROM, internal RAM, internal peripheral functions, and the external interface. The functions of the bus controller include parallel instruction supply and data access, handling slow devices when accessing external space, and arbitrating bus usage when an external bus master device is connected. A block diagram of the bus controller is shown below.

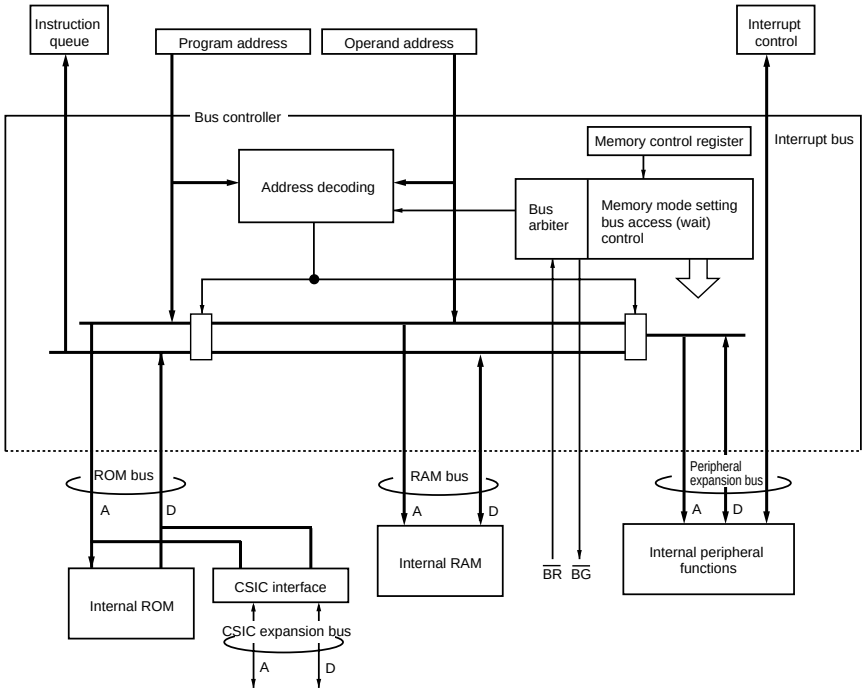


Figure 2-4 Function Block Diagram of the Bus Controller

The CSIC expansion bus accesses the OSD, closed-caption, I^2C, PWM, and remote signal receiver registers. Accesses of the CSIC expansion bus always have one wait and are set up in the memory control register (MEMCTR). You can set the number of waits for the peripheral expansion bus (I/O bus) connected to the internal peripheral functions.

2.11.2 Control Registers

Two registers control bus interface functions: the memory control register (MEMCTR) and the expansion address control register (EXADV).

MEMCTR: Memory Control Register **x'03F01'**

Bit:	7	6	5	4	3	2	1	0
	IOW1	IOW0	IVBM	EXMEM	EXWH	IRWE	EXW1	EXW0
Reset:	1	1	0	0	1	0	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

IOW[1:0]: Set number of waits when accessing special register area.

00: No waits (279.3 ns with 14.32-MHz bus cycle)
01: Setting disabled (419.0 ns with 14.32-MHz bus cycle)
10: Two waits (558.6 ns with 14.32-MHz bus cycle)
11:Setting disabled (698.3 ns with 14.32-MHz bus cycle)

IVBM: Sets base address for interrupt vector table.

0: Interrupt vector base = x’04000’
1: Interrupt vector base = x’00100’

EXMEM: Switches external memory expansion mode.

0: Do not expand to external memory.
1: Disabled.

EXWH: Switches between fixed wait mode and handshake mode.

0: Disabled.
1: Fixed wait mode.

IRWE: Sets software writing of interrupt request flag.

0: Writing with software disabled. The state of the interrupt request flag (xxxIR) does not change even if data is written to an interrupt control register (xxxICR).
1: Writing with software enabled.

EXW[1:0]: Set fixed wait states

00: No waits (279.3 ns with 14.32-MHz bus cycle)
01: Setting disabled (419.0 ns with 14.32-MHz bus cycle)
10: Two waits (558.6 ns with 14.32-MHz bus cycle)
11:Setting disabled (698.3 ns with 14.32-MHz bus cycle)

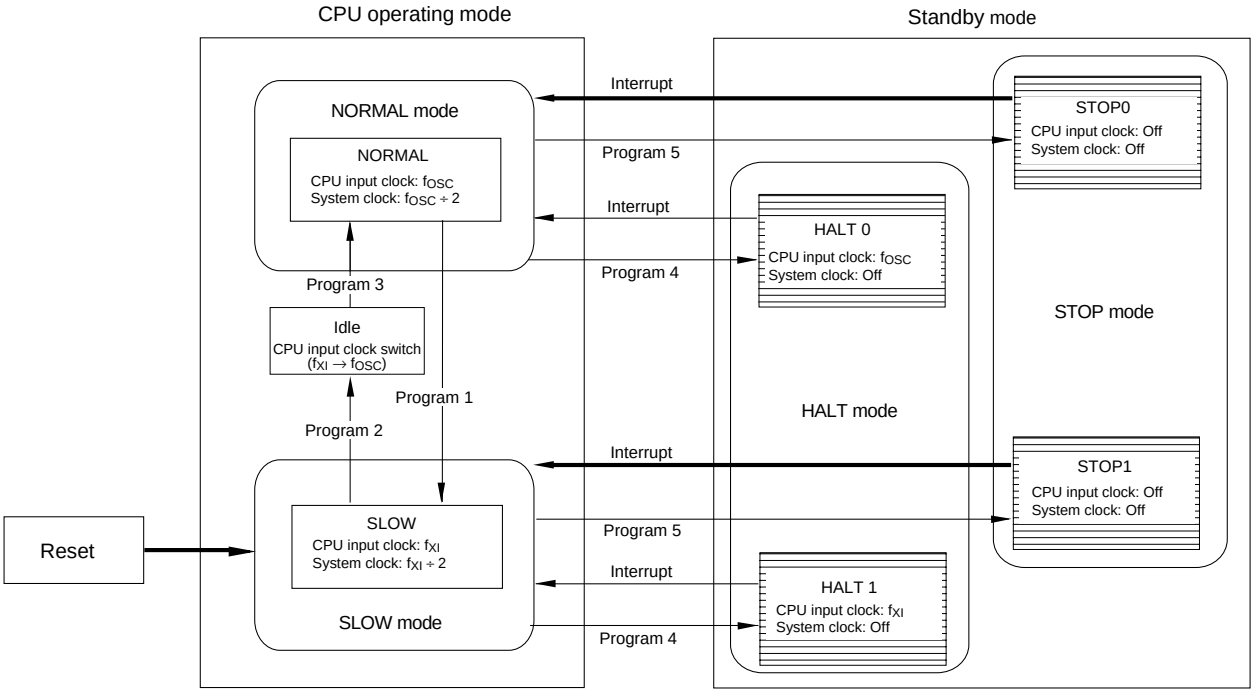
The wait space for the EXW[1:0] applies to devices connected to the CSIC interface. Three waits are inserted when the reset is cleared. Be sure to set the start of the initialization routine to either no waits or two waits.

The wait space for IOW[1:0] is the special register area (I/O space) at x'03F00'-x'03FFF'. Three waits are inserted when the reset is cleared. Be sure to set the start of the initialization routine to either no waits or two waits. Select no waits for high-performance system architectures.

2.12 Standby Function

2.12.1 Overview

The MN101C46F has two sets of oscillation pins for the system clock (high and low oscillation). It has two CPU operating modes (NORMAL and SLOW) and two standby modes (HALT and STOP). You can decrease power consumption by making effective use of these modes.



Notes: 1. ≡≡≡: CPU off

2. ◄━━: Add oscillator stabilization wait
3. f_{OSC}: High-speed operating clock (external oscillation ÷ 2)
4. f_{XI}: Low-speed operating clock (external oscillation ÷ 8)

Figure 2-5 Transitions between Operating Modes

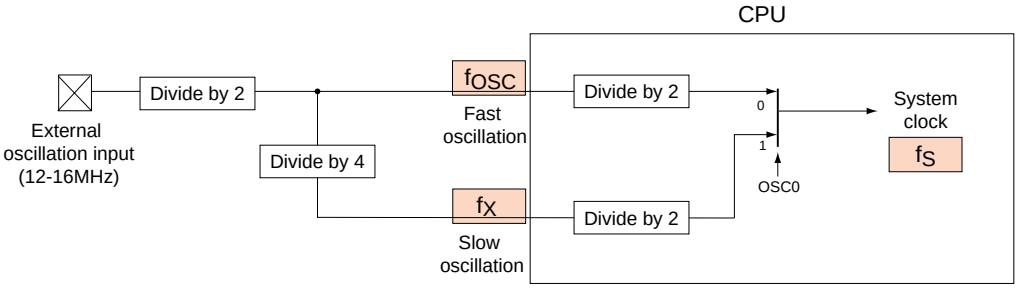


Figure 2-6 Clock Switching Circuit

The HALT States (HALT0 and HALT1)

- In the HALT states, the CPU is halted but the oscillator continues to run. You can return immediately to the operating state with an interrupt.
- In HALT0, both the fast and slow oscillators run. An interrupt returns operation to NORMAL mode.

The STOP States (STOP0 and STOP1)

- In these states, both the CPU and the oscillators stop. Use an interrupt to restart the oscillators, and then return to operating mode after oscillation has stabilized.
- From STOP0, an interrupt returns operation to NORMAL mode.
- From STOP1, an interrupt returns operation to SLOW mode.

Slow Operation (SLOW)

In this mode, programs run at the slow operating clock. This reduces power consumption.

Idling (IDLE)

Idling is used to make the program wait for the fast operating clock to stabilize when going between SLOW mode and NORMAL mode.

Pay particular attention when reducing power supply by invoking STOP or HALT mode that the current flowing to or from pins and input pin levels are not unstable. For output pins, either match output to an external level or control the direction from the input side. For input pins, fix the level externally.

The MN101C46F has one system-clock generating circuit set. It then produces two clocks OSC and XI by dividing the system clock internally. OSC is the fast clock (for NORMAL mode) and XI is the slow clock (for SLOW mode). Use the CPU mode control register (CPUM) to move between NORMAL and SLOW mode or to invoke standby modes. The normal reset operations and interrupts are available as sources to return from standby modes. Waits for oscillation to stabilize are inserted during reset operations and when returning from STOP mode; they are not inserted when returning from HALT mode. The CPU automatically returns the system clock's oscillation mode to the mode it was in prior to invoking standby mode.

2.12.2 CPU Mode Control Register

Use the settings of flags in the CPU mode control register (CPUM) to change modes.

CPUM: CPU Mode Control Register **x'03F00'**.

Bit:	7	6	5	4	3	2	1	0
	—	OSC SEL1	OSC SEL0	OSC DBL	STOP	HALT	OSC1	OSC0
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 2-8 Controlling the Operating Mode and Generating/Halting Clock Oscillation

					Modes			
Operating Mode	**STOP**	**HALT**	**OSC1**	**OSC0**	**OSCI/ OSCO**	**XI/XO**	**CPU Input Clock**	**CPU**
NORMAL	0	0	0	0	Oscillates	Oscillates	OSCI	Runs
IDLE	0	0	0	1	Oscillates	Oscillates	XI	Runs
SLOW	0	0	1	1	Oscillates	Oscillates	XI	Runs
HALT0	0	1	0	0	Oscillates	Oscillates	OSCI	Stopped
HALT1	0	1	1	1	Oscillates	Oscillates	XI	Stopped
STOP0	1	0	0	0	Stopped	Stopped	Stopped	Stopped
STOP1	1	0	1	1	Stopped	Stopped	Stopped	Stopped

There are three steps to invoking HALT or STOP mode from NORMAL mode.

1. To return via a maskable interrupt on the PSW, set MIE to 1 and set an IM value that allows the recovery source interrupt to be received.

2. Check that the interrupt request flag (xxxIR) of the maskable interrupt control register (xxxICR) has been cleared, then set the interrupt enable flag (XXXIE) for the recovery source.

3. Set the CPUM to invoke HALT or STOP mode.

To clear an interrupt request flag using software, set the IRWE flag in the memory control register (MEMCTR).

2.12.3 Moving between SLOW and NORMAL Modes

The MN101C46F has two CPU operating modes, NORMAL and SLOW. To move between SLOW and NORMAL modes, you must transit an idle state.

The following is an example of a program that moves between NORMAL and SLOW modes.

Program 1

```
    MOV   x'3',D0    ; Invokes SLOW mode
    MOV   D0, (CPUM)
```

If the slow clock is running with sufficient stability, you can invoke SLOW mode from NORMAL mode with a simple write to the CPU mode control register. This does not require that you transit the idle state.

To switch from SLOW mode back to NORMAL mode, start the fast clock oscillating and hold the program in idle until the clock becomes sufficiently stable. During idle, the CPU runs at the slow clock.

Oscillation stabilization requires the same length of time as a reset. Unlike the reset, however, you do not need to count with a program. We recommend that you discuss the oscillation stabilization time with the oscillator manufacturer.

The following are examples of programs that move from SLOW back to NORMAL mode.

Program 2

```
    MOV   x'01',D0   ; Invokes IDLE mode
    MOV   D0, (CPUM)
```

Program 3

```
    MOV   x'0B',D0   ; Loop for waiting 10-100 µs
LOOP ADD  -1,D0      ; when at 3.58 MHz and
    BNE   LOOP       ; moving from slow (3.58 MHz)
    SUB   D0,D0      ; to fast (14.32 MHz) clocks
    MOV   D0, (CPUM) ; Sets NORMAL mode
```

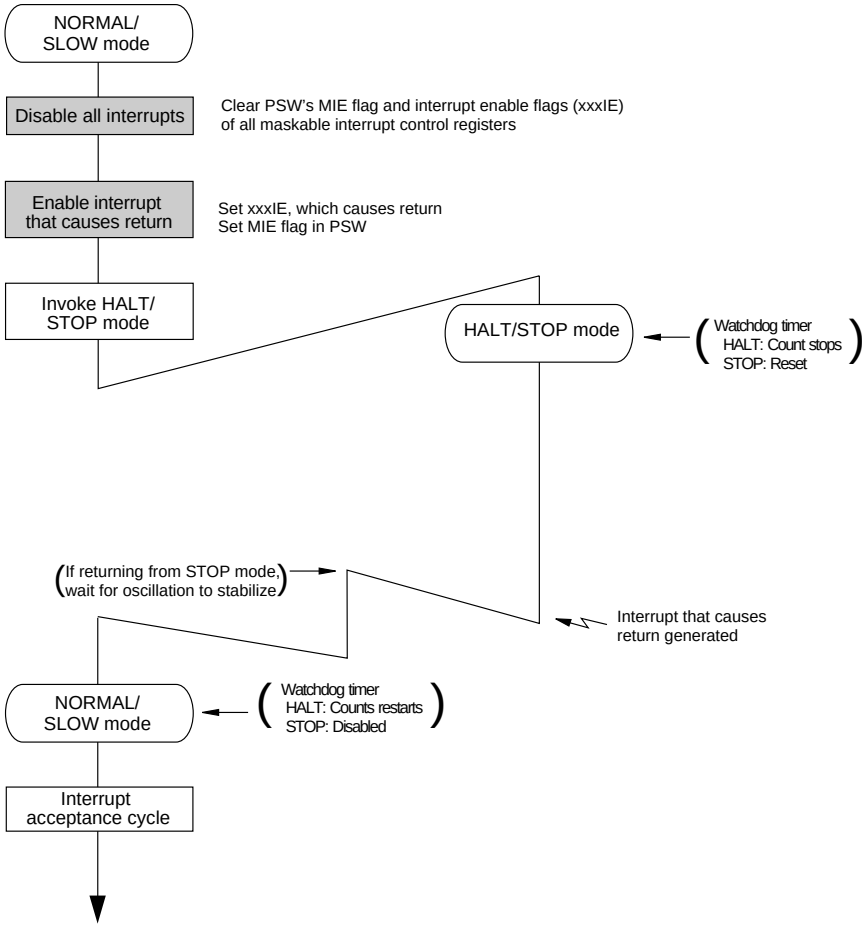
2.12.4 Invoking the Standby Mode

Use programming to invoke standby modes from CPU operating modes; use interrupts to return from standby modes to CPU operating modes.

Set the following to invoke standby modes.

1. Clear the interrupt enable flag (MIE) in the processor status word (PSW) and the interrupt enable flag (xxxIE) in the maskable interrupt control register (xxxICR) to disable all interrupts temporarily.

2. Specify the interrupt source that will return the standby mode to a CPU operating mode and set the xxxIE in the maskable interrupt control register for only that return. Also set the MIE flag in the PSW.



Note: Operations in parentheses are performed by hardware.

Figure 2-7 Sequence for Invoking and Exiting Standby Modes

You cannot return to a CPU operating mode with a maskable interrupt unless the interrupt has been enabled and the level is equal to or higher than the mask level set in the PSW for the priority of the interrupt to be used before invoking HALT or STOP mode.

■ Invoking the HALT Mode

Invoke HALT0 from the NORMAL mode and HALT1 from the SLOW mode. Only the CPU halts; the oscillation state is maintained. Return from HALT mode with an interrupt or reset. A reset will cause an ordinary reset operation; an interrupt will return to the mode prevailing before HALT was invoked. If you invoke HALT mode with the watchdog timer enabled, the watchdog timer will stop counting. It will resume counting once you return to a CPU operating mode.

Program 4

```
MOV   x'4',D0     ; Invokes HALT mode
MOV   D0, (CPUM)
NOP               ; Executes up to 3 instructions
NOP               ; depending on pipeline state
NOP               ; after writing to CPUM.
```

■ Invoking STOP Mode

Invoke STOP0 from NORMAL mode and STOP1 from SLOW mode. Oscillation and the CPU halt for both cases. Return from STOP0 and STOP1 with an interrupt or a reset.

Program 5

```
MOV   x'8',D0    ; Invokes STOP mode
MOV   D0, (CPUM)
NOP              ; Executes up to 3 instructions
NOP              ; depending on pipeline state
NOP              ; after writing to CPUM.
```

Right after the instruction of the transition to HALT, STOP mode, NOP instruction should be inserted 3 times. By these commands, transition to STOP / HALT mode is always executed during NOP instructions, and in programming the state of pipeline need not to be mind.

2.13 Setting the Clock Switch Register

Always set the MN101C46F's CPU mode control register (CPUM) flags and oscillation frequency control register (OSCMD) flags.

OSCMD: Oscillation frequency control register **x'03F0D'**

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	Reserved	Reserved
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R/W	R/W

Reserved: Always set to 0.

CPUM: CPU mode control register **x'03F00'**

Bit:	7	6	5	4	3	2	1	0
	—	OSC SEL1	OSC SEL0	OSC DBL	STOP	HALT	OSC1	OSC0
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

OSCSEL[1:0]: Clock division (NORMAL mode)

00: 1
01: Do not set.
10: Do not set.
11: Do not set.

OSCDBL: Always set to 0.

When switching clocks, set the OSCDBL, OSCSEL, and OSC0 flags separately. Even flags that are mapped to the same special register must be set in two separate operations.

2.14 Resets

2.14.1 Reset Operation

The CPU is internally reset and all registers initialize when the NRST pin (P36) is driven low.

2.14.1.1 Invoking the Reset Mode

There are two ways to invoke the reset mode.

1. Drive the NRST pin low.
   Drive the NRST pin low for at least four cycles of the slow clock (f_X).

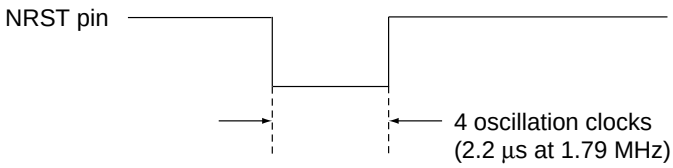


Figure 2-8 Minimum Reset Pulse Width

Use a circuit that provides a pulse with a sufficiently long low level for an instant shut-off if you are connecting a supply voltage lowering circuit to the NRST pin. A reset might also be produced if the oscillation clock has a pulse with a low level time of four clocks or less, so pay attention to noise.

2. You can also invoke the reset mode through programming (a software reset) by outputting low to pin P36 (NRST) by setting the P3OUT6 flag in the P3OUT register to 0. When a reset within the MN101C46F initializes the registers, the P3OUT6 flag is set to 1, and the reset is cleared. See section 4.3, “I/O Port Control Registers,” on page 85.

The MN101C46F starts up in SLOW mode if a slow oscillation is used as the base clock.

2.14.1.2 Operation Sequence During Resets

1. When the reset pin changes from L to H, the internal 14-bit counter (also used as the watchdog timer) starts counting the system clocks. The length of time required from the start of counting to overflow is called the oscillation stabilization wait.
2. Internal registers and special registers are initialized during the reset period.
3. Internal resets are cleared when the oscillation stabilization wait ends and program execution begins for the address written in the vector table at address x’04000’.

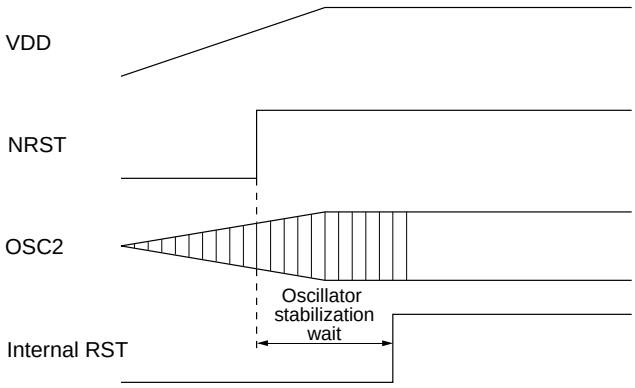


Figure 2-9 Reset Clearing Sequence

2.14.2 Oscillation Stabilization Wait

The oscillation stabilization wait is the time required for a stopped oscillation circuit to reach stable oscillation. An oscillation stabilization wait is inserted automatically when a reset is cleared or when returning from the STOP mode. You can select the oscillation stabilization wait when returning from the STOP mode by setting the oscillation stabilization wait control register (DLYCTR). The oscillation stabilization wait is fixed for reset clearing.

The timer that counts the oscillation stabilization wait is also used as a watchdog timer. It also functions as an override detection timer if not clearing a reset or returning from STOP mode. When resetting from the STOP mode, the watchdog timer is initialized and counting begins from an initial value (x'0000') using the system clock (f_S) as the clock source. After the oscillation stabilization wait ends, the timer continues counting as a watchdog timer. See section 9, "Watchdog Timer," on page 169.

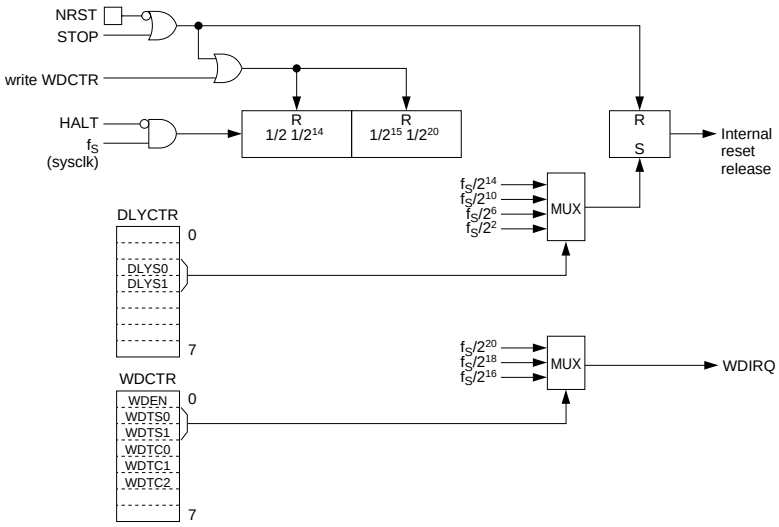


Figure 2-10 Function Block Diagram of the Oscillator Stabilization Wait

DLYCTR: Oscillation Stabilization Wait Control Register **x'03F03'**

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	DLYS1	DLYS0	—	—
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R/W	R/W	R	R

DLYS[1:0]: Oscillation stabilization wait period select

The oscillation stabilization wait period after a reset is $f_S/2^{14}$.

00: 2^{14}
01: 2^{10}
10: 2^6
11: 2^2

2.14.2.1 Controlling the Oscillation Stabilization Wait

When returning from the STOP mode, you can select an oscillation stabilization wait of 2^{14},2^{10}, 2^6, or 2^2 system clocks by setting bits [3:2] (DLYS1 and DLYS0) of DLYCTR.

When clearing a reset, the oscillation stabilization wait is fixed at 2^{14} system clocks. Select the system clocks in the CPU mode control register (CPUM).

Table 2-9 Oscillation Stabilization Wait

DLYS1	DLYS2	Oscillation Stabilization Wait
0	0	2^{14} system clocks
0	1	2^{10} system clocks
1	0	2^6 system clocks
1	1	2^2 system clocks

3 Interrupts

3.1 Description

For faster response, the MN101C46F uses a vector system that jumps directly to an interrupt service routine. The MN101C46F's interrupt vectors include a reset, a nonmaskable interrupt (NMI), six external interrupts, and nine internal interrupts (peripheral function interrupts).

The operation of normal interrupts, other than the reset, follows a sequence consisting of an interrupt request, acceptance of the interrupt, and hardware processing. The hardware processing of interrupts consists of saving the program counter (PC), processor status word (PSW), and handy addressing information (HA) to the stack and jumping to the address specified by the vector. After the interrupt service routine quits, the saved data can be restored using an RTI instruction. After an interrupt occurs, jumping to the interrupt service routine takes a maximum of 12 machine cycles; the return takes a maximum of 11 machine cycles.

Interrupt control registers set up for each interrupt are used to control the interrupt function. Interrupt control registers each have an interrupt request flag (IR), and interrupt enable flag (IE), and an interrupt level flag field (LV[1:0]).

The interrupt request flag (IR) is set to 1 when an event that is an interrupt source occurs. It is cleared to 0 when the interrupt is accepted. The interrupt request flag is hardware operated, but it can also be written to using software.

The interrupt enable flag (IE) enables the specified interrupt. The nonmaskable interrupt (NMI) has no interrupt enable flag; if its interrupt request flag is set, it is accepted unconditionally. Maskable interrupts have interrupt enable flags. The interrupt enable flags of maskable interrupts are valid if the maskable interrupt enable flag (MIE flag) of the PSW is set to 1.

Vector numbers are preset for maskable interrupts in the hardware, but you can give interrupts priorities with a user program by setting the interrupt level flag field (LV[1:0]). There are three levels of interrupt priorities; when multiple interrupts have the same priority level, the one with the lowest vector number has the highest priority. A maskable interrupt is accepted if the level set for it in the interrupt level field (LV[1:0]) is higher than the level in the interrupt mask level field (IM[1:0]) of the PSW. No level is specified for nonmaskable interrupts; they are always accepted with top priority.

3.2 Functions

Table 3-1 Interrupt Functions

Interrupt Types	Reset Interrupt	Nonmaskable Interrupt	Maskable Interrupts
Vector No.	0	1	2–27
Table address	x'04000'	x'04004'	x'04008'–x'0406C'
Start address	Specify with the vector table.		
Interrupt level	—	—	Levels 0–2 (programmed)
Interrupt source	External RST pin input	Software fault detection, PI interrupts	Interrupts from external pin input or internal peripheral functions
Interrupt generation	Input directly to CPU core.	Input to CPU core from the nonmaskable interrupt control register (NMICR).	The interrupt request level set in the interrupt level flag (xxxLVn) of the maskable interrupt control register (xxxICR) is input to the CPU core.
Acceptance	Always accepted.	Always accepted.	Acceptance determined by the interrupt mask level (IM) of the PSW and interrupt control of the register (xxxICR).
Machine cycles needed for acceptance	12	12	12
PSW status after acceptance	All flags cleared to 0.	Interrupt mask level flag of PSW cleared to 00.	The interrupt mask level of the PSW is set to the interrupt level flag (xxxLVn) setting. (Interrupt requests whose level is the same or lower than the accepted level are masked.)

3.3 Block Diagram

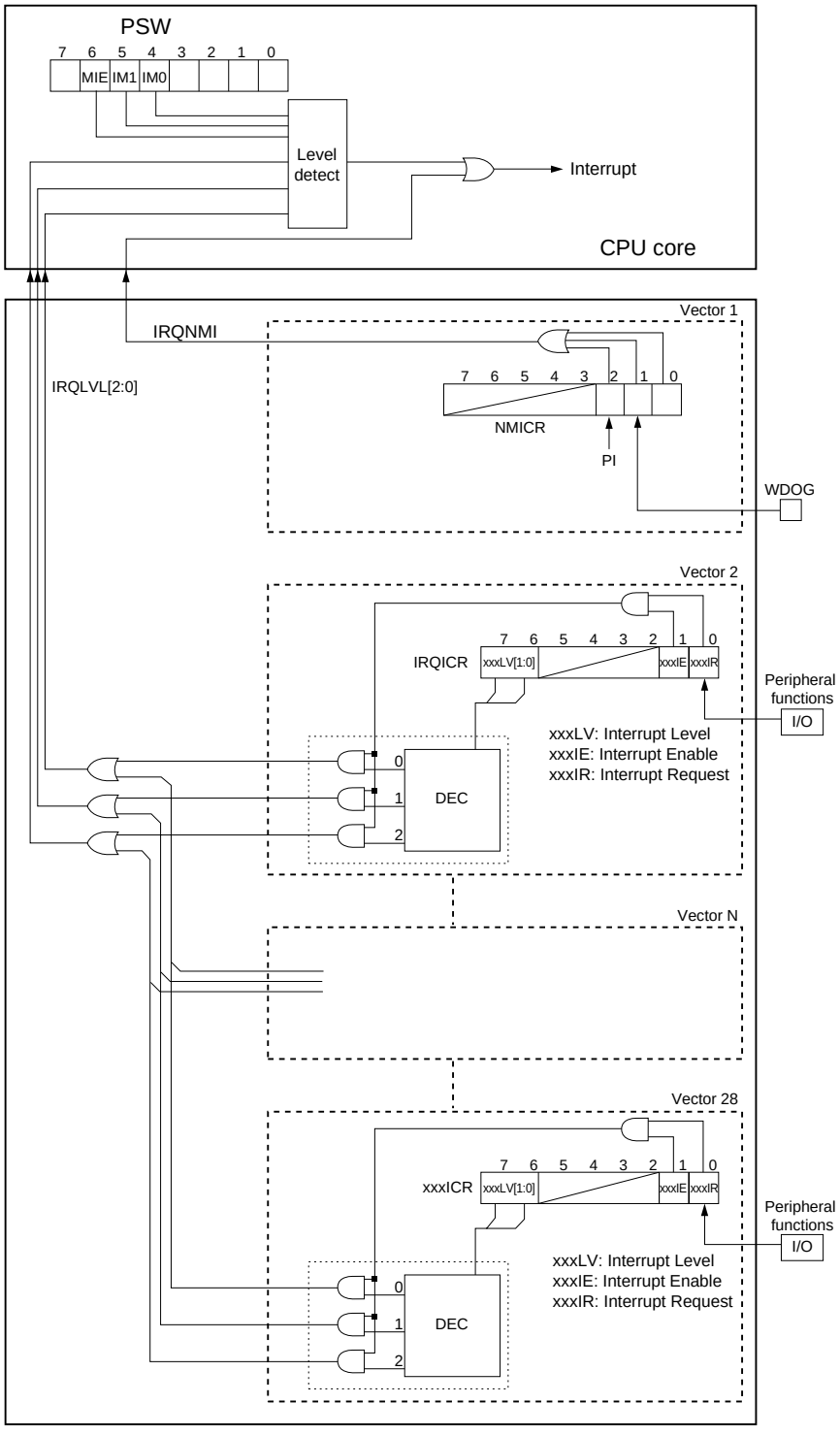


Figure 3-1 Block Diagram for Interrupt Functions

3.4 Operation

3.4.1 Interrupt Handling Sequence

The operation of normal interrupts (other than the reset) follows a sequence consisting of an interrupt request, acceptance of the interrupt, and hardware processing. The hardware processing of interrupts consists of saving the program counter (PC), processor status word (PSW), and handy addressing information (HA) to the stack and jumping to the address specified by the vector. After the interrupt service routine quits, the register values saved when the interrupt was accepted are restored using an RTI instruction and operation returns to the program that was running when that interrupt was accepted.

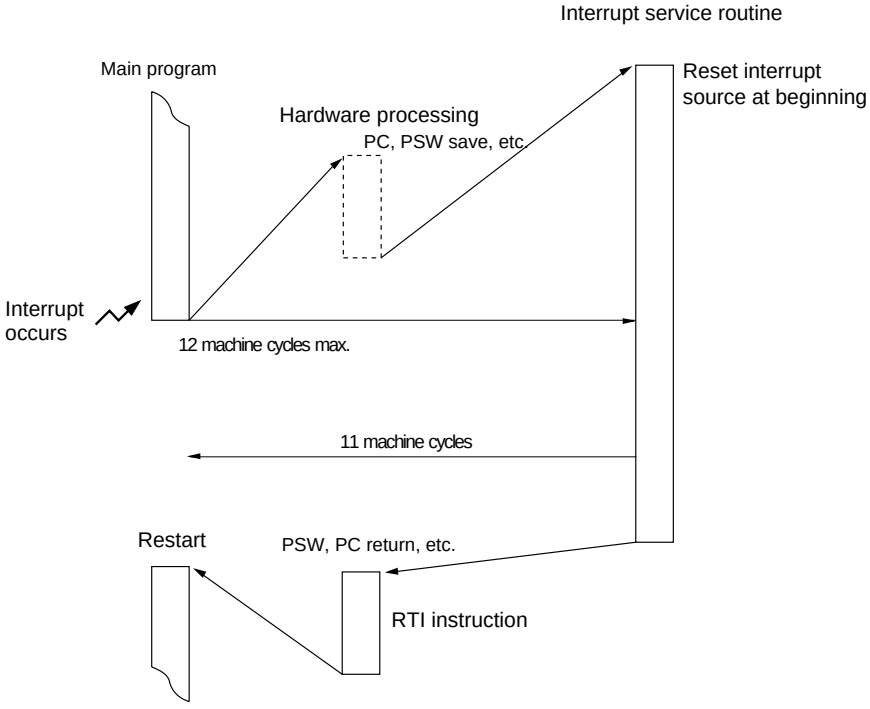


Figure 3-2 Interrupt Handling Sequence (Maskable Interrupts)

3.4.2 Interrupt Vector Addresses and Interrupt Groups

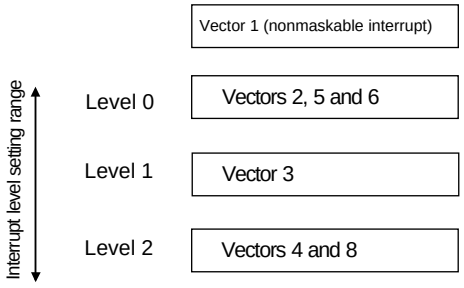
The table below shows the relationship between interrupt vector addresses and interrupt groups.

Table 3-2 Interrupt Vector Addresses and Interrupt Groups

Vector No.	Address	Interrupt Group (Source)		Control Register (Address)	
0	x'04000'	Reset	—	—	—
1	x'04004'	Nonmaskable interrupt	NMI	NMICR	x'03FE1'
2	x'04008'	External interrupt 0	IRQ0	IRQ0ICR	x'03FE2'
3	x'0400C'	External interrupt 1	IRQ1	IRQ1ICR	x'03FE3'
4	x'04010'	External interrupt 2	IRQ2	IRQ2ICR	x'03FE4'
5	x'04014'	External interrupt 3	IRQ3	IRQ3ICR	x'03FE5'
6	x'04018'	External interrupt 4	IRQ4	IRQ4ICR	x'03FE6'
7	x'0401C'	External interrupt 5	IRQ5	IRQ5ICR	x'03FE7'
8	x'04020'	Reserved	—	—	—
9	x'04024'	Reserved	—	—	—
10	x'04028'	Reserved	—	—	—
11	x'0402C'	Timer 2 interrupt	TM2IRQ	TM2ICR	x'03FEB'
12	x'04030'	Timer 3 interrupt	TM3IRQ	TM3ICR	x'03FEC'
13	x'04034'	Timer 4 interrupt	TM4IRQ	TM4ICR	x'03FED'
14	x'04038'	Reserved	—	—	—
15	x'0403C'	Reserved	—	—	—
16	x'04040'	Reserved	—	—	—
17	x'04044'	Reserved	—	—	—
18	x'04048'	Reserved	—	—	—
19	x'0404C'	Closed-caption decoder 0 interrupt	VBI0IRQ	VBI0ICR	x'03FF3'
20	x'04050'	Closed-caption decoder 1 interrupt	VBI1IRQ	VBI1ICR	x'03FF4'
21	x'04054'	OSD interrupt	OSDIRQ	OSDICR	x'03FF5'
22	x'04058'	I^2C interrupt	I2CIRQ	I2CICR	x'03FF6'
23	x'0405C'	Closed-caption decoder 0 VSYNC (VBIV0) interrupt	VBIV0IRQ	VBIV0ICR	x'03FF7'
24	x'04060'	Closed-caption decoder 1 VSYNC (VBIV1) interrupt	VBIV1IRQ	VBIV1ICR	x'03FF8'
25	x'04064'	Reserved	—	—	—
26	x'04068'	A/D conversion interrupt	ADIRQ	ADICR	x'03FFA'
27	x'0406C'	Remote control interrupt	RMIRQ	RMICR	x'03FFB'
28	x'04070'	Reserved	—	—	—
29	x'04074'	Reserved	—	—	—
30	x'04078'	Reserved	—	—	—

3.4.3 Interrupt Levels and Priorities

The MN101C46F assigns vector numbers and interrupt control registers for each interrupt (except reset interrupts). It sets interrupt levels by interrupt groups (except for the reset and nonmaskable interrupts) using programming. There are three levels of interrupt priority; when multiple interrupts have the same priority level, the one with the lowest vector number has the highest priority. (When a vector 3 interrupt set at level 1 occurs at the same time as a vector 4 interrupt set at level 2, the vector 3 interrupt is accepted.)



Priority	Interrupt vector No.
1	Vector 1
2	Vector 2
3	Vector 5
4	Vector 6
5	Vector 3
6	Vector 4
7	Vector 8

Figure 3-3 Example of Interrupt Levels

3.4.4 How Interrupts Are Accepted

After an interrupt source occurs, it goes through the following sequence until it is accepted.

1. The interrupt request flags (xxxIR) of the external interrupt control register (IRQnICR) and internal interrupt control register (xxxICR) corresponding to the interrupt source are set to 1.

2. If the interrupt enable flag (xxxIE) for that interrupt request flag is 1, an interrupt request signal is output to the CPU.

3. The interrupt request signal is the interrupt level information set for the individual interrupt. The level set in the interrupt level flag (xxxLV[1:0]) is output to the CPU.

4. If the level of the interrupt request of the output interrupt request signal is higher than the level set in the interrupt mask level field (IM[1:0]) of the PSW and the PSW's interrupt enable flag (MIE) is 1 (enabled), the interrupt is accepted.

5. After the interrupt is accepted, the interrupt request flag (xxxIR) is cleared.

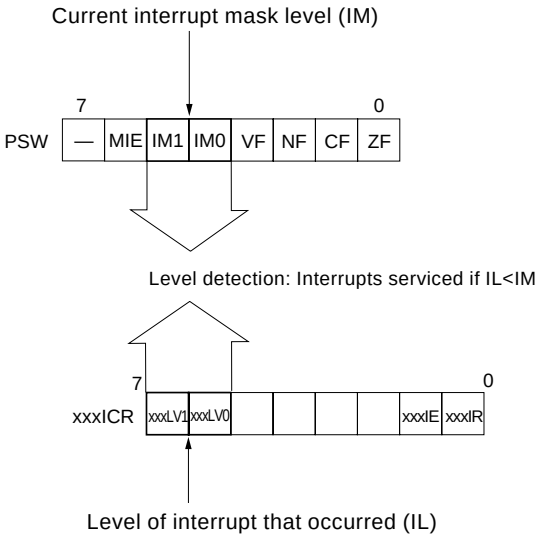


Figure 3-4 Interrupt Acceptance Process

The mask interrupt enable flag (MIE) is set to 0 in the following cases, disabling interrupts.

- When the program writes a 0 to MIE in the PSW.
- When a reset is input.

The MIE flag is set to 1 in the following cases, enabling interrupts.

- When a program writes a 1 to the MIE in the PSW.

The value in the interrupt mask level field (IM[1:0]) is changed in the following cases.

- When new values are written by a program in the IM[1:0] field of the PSW.
- IM[1:0] become 00 when a reset is input.
- IM[1:0] change to the corresponding interrupt level value when a maskable interrupt is accepted.
- When an RTI instruction is executed at the very end of the interrupt service routine program, IM[1:0] is restored to the mask level value prevailing before the interrupt was accepted.

The interrupt enable flag (xxxIE) is not cleared after the interrupt is accepted.

The MIE in the PSW is not cleared to 0 when an interrupt is accepted.

If a nonmaskable interrupt and a maskable interrupt occur at the same time, the nonmaskable interrupt takes priority.

3.4.5 Interrupt Acceptance Operation

The MN101C46F uses hardware to save the program's return address, PSW, and the like to the stack when an interrupt is accepted. It then jumps to the start address of the interrupt program specified in the interrupt vector table. The following shows the sequence of hardware processing when an interrupt is accepted.

1. The stack pointer (SP) value is updated.
   (SP-6) → (SP)

2. The handy address register (HA) is saved to the stack.
   High order HA → (SP+5)
   Low order HA → (SP+4)

3. Program counter data (PC = top return address) is saved to the stack.
   PC (bits 18:17, 0) → (SP+3)
   PC (bits 16:9) → (SP+2)
   PC (bits 8:1) → (SP+1)

4. The PSW is saved to the stack.
   PSW → (SP)

5. The xxxLVn of the accepted interrupt is copied to the IMn of the PSW.
   Interrupt level (xxxLVn) → IMn

6. Operation jumps to vector table address.

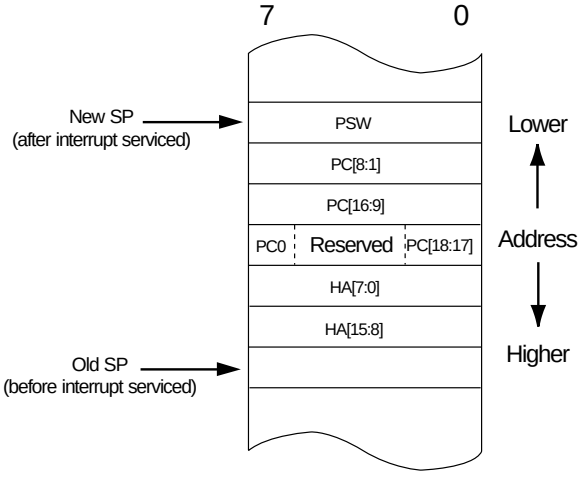


Figure 3-5 Stack Status during Interrupts

3.4.6 Interrupt Return Operation

After the values of registers and the like that were saved by interrupt servicing have been restored by the program (by a POP instruction), operation returns to the program that was being executed when the interrupt was accepted via an RTI instruction.

The following shows the sequence of operation for the return-from-interrupt instruction (RTI)

1. The contents of the PSW saved to the stack (SP) are restored.
2. The data of the program counter (PC=Top return address) saved to the stack (SP+1, 2 or 3) is restored.
3. The handy address register (HA) is restored from the stack (SP+4 or 5)
4. The SP value is updated. (SP+6) → (SP)
5. Operation jumps to the address indicated by the PC.

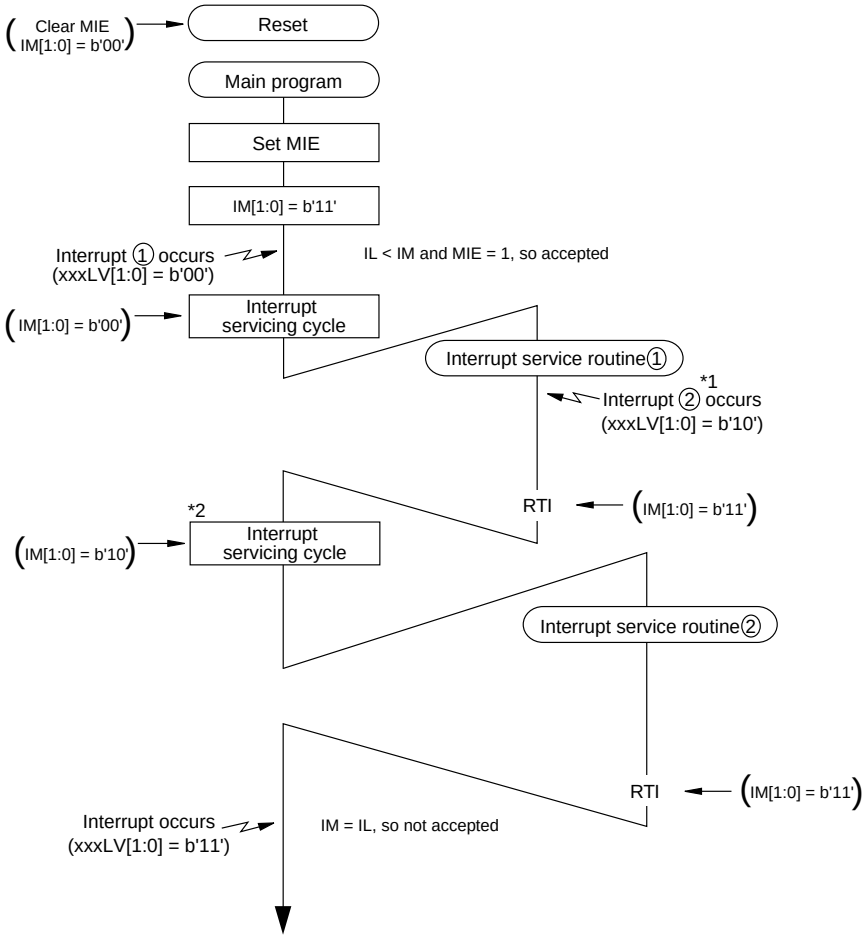
The handy address register is an internal register saved so that handy addressing is not affected by interrupts.

Data and address registers are not saved, so put a PUSH instruction in the program if you need to save them to the stack.

Bits 6:2 of the address where the program counter (bits 18:17 and 0) is saved are reserved. Do not have the program change them.

3.4.7 Maskable Interrupts

The sequence of operation when an interrupt with a low mask level occurs during interrupt servicing is shown below (interrupt 1: xxxLV[1:0]="00", interrupt 2: xxxLV[1:0]="10").



Notes:
1. Interrupts that occur during interrupt service routine 1 are accepted as nested interrupts if IL < IM. If IL> IM, they are not accepted.
2. If interrupt 2, which occurs during interrupt service routine 1, is not accepted because IL ≥IM, it is accepted when interrupt service routine 1 ends.
3. Operations in parentheses are performed by hardware.

Figure 3-6 Processing Sequence for Maskable Interrupts

3.4.8 Nested Interrupts

Once an interrupt is received, the MN101C46F automatically disables acceptance of interrupts with lower levels. When an interrupt is received, its xxxLV[1:0] is copied to the processor status word's IM[1:0]. Thus, after an interrupt is received, interrupts with levels lower than that of the received interrupt are automatically disabled, but interrupts with levels higher than the received interrupt are accepted as nested interrupts. In general, priority is assigned by levels even during interrupt handling; however, you can still control nested interrupts through the following procedures.

1. To disable nested interrupts, do either of the following:
   - ♦ Clear MIE in the PSW to 0.
   - ♦ Rewrite IM[1:0] in the PSW to raise the mask level.
2. To enable interrupts with levels lower than the received interrupt:
   - ♦ Rewrite IM[1:0] in the PSW to lower the mask level.

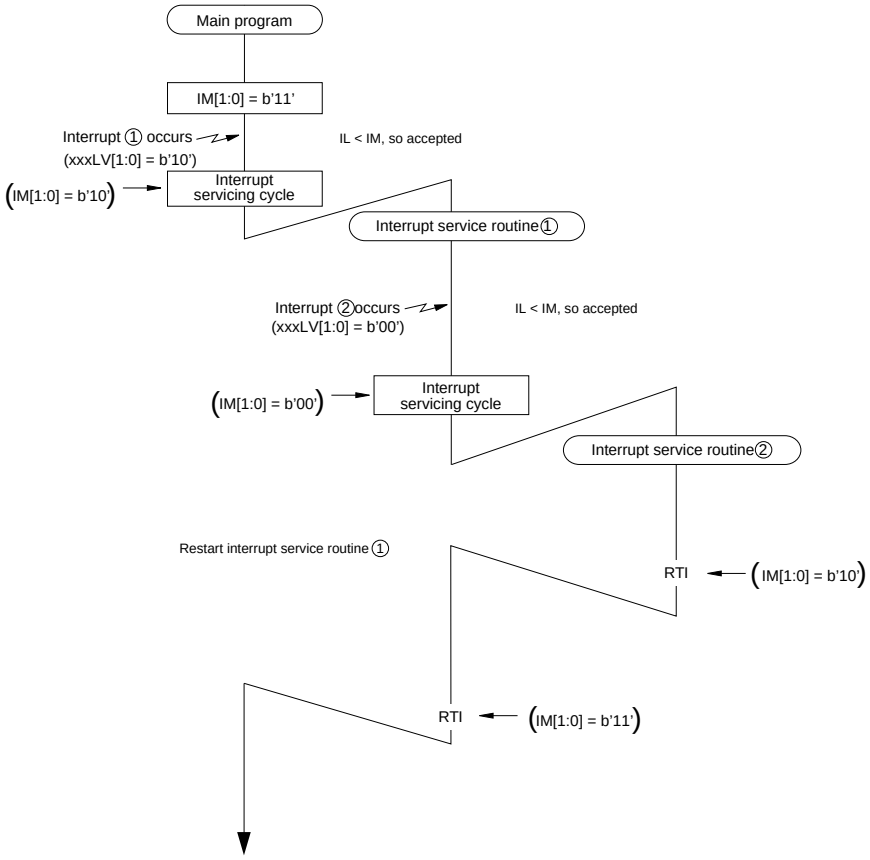
Interrupt nesting can only be enabled for those interrupts whose level is higher than the interrupt mask level (IM) of the PSW.

You can accept an interrupt with priority below that of the interrupt currently being processed by forcibly re-writing its IM, but be sure the nested interrupts do not overflow the stack.

Do not perform operations in the maskable interrupt control register (xxxICR) when interrupt nesting is enabled. If you need to do an operation, first clear the MIE flag of the PSW to disable interrupts.

The operating sequence for nested interrupts is shown below.

(Interrupt 1: xxxLV[1:0]= 10, Interrupt 2: xxxLV[1:0]= 00)



Note: Operations in parentheses are performed by hardware.

Figure 3-7 Processing Sequence for Nested Interrupts

3.5 Setting the Interrupt Flags

3.5.1 Using Software to Rewrite Interrupt Request Flags (IR)

Interrupt request flags are operated by hardware and are set to 1 when an interrupt source event occurs. When the interrupt is accepted, they are cleared to 0. To rewrite an interrupt request flag by software, you must set the IRWE flag in the MEMCTR register.

3.5.2 Setting the Interrupt Flags

The table below shows how interrupt flags are set and describes the flags (including changing interrupt request flags using software).

Table 3-3 Setting Interrupt Flags

Procedure	Description
Disable all maskable interrupts	Clear the MIE flag in the PSW to disable all maskable interrupts. Be sure to do this if you are changing the interrupt control register.
Set up interrupt sources	This procedure makes selections for interrupt sources such as choosing the interrupt edges, changing the interrupt periods of timers, and the like.
Write-enable interrupt request flags	This operation sets the IRWE flag of the memory control register (MEMCTR) and enables writes to interrupt request flags using software. You only need to do this if you change the interrupt request flag with software.
Rewrite the interrupt request flags	This operation rewrites interrupt request flags (xxxIR) in the interrupt control register (xxxICR).
Write-disable interrupt request flags	This operation clears the IRWE flag and disables writing to the interrupt request flags using software.
Set the interrupt levels	This operation sets interrupt levels using the xxxLV[1:0] flag of the interrupt control register (xxxICR). Set the IM[1:0] flag of the PSW if you need to change the interrupt acceptance level of the CPU.
Enable interrupts	This operation sets the xxxIE flag of the interrupt control register (xxxICR) to enable interrupts.
Enable all maskable interrupts	This operation sets the MIE flag of the PSW and enables maskable interrupts.

3.6 Interrupt Control Registers

The interrupt control registers are comprised of the nonmaskable interrupt control register (NMICR), external interrupt control registers (IRQnICR), and internal interrupt control registers (xxxICR).

Table 3-4 Interrupt Control Registers

Register	Address	R/W	Register name
NMICR	x'03FE1'	R/W	Nonmaskable interrupt control register
IRQ0ICR	x'03FE2'	R/W	External interrupt 0 control register
IRQ1ICR	x'03FE3'	R/W	External interrupt 1 control register
IRQ2ICR	x'03FE4'	R/W	External interrupt 2 control register
IRQ3ICR	x'03FE5'	R/W	External interrupt 3 control register
IRQ4ICR	x'03FE6'	R/W	External interrupt 4 control register
IRQ5ICR	x'03FE7'	R/W	External interrupt 5 control register
TM2ICR	x'03FEB'	R/W	Timer 2 interrupt control register (timer 2 compare match)
TM3ICR	x'03FEC'	R/W	Timer 3 interrupt control register (timer 3 compare match)
TM4ICR	x'03FED'	R/W	Timer 4 interrupt control register (timer 4 compare match)
VBI0ICR	x'03FF3'	R/W	Closed-caption decoder 0 interrupt control register
VBI1ICR	x'03FF4'	R/W	Closed-caption decoder 1 interrupt control register
OSDICR	x'03FF5'	R/W	OSD interrupt control register
I2CICR	x'03FF6'	R/W	I^2C interrupt control register
VBIV0ICR	x'03FF7'	R/W	Closed-caption decoder 0 VSYNC (VBIV0) interrupt control register
VBIV1ICR	x'03FF8'	R/W	Closed-caption decoder 1 VSYNC (VBIV1) interrupt control register
ADICR	x'03FFA'	R/W	A/D conversion interrupt control register
RMCICR	x'03FFB'	R/W	Remote control interrupt control register

Disable all maskable interrupts with the MIE flag of the PSW register before writing to the interrupt control register.

Setting the interrupt level specification flag (xxxLVn) to level 3 disables that vector's interrupts regardless of the interrupt enable flag and interrupt request flag.

NMICR: Nonmaskable Interrupt Control Register **x'03EF1'**

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	PIR	WDIR	Reserved
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R/W	R/W	R/W

PIR: Program interrupt request flag
- 0: No interrupt request
- 1: Generate interrupt request

WDIR: Watchdog interrupt request flag
- 0: No interrupt request
- 1: Generate interrupt request

Reserved: Always set to 0.

The nonmaskable interrupt control register stores nonmaskable interrupt requests. When a nonmaskable interrupt occurs, it is accepted regardless of the interrupt mask level (IMn) in the PSW, and operation jumps to the address written at x'04004' in the interrupt vector table. The watchdog timer overflow interrupt request flag (WDIR) is set to 1 when the watchdog timer overflows. The program interrupt request flag (PIR) is set to 1 when an undefined instruction is executed. Forcibly generate a nonmaskable interrupt by setting either the PIR flag or the WDIR flag with an instruction.

IRQ0ICR: External Interrupt 0 Control Register **x'03FE2' to x'03FE71'**

Bit:	7	6	5	4	3	2	1	0
	IRQn LV1	IRQn LV0	REDGn	—	—	—	IRQnIE	IRQnIR
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R	R	R	R/W	R/W

The external interrupt n control register (IRQnICR) is the register that controls interrupt levels, valid edges, interrupt enables, and interrupt requests for external interrupt n. Use interrupt control registers with the maskable interrupt enable flag (MIE) of the PSW at 0. n = 0 to 5.

IRQnLV[1:0]: External interrupt level specification flag

Sets a CPU level between 0–3 for the interrupt.

REDGn: External interrupt enable edge specification flag
- 0: Negative edge
- 1: Positive edge

IRQnIE: External interrupt enable flag
- 0: Disable interrupt
- 1: Enable interrupt

IRQnIR: External interrupt request flag
- 0: No interrupt request
- 1: Generate interrupt request

TMnICR: Timer n Interrupt Control Register **x'03FEB' to x'03FED'**

Bit:	7	6	5	4	3	2	1	0
	TMn LV1	TMn LV0	—	—	—	—	TMnIE	TMnIR
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R	R	R	R	R/W	R/W

The timer n interrupt control register (TMnICR) is the register that controls interrupt levels, valid edges, interrupt enables, and interrupt requests for timer n interrupts. Use interrupt control registers with the maskable interrupt enable flag (MIE) of the PSW at 0. n = 2 to 4.

TMnLV[1:0]: Interrupt level specification flag

A 2-bit flag that sets the interrupt level. Determines which of the CPU's levels 0–3 is assigned to the interrupt.

TMnIE: Interrupt enable flag

0: Disable interrupt
1: Enable interrupt

TMnIR: Interrupt request flag

0: No interrupt request
1: Generate interrupt request

VBI0ICR: Closed-Caption Decoder 0 Interrupt Control Register **x'03FF3**'

Bit:	7	6	5	4	3	2	1	0
	VB0LV1	VB0LV0	—	—	—	—	VB0IE	VB0IR
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R	R	R	R	R/W	R/W

The closed-caption decoder 0 interrupt control register (VBI0ICR) controls interrupt levels, interrupt enables, and interrupt requests for closed-caption decoder 0 interrupts. Use interrupt control registers with the maskable interrupt enable flag (MIE) of the PSW at 0.

VB0LV[1:0]: Interrupt level specification flag

A 2-bit flag that sets the interrupt level. Determines which of the CPU's levels 0–3 is assigned to the interrupt.

VB0IE: Interrupt enable flag

0: Disable interrupt
1: Enable interrupt

VB0IR: Interrupt request flag

0: No interrupt request
1: Generate interrupt request

VBI1ICR: Closed-Caption Decoder 1 Interrupt Control Register **x'03FF4'**

Bit:	7	6	5	4	3	2	1	0
	VB1LV1	VB1LV0	—	—	—	—	VB1IE	VB1IR
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R	R	R	R	R/W	R/W

The closed-caption decdoer 1 interrupt control register (VBI1ICR) controls interrupt levels, interrupt enables, and interrupt requests for closed-caption decoder 1 interrupts. Use interrupt control registers with the maskable interrupt enable flag (MIE) of the PSW at 0.

VB1LV[1:0]: Interrupt level specification flag

A 2-bit flag that sets the interrupt level. Determines which of the CPU's levels 0–3 is assigned to the interrupt.

VB1IE: Interrupt enable flag

0: Disable interrupt
1: Enable interrupt

VB1IR: Interrupt request flag

0: No interrupt request
1: Generate interrupt request

VBIV0ICR: Closed-Caption Decoder 0 VSYNC Interrupt Control Register **x'03FF7**'

Bit:	7	6	5	4	3	2	1	0
	VBV0 LV1	VBV0 LV0	—	—	—	—	VBV0IE	VBV0IR
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R	R	R	R	R/W	R/W

The closed-caption decoder 0 interrupt control register (VBIV0ICR) controls interrupt levels, interrupt enables, and interrupt requests for VBIV0 interrupts. Use interrupt control registers with the maskable interrupt enable flag (MIE) of the PSW at 0.

VBV0LV[1:0]: Interrupt level specification flag

A 2-bit flag that sets the interrupt level. Determines which of the CPU's levels 0–3 is assigned to the interrupt.

VBV0IE: Interrupt enable flag

0: Disable interrupt
1: Enable interrupt

VBV0IR: Interrupt request flag

0: No interrupt request
1: Generate interrupt request

VBIV1ICR: Closed-Caption Decoder 1 VSYNC Interrupt Control Register **x'03FF8'**

Bit:	7	6	5	4	3	2	1	0
	VBV1 LV1	VBV1 LV0	—	—	—	—	VBV1IE	VBV1IR
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R	R	R	R	R/W	R/W

The closed-caption decoder interrupt control register (VBIV1ICR) controls interrupt levels, interrupt enables, and interrupt requests for VBIV1 interrupts. Use interrupt control registers with the maskable interrupt enable flag (MIE) of the PSW at 0.

VBV1LV[1:0]: Interrupt level specification flag

A 2-bit flag that sets the interrupt level. Determines which of the CPU's levels 0–3 is assigned to the interrupt.

VBV1IE: Interrupt enable flag
0: Disable interrupt
1: Enable interrupt

VBV1IR: Interrupt request flag
0: No interrupt request
1: Generate interrupt request

OSDICR: OSD Interrupt Control Register **x'03FF5'**

Bit:	7	6	5	4	3	2	1	0
	OSDLV1	OSDLV0	—	—	—	—	OSDIE	OSDIR
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R	R	R	R	R/W	R/W

The OSD interrupt control register (OSDICR) is the register that controls interrupt levels, interrupt enables, and interrupt requests for OSD interrupts. Use interrupt control registers with the maskable interrupt enable flag (MIE) of the PSW at 0.

OSDLV[1:0]: Interrupt level specification flag

A 2-bit flag that sets the interrupt level. Determines which of the CPU's levels 0–3 is assigned to the interrupt.

OSDIE: Interrupt enable flag
0: Disable interrupt
1: Enable interrupt

OSDIR: Interrupt request flag
0: No interrupt request
1: Generate interrupt request

I2CICR: I^{2}C Interrupt Control Register **x'03FF6'**

Bit:	7	6	5	4	3	2	1	0
	I2CLV1	I2CLV0	—	—	—	—	I2CIE	I2CIR
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R	R	R	R	R/W	R/W

The I^{2}C interrupt control register (I2CICR) is the register that controls interrupt levels, interrupt enables, and interrupt requests for I^{2}C interrupts. Use interrupt control registers with the maskable interrupt enable flag (MIE) of the PSW at 0.

I2CLV[1:0]: Interrupt level specification flag

A 2-bit flag that sets the interrupt level. Determines which of the CPU's levels 0–3 is assigned to the interrupt.

I2CIE: Interrupt enable flag

0: Disable interrupt
1: Enable interrupt

I2CIR: Interrupt request flag

0: No interrupt request
1: Generate interrupt request
<NEW>

ADICR: A/D Conversion Interrupt Control Register **x'03FFA'**

Bit:	7	6	5	4	3	2	1	0
	ADLV1	ADLV0	—	—	—	—	ADIE	ADIR
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R	R	R	R	R/W	R/W

The A/D conversion interrupt control register (ADICR) is the register that controls interrupt levels, interrupt enables, and interrupt requests for A/D conversion interrupts. Use interrupt control registers with the maskable interrupt enable flag (MIE) of the PSW at 0.

ADLV[1:0]: Interrupt level specification flag

A 2-bit flag that sets the interrupt level. Determines which of the CPU's levels 0–3 is assigned to the interrupt.

ADIE: Interrupt enable flag

0: Disable interrupt
1: Enable interrupt

ADIR: Interrupt request flag

0: No interrupt request
1: Generate interrupt request

RMCICR: Remote Control Interrupt Control Register **x'03FFB'**

Bit:	7	6	5	4	3	2	1	0
	RMC LV1	RMC LV0	—	—	—	—	RMCIE	RMCIR
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R	R	R	R	R/W	R/W

The remote control interrupt control register (RMCICR) is the register that controls interrupt levels, interrupt enables, and interrupt requests for remote control interrupts. Use interrupt control registers with the maskable interrupt enable flag (MIE) of the PSW at 0.

RMCLV[1:0]: Interrupt level specification flag

A 2-bit flag that sets the interrupt level. Determines which of the CPU's levels 0–3 is assigned to the interrupt.

RMCIE: Interrupt enable flag

0: Disable interrupt
1: Enable interrupt

RMCDIR: Interrupt request flag

0: No interrupt request
1: Generate interrupt request

4 I/O Ports

4.1 Description

The MN101C46F contains 35 pins that form general-purpose I/O ports. Ports 0, 1, 2, and 3 are 8-bit ports and port 4 is a 3-bit port. All of these pins have alternate functions.

Table 4-1 I/O Port Pins

Port	Associated Pins
Port 0	P07–P00
Port 1	P17–P10
Port 2	P27–P20
Port 3	P37–P30
Port 4	P42–P40

4.2 I/O Port Circuit Diagrams

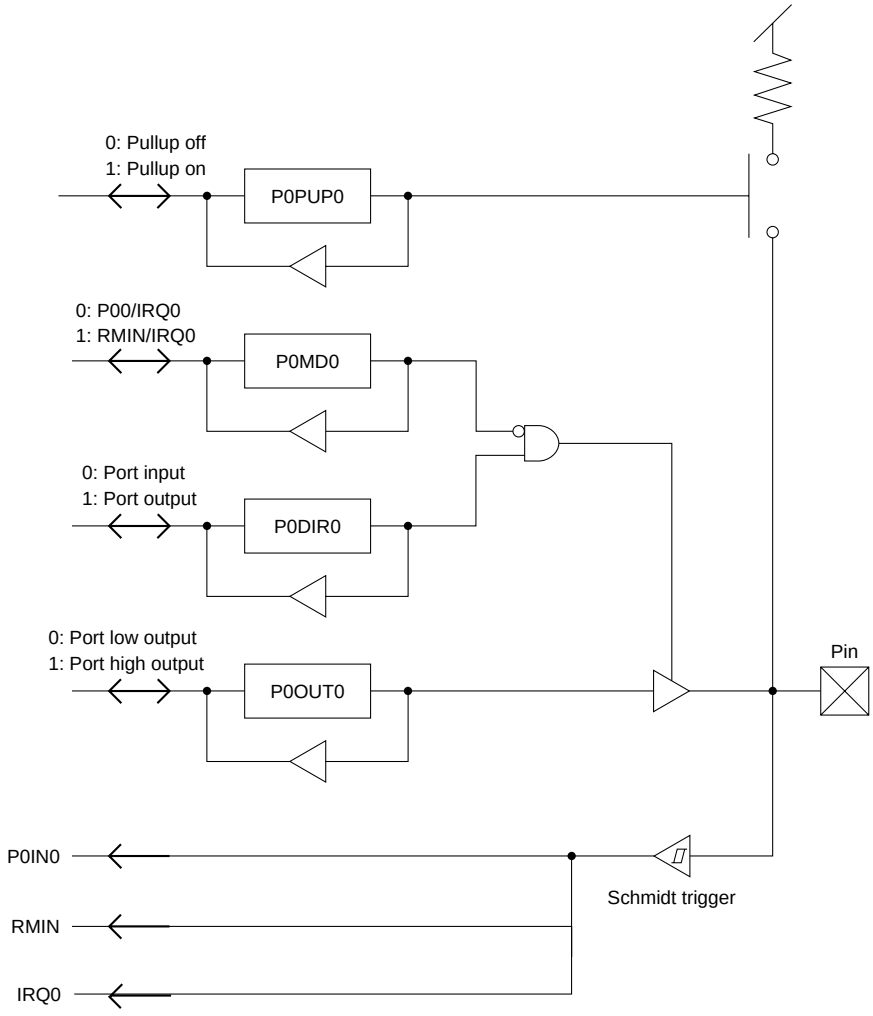


Figure 4-1 P00/RMIN/IRQ0 (Port 0)

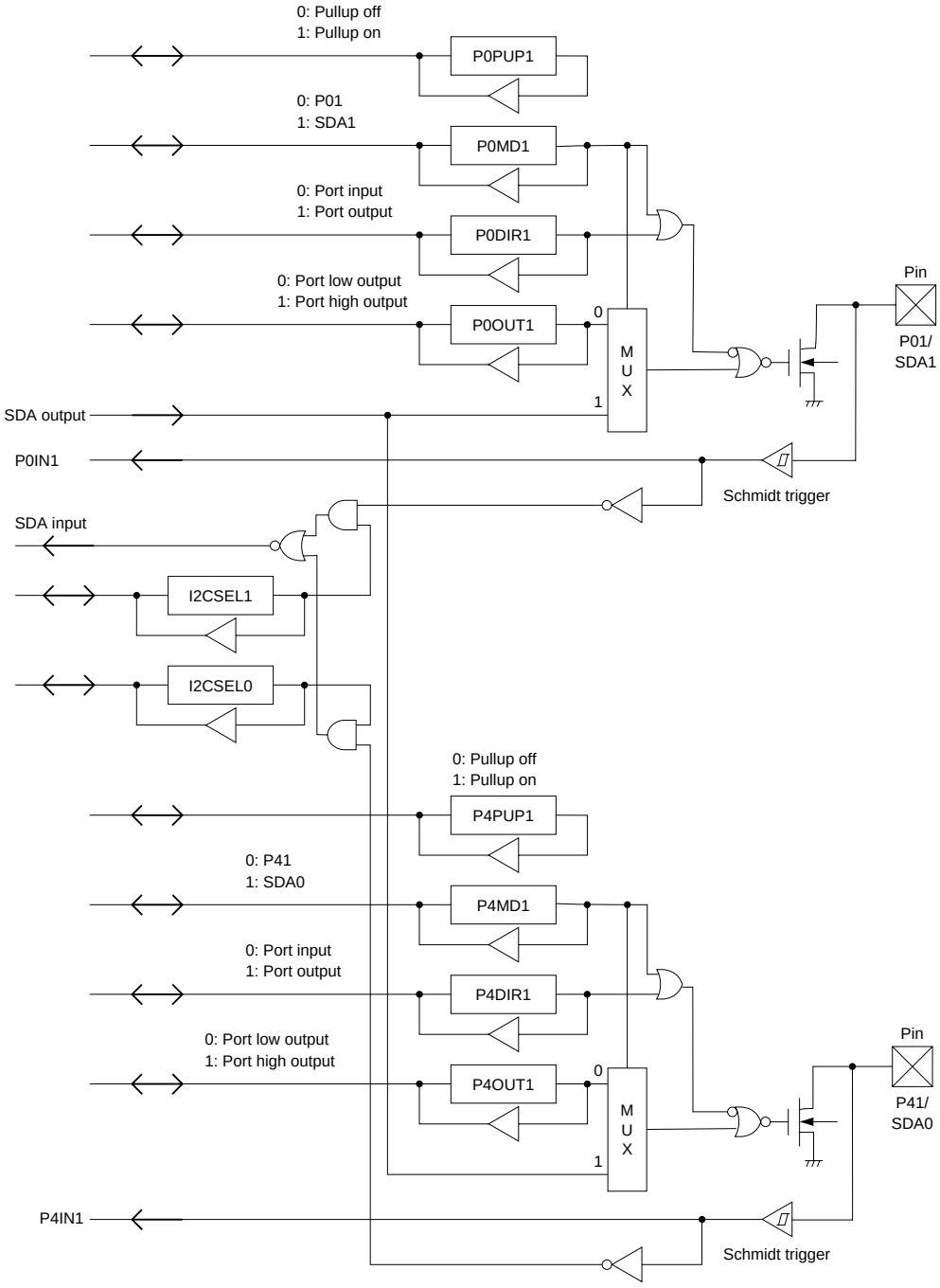


Figure 4-2 P01/SDA1 and P41/SDA0 (Dual-Use I^2C Pins)

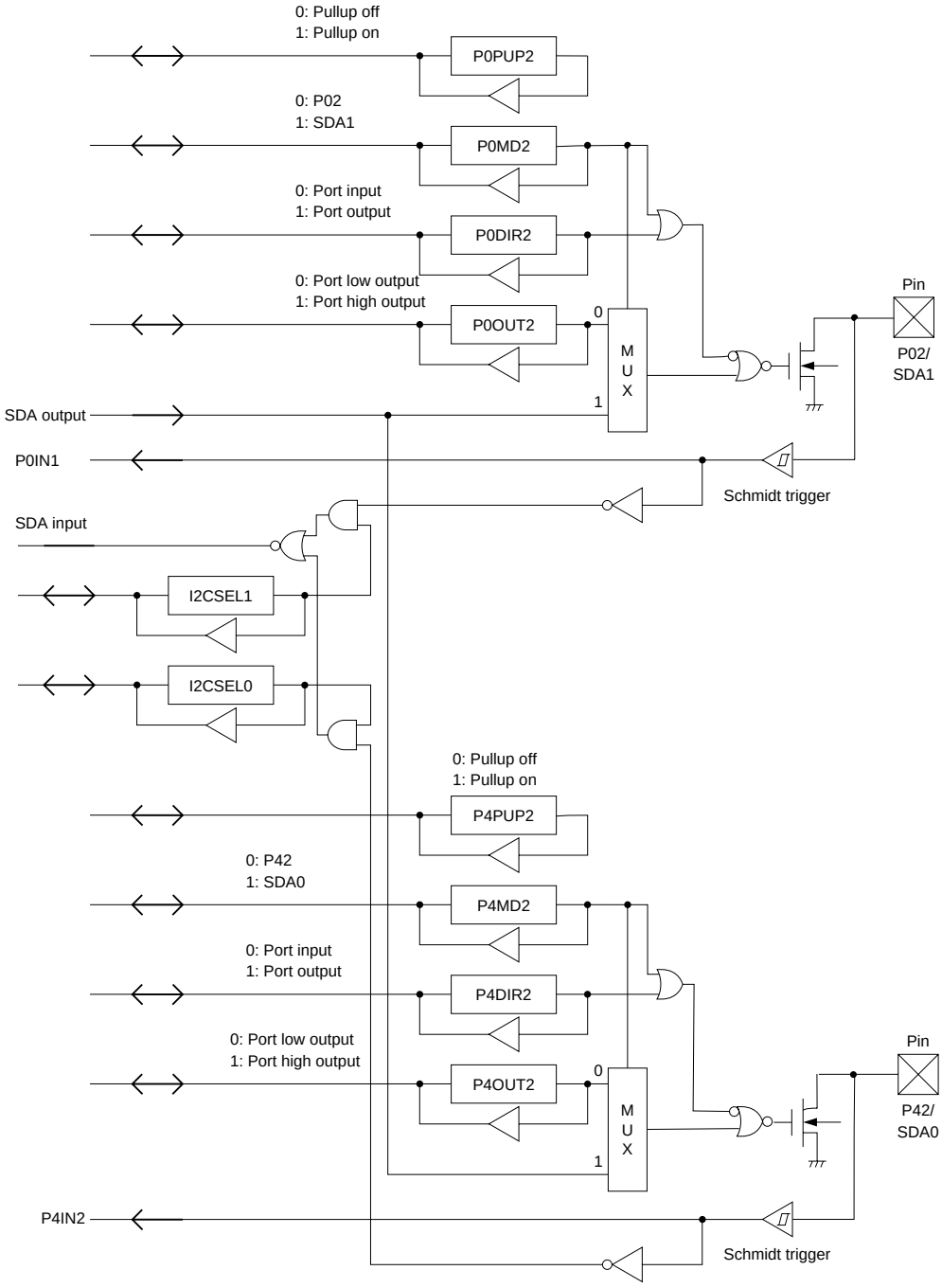


Figure 4-3 P02/SCL1 and P42/SCL0 (Dual-Use I^2C Pins)

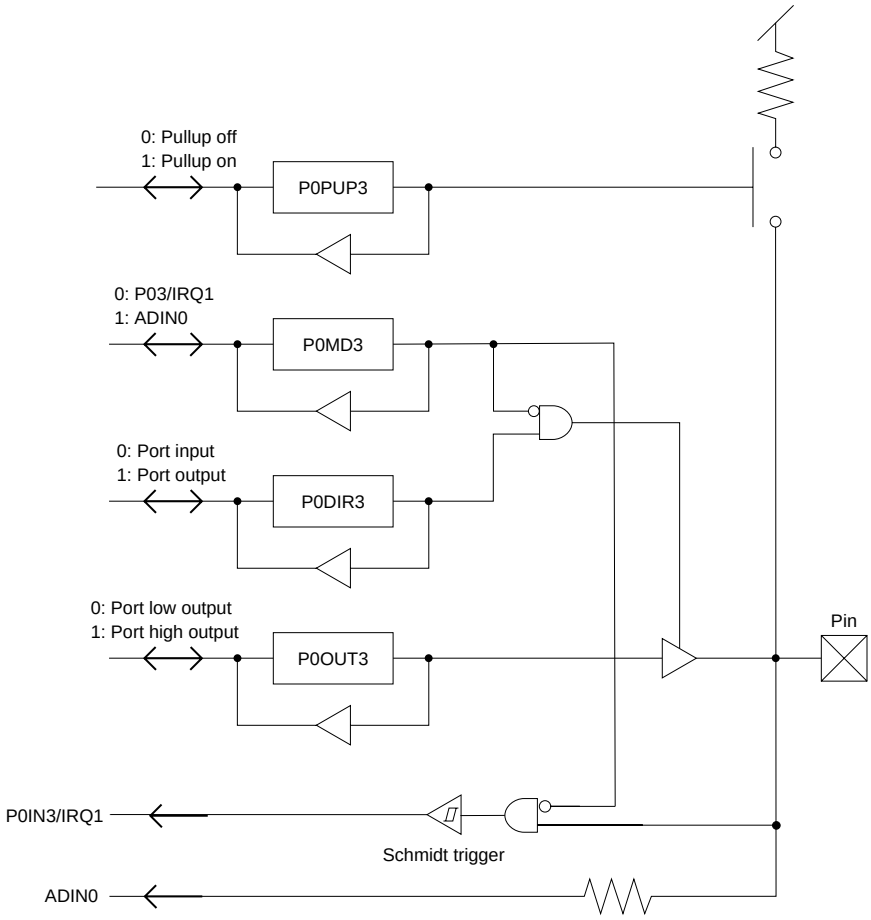
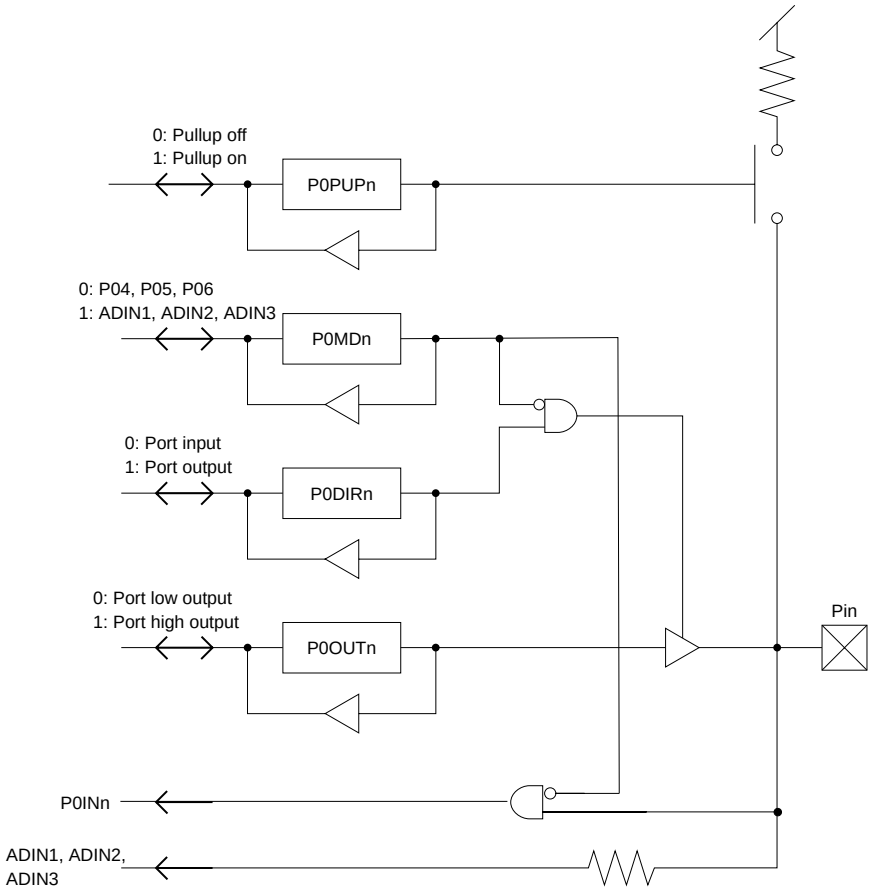
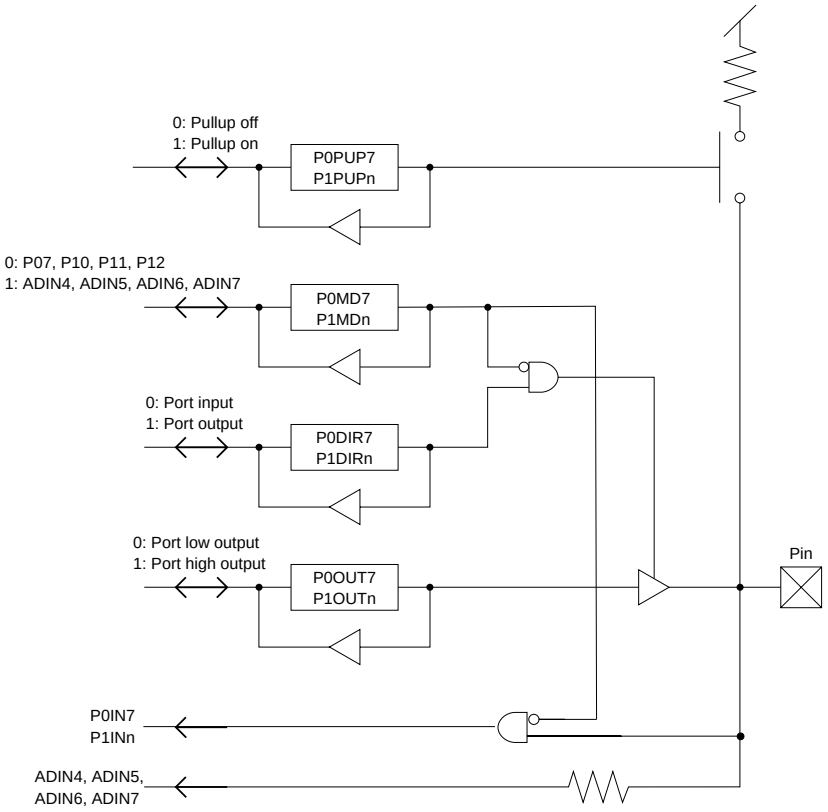


Figure 4-4 P03/ADIN0/IRQ1 (Port 0)



Note: n = 4: P04, n = 5: P05, n = 6: P06

Figure 4-5 P04/ADIN1, P05/ADIN2, and P06/ADIN3 (Port 0)



Note: n = 0: P10, n = 1: P11, n = 2: P12

Figure 4-6 P07/ADIN4 (Port 0), P10/ADIN5, P11/ADIN6, and P12/ADIN7 (Port 1)

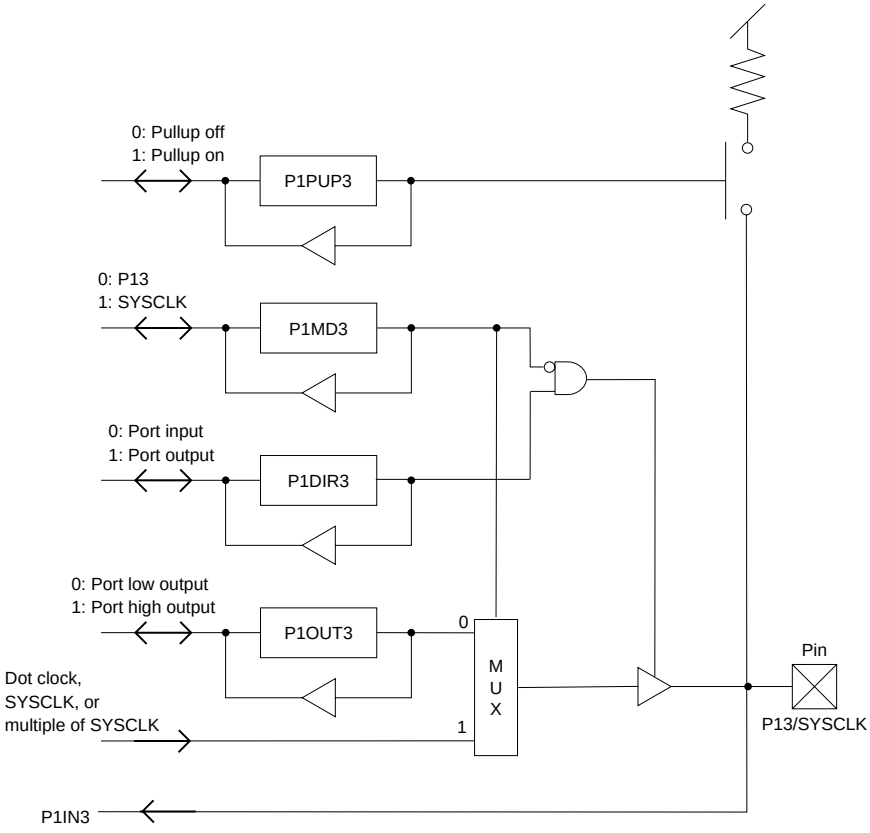


Figure 4-7 P13/SYSCLK (Port 1)

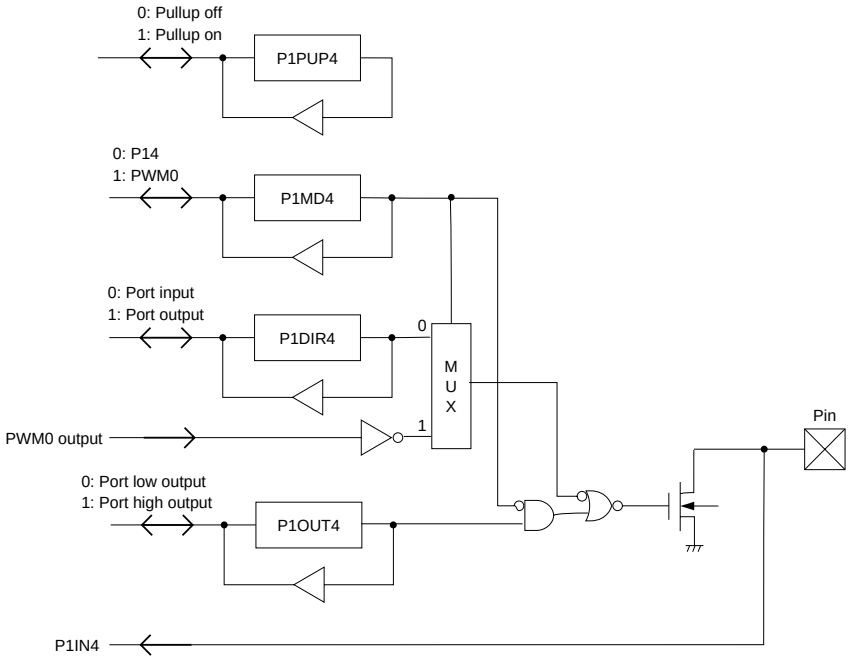
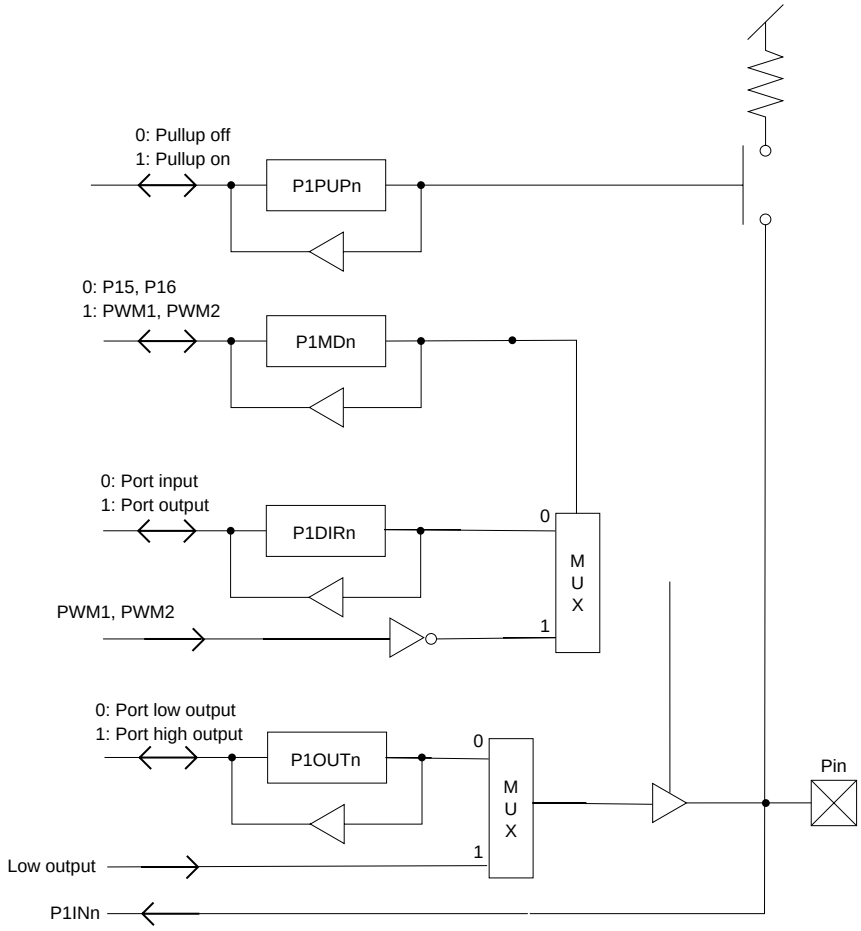


Figure 4-8 P14/PWM0 (Port 1)



Note: n = 5: P15, n = 6: P16

Figure 4-9 P15/PWM1 and P16/PWM2 (Port 1)

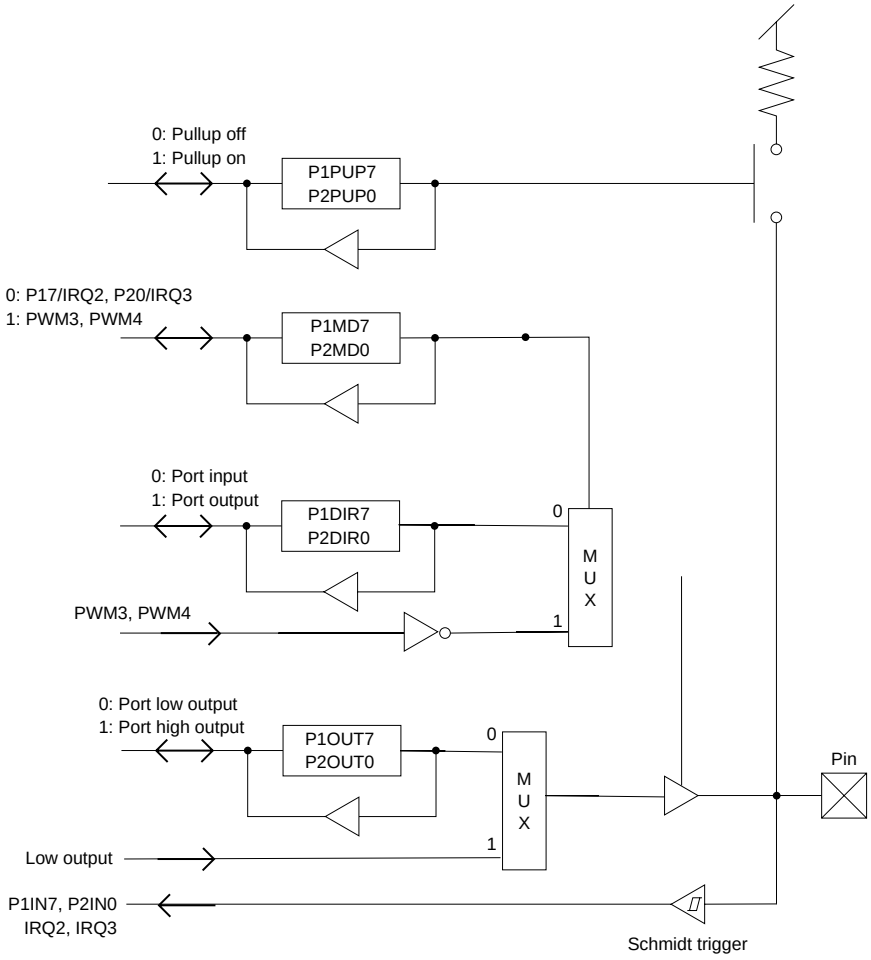


Figure 4-10 P17/PWM3/IRQ2 (Port 1) and P20/PWM4/IRQ3 (Port 2)

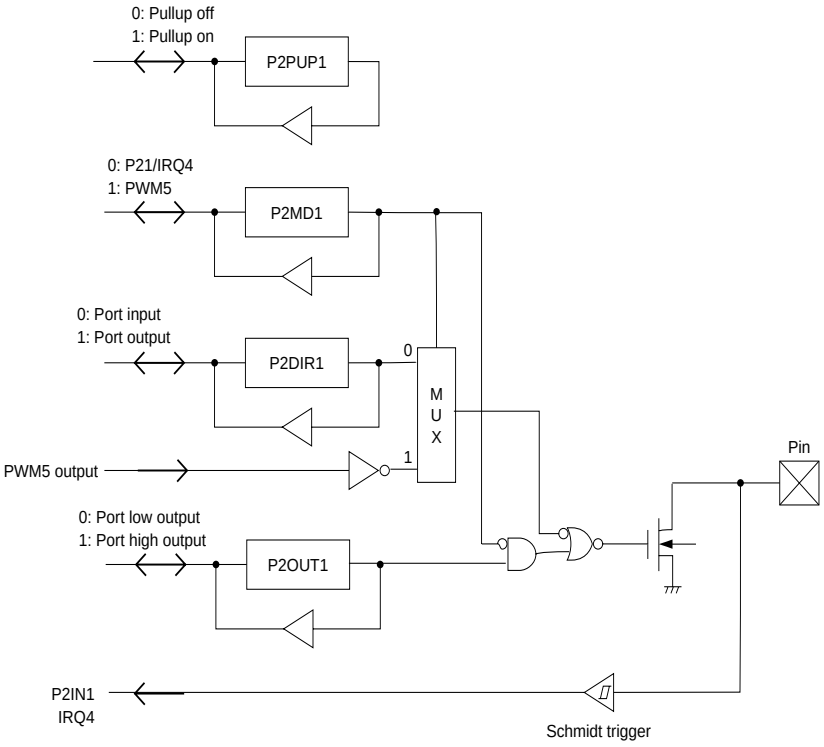
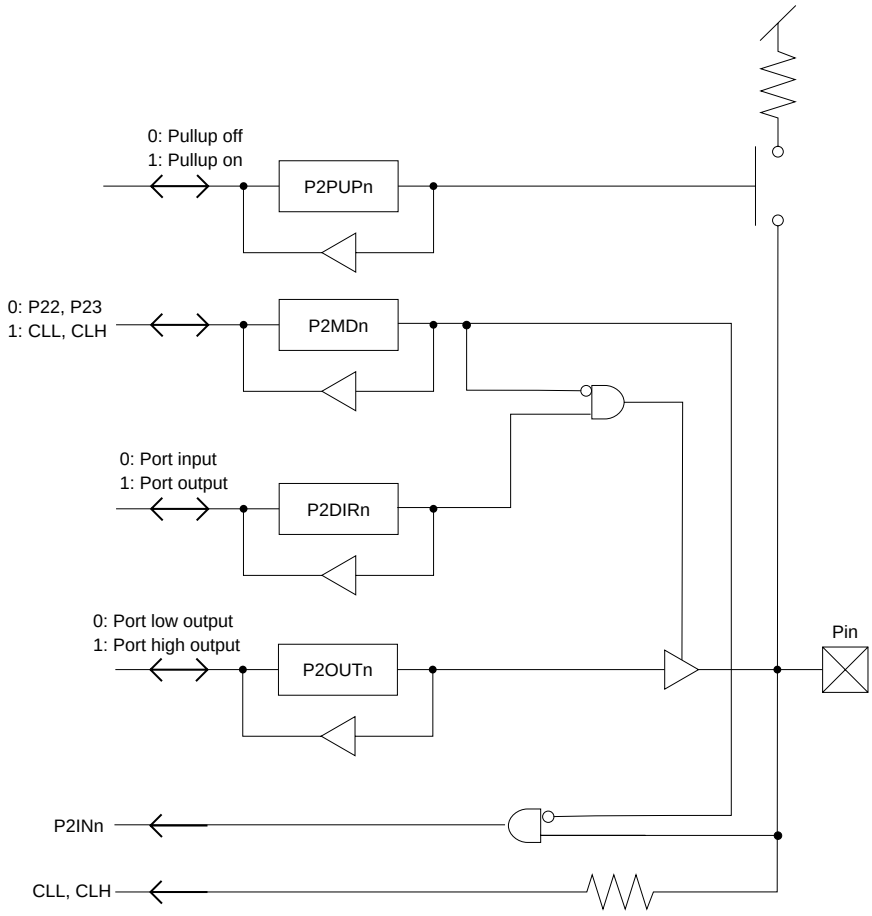
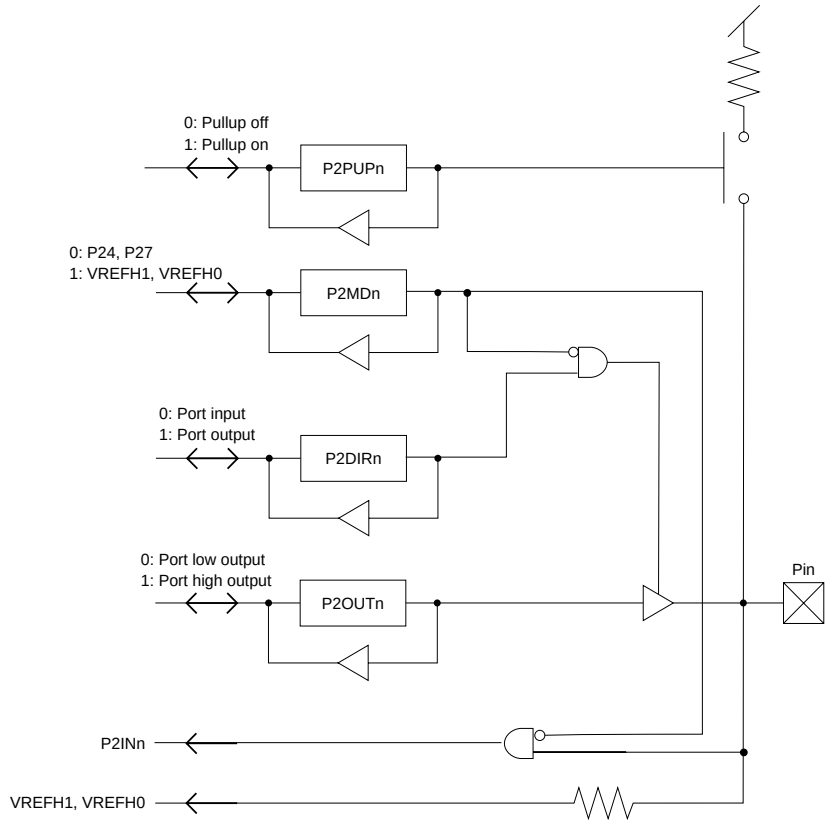


Figure 4-11 P21/PWM5/IRQ4 (Port 2)



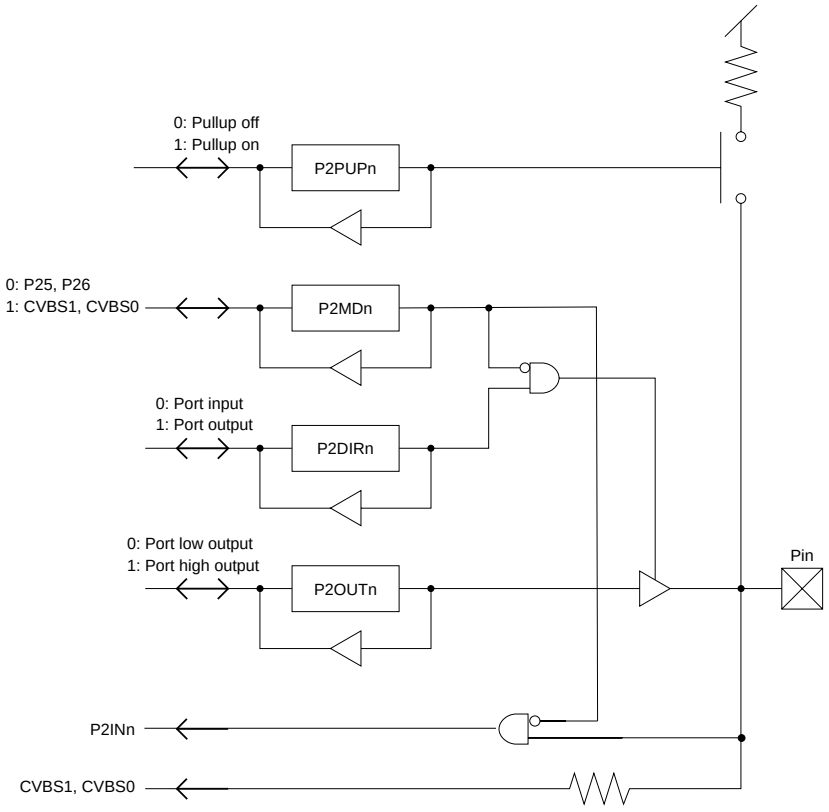
Note: n = 2: P22, n = 3: P23

Figure 4-12 P22/CLL and P23/CLH (Port 2)



Note: n = 4: P24, n = 7: P27

Figure 4-13 P24/VREFH1 and P27/VREFH0 (Port 2)



Note: n = 5: P25, n = 6: P26

Figure 4-14 P25/CVBS1 and P26/CVBS0 (Port 2)

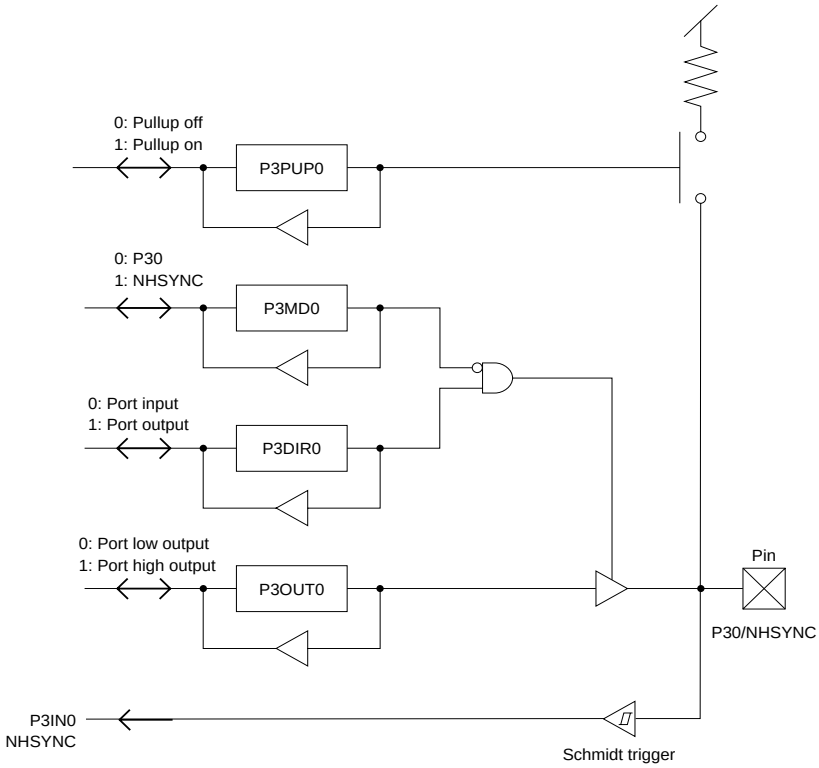


Figure 4-15 P30/NHSYNC (Port 3)

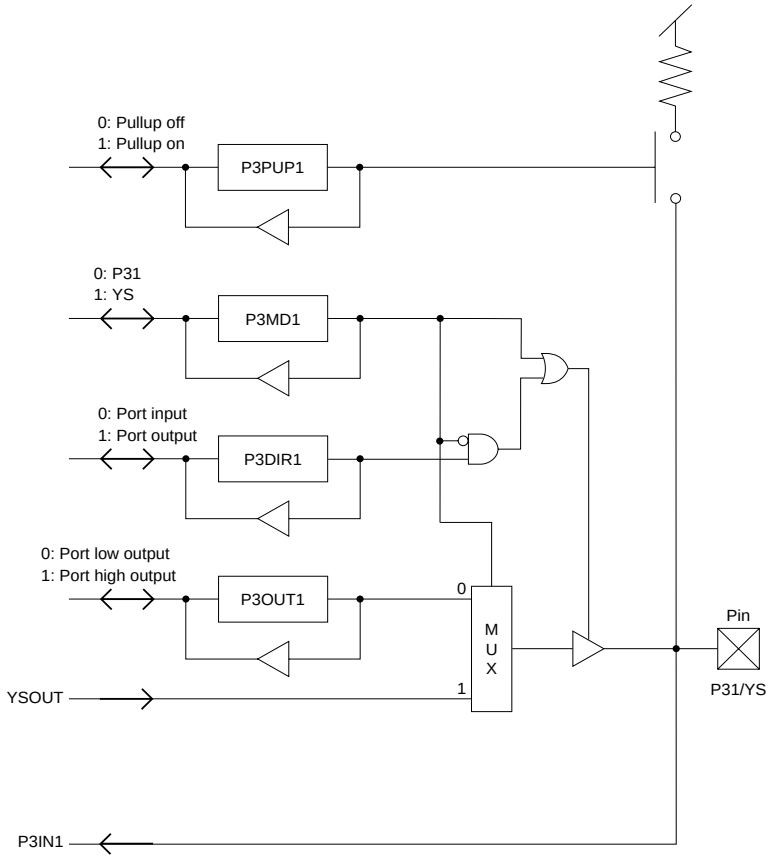
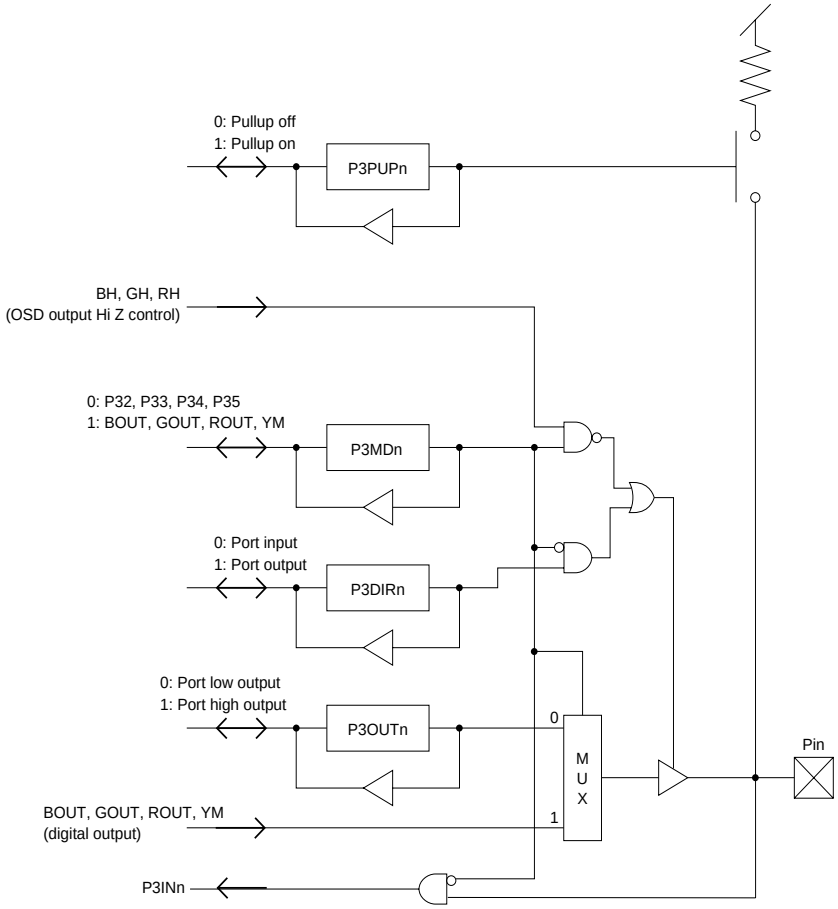


Figure 4-16 P31/YS (Port 3)



Note: n = 2: P32, n = 3: P33, n = 4: P34, n = 5: P35

Figure 4-17 P32/BOUT, P33/GOUT, P34/ROUT, and P35/YM (Port 3)

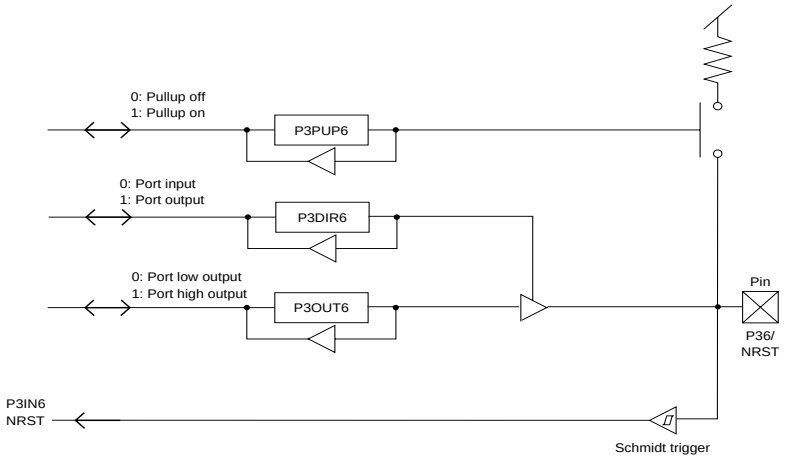


Figure 4-18 P36/NRST (Port 3)

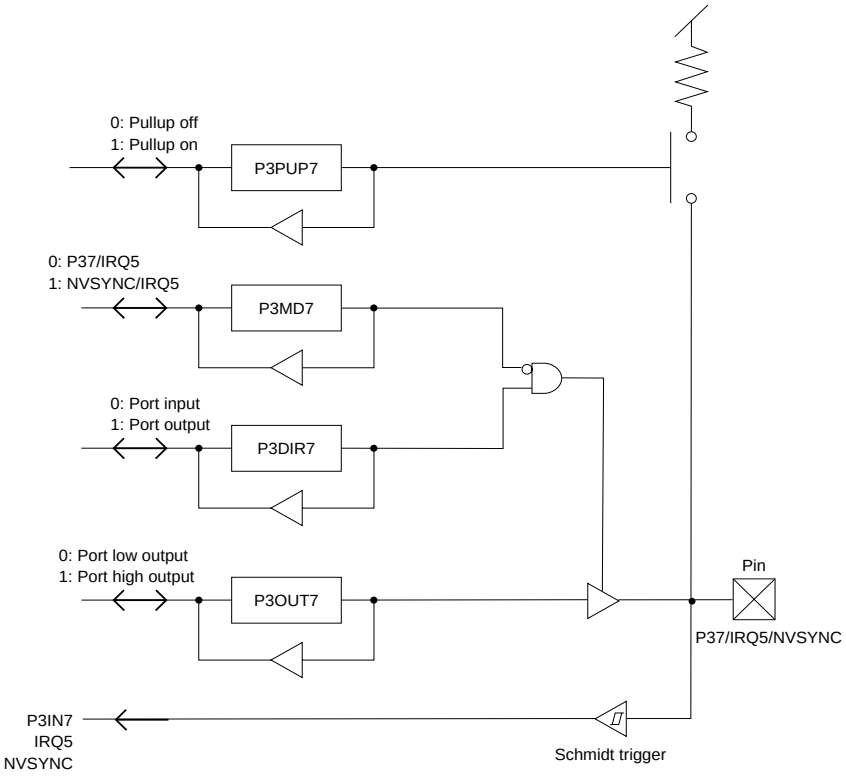


Figure 4-19 P37/NVSYNC/IRQ5 (Port 3)

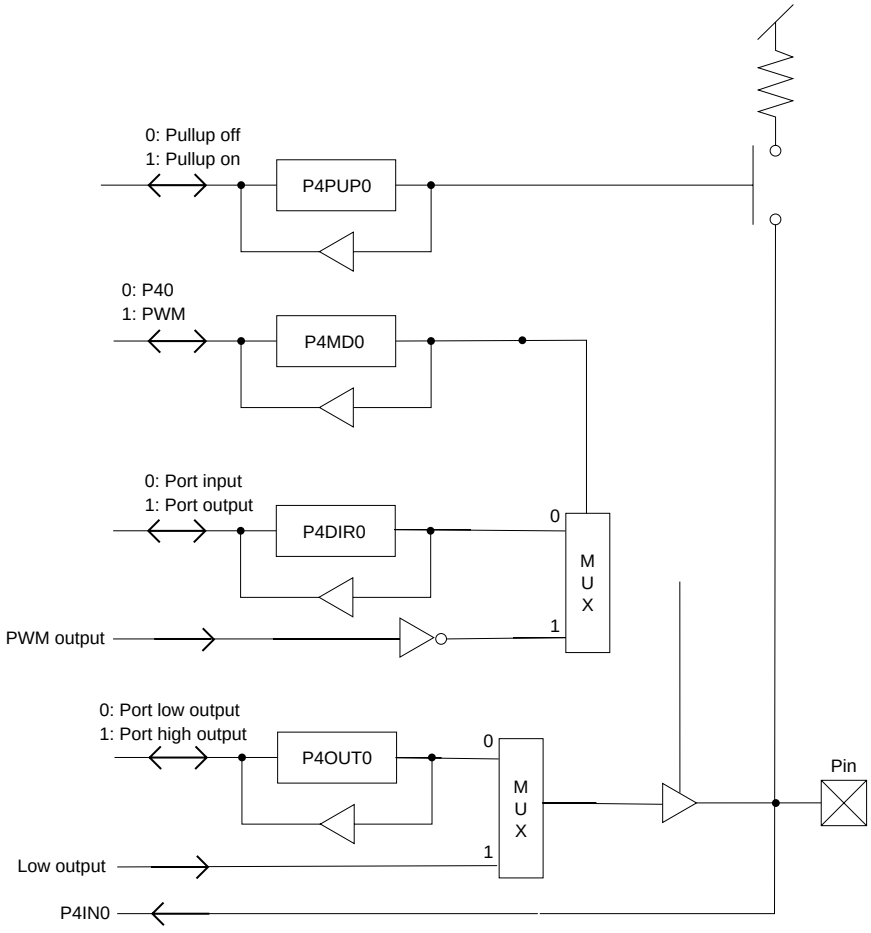


Figure 4-20 P40/PWM (Port 4)

4.3 I/O Port Control Registers

P0PLU–P3PLU: Ports 0–3 Pullup Resistor Control Registers **x'03F40'–x'03F43'**

Bit:	7	6	5	4	3	2	1	0
	PnPLU7	PnPLU6	PnPLU5	PnPLU4	PnPLU3	PnPLU2	PnPLU1	PnPLU0
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P4PLU: Port 4 Pullup Resistor Control Register **x'03F44'**

Bit:	7	6	5	4	3	2	1	0
	0	0	0	0	0	P4PLU2	P4PLU1	P4PLU0
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R/W	R/W	R/W

The PnPLU registers control the port pullup resistors. The bit number corresponds to the associated pin number. For instance, P0PLU7 applies to the P07 pin. These are 8-bit access registers.

0: Pullup resistor off

1: Pullup resistor on

P0OUT–P3OUT: Ports 0–3 Output Control Registers **x'03F10'–x'03F13'**

Bit:	7	6	5	4	3	2	1	0
	PnOUT7	PnOUT6	PnOUT5	PnOUT4	PnOUT3	PnOUT2	PnOUT1	PnOUT0
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Writing a 1 to P3DIR6 and a 0 to P3OUT6 causes a reset.

P4OUT: Port 4 Output Control Register **x'03F14'**

Bit:	7	6	5	4	3	2	1	0
	0	0	0	0	0	P4OUT2	P4OUT1	P4OUT0
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R/W	R/W	R/W

The PnOUT registers contain the port output data. The bit number corresponds to the associated pin number. For instance, P0OUT7 applies to the P07 pin. These are 8-bit access registers.

P0IN – P3IN: Ports 0–3 Input Registers **x'03F20' – x'03F23'**

Bit:	7	6	5	4	3	2	1	0
	PnIN7	PnIN6	PnIN5	PnIN4	PnIN3	PnIN2	PnIN1	PnIN0
Reset:	Pin	Pin	Pin	Pin	Pin	Pin	Pin	Pin
R/W:	R	R	R	R	R	R	R	R

P4IN: Port 4 Input Register **x'03F24'**

Bit:	7	6	5	4	3	2	1	0
	0	0	0	0	0	P4IN2	P4IN1	P4IN0
Reset:	0	0	0	0	0	Pin	Pin	Pin
R/W:	R	R	R	R	R	R	R	R

The PnIN registers contain the port input data. The bit number corresponds to the associated pin number. For instance, P0IN7 applies to the P07 pin. These are 8-bit access registers.

P0DIR – P3DIR: Ports 0–3 I/O Control Registers **x'03F30' – x'03F33'**

Bit:	7	6	5	4	3	2	1	0
	PnDIR7	PnDIR6	PnDIR5	PnDIR4	PnDIR3	PnDIR2	PnDIR1	PnDIR0
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P4DIR: Port 4 I/O Control Register **x'03F34'**

Bit:	7	6	5	4	3	2	1	0
	0	0	0	0	0	P4DIR2	P4DIR1	P4DIR0
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R/W	R/W	R/W

The PnDIR registers control the I/O direction of the ports. The bit number corresponds to the associated pin number. For instance, P0DIR7 applies to the P07 pin. These are 8-bit access registers.

0: Input

1: Output

P0MD: Port 0 Output Mode Register **x'03F28'**

Bit:	7	6	5	4	3	2	1	0
	P0MD7	P0MD6	P0MD5	P0MD4	P0MD3	P0MD2	P0MD1	P0MD0
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P0MD is an 8-bit access register.

P0MD7: P07 function switch
- 0: P07
- 1: ADIN4

P0MD6: P06 function switch
- 0: P06
- 1: ADIN3

P0MD5: P05 function switch
- 0: P05
- 1: ADIN2

P0MD4: P04 function switch
- 0: P04
- 1: ADIN1

P0MD3: P03 function switch
- 0: P03/IRQ1
- 1: ADIN0/IRQ1

P0MD2: P02 function switch
- 0: P02
- 1: SCL1

P0MD1: P01 function switch
- 0: P01
- 1: SDA1

P0MD0: P00 output switch
- 0: P00/IRQ0
- 1: RMIN/IRQ0

P1MD: Port 1 Output Mode Register **x'03F29'**

Bit:	7	6	5	4	3	2	1	0
	P1MD7	P1MD6	PIMD5	P1MD4	PIMD3	P1MD2	PIMD1	P1MD0
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P1MD is an 8-bit access register.

P1MD7: P17 output switch
- 0: P17/IRQ2
- 1: PWM3/IRQ2

P1MD6: P16 output switch
- 0: P16
- 1: PWM2

P1MD5: P15 output switch
- 0: P15
- 1: PWM1

P1MD4: P14 output switch
- 0: P14
- 1: PWM0

P1MD3: P13 output switch
- 0: P13
- 1: SYSCLK or clock divided from SYSCLK

P1MD2: P12 function switch
- 0: P12
- 1: ADIN7

P1MD1: P11 function switch
- 0: P11
- 1: ADIN6

P1MD0: P10 function switch
- 0: P10
- 1: ADIN5

P2MD: Port 2 Output Mode Register **x'03F2A'**

Bit:	7	6	5	4	3	2	1	0
	P2MD7	P2MD6	P2MD5	P2MD4	P2MD3	P2MD2	P2MD1	P2MD0
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P2MD is an 8-bit access register.

P2MD7: P27 function switch
- 0: P27
- 1: VREFH0

P2MD6: P26 function switch
- 0: P26
- 1: CVBS0

P2MD5: P25 function switch
- 0: P25
- 1: CVBS1

P2MD4: P24 function switch
- 0: P24
- 1: VREFH1

P2MD3: P23 function switch
- 0: P23
- 1: CLH

P2MD2: P22 function switch
- 0: P22
- 1: CLL

P2MD1: P21 output switch
- 0: P21/IRQ4
- 1: PWM5/IRQ4

P2MD0: P20 output switch
- 0: P20/IRQ3
- 1: PWM4/IRQ3

Driving P2MD5 and P2MD6 high switches the functions of P22 and P23 to CLH and CLL.

Always set P2MD5 and P2MD6 to 1or 0 simultaneously. When one is set to 1 and the other is set to 0, closed-caption decoders don't work correctly.

P3MD: Port 3 Output Mode Register **x'03F2B'**

Bit:	7	6	5	4	3	2	1	0
	P3MD7	P3MD6	P3MD5	P3MD4	P3MD3	P3MD2	P3MD1	P3MD0
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P3MD is an 8-bit access register.

P3MD7: P37 function switch
- 0: P37/IRQ5
- 1: NSYNC/IRQ5

P3MD6: This bit exists, but contains no function.

P3MD5: P35 output switch
- 0: P35
- 1: YM

P3MD4: P34 output switch
- 0: P34
- 1: ROUT

P3MD3: P33 output switch
- 0: P33
- 1: GOUT

P3MD2: P32 output switch
- 0: P32
- 1: BOUT

P3MD1: P31 output switch
- 0: P31
- 1: YS

P3MD0: P30 function switch
- 0: P30
- 1: NHSYNC

P4MD: Port 4 Output Mode Register **x'03F2C'**

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	P4MD2	P4MD1	P4MD0
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R/W	R/W	R/W

P4MD is an 8-bit access register.

P4MD2: P42 function switch
- 0: P42
- 1: SCL0

P4MD1: P41 function switch
- 0: P41
- 1: SDA0

P4MD0: P40 output switch
- 0: P40
- 1: PWM

PCNT0: Port Control Register 0 **x'03F4A'**

Bit:	7	6	5	4	3	2	1	0
	I2C SEL1	I2C SEL0	OSD POFF	ADC1 ON	ADC0 ON	RMC OFF	VBI1 OFF	VBI0 OFF
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

PCNT0 is an 8-bit access register.

I2CSEL1: SDA1, SCL1 enable
- 0: Disable
- 1: Enable

I2CSEL0: SDA0, SCL0 enable
- 0: Disable
- 1: Enable

OSDPOFF: OSD enable
- 0: Disable
- 1: Enable

ADC1ON: ADC enable for closed-caption decoder 1
- 0: Disable
- 1: Enable

ADC0ON: ADC enable for closed-caption decoder 0
- 0: Disable
- 1: Enable

RMCOFF: IR remote signal receiver enable
- 0: Enable
- 1: Disable

VBI1OFF: Closed-caption decoder 1 enable

0: Enable

1: Disable

VBI0OFF: Closed-caption decoder 0 enable

0: Enable

1: Disable

PCNT2: Port Control Register 2 **x'03F4E'**

Bit:	7	6	5	4	3	2	1	0
	—	—	—	I2C OFF	PWM OFF	OSD REGE	SCLKF1	SCLKF0
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R/W	R/W	R/W	R/W

I2COFF: I^2C function enable

0: Enable

1: Disable

PWMOFF: PWM function enable

0: Enable

1: Disable

OSDREGE: OSD registers read/write enable

0: Disable

1: Enable

SCLKF[1:0]: SYSCLK frequency select

00: SYSCLK × 4096

01: Dot clock

10: SYSCLK × 2

11: SYSCLK

5 Prescalar

5.1 Description

The MN101C46F has two prescalars that can be used simultaneously and shared between different peripheral functions. They use the f_{OSC} and f_S as their base clocks for counting. Their hardware is structured as follows.

- Prescalar 0 (f_{OSC} based) 7-bit prescalar
- Prescalar 1 (f_S based) 3-bit prescalar

Prescalar 0 outputs divided clocks of $f_{OSC}/2$, $f_{OSC}/4$, $f_{OSC}/16$, $f_{OSC}/32$, $f_{OSC}/64$, and $f_{OSC}/128$. Prescalar 1 outputs divided clocks of $f_S/2$, $f_S/4$, and $f_S/8$. Use prescalars when you are employing divided clocks based on f_{OSC} or f_S in the following peripheral functions.

- Timer 2 (8-bit timer counter)
- Timer 3 (8-bit timer counter)
- Timer 4 (8-bit timer counter)
- D/A conversion function

See section 2.10.3, “Special Function Registers,” on page 32 for more information on f_{OSC} and f_S.

5.1.1 Peripheral Functions that Use Prescalar Output

The table below lists the clocks that are selectable for each of the peripheral functions that use the prescalar block.

Table 5-1 Peripheral Functions that Use Prescalar Output

Selectable Divided Clocks	Peripheral Function			
	Timer 2	Timer 3	Timer 4	DAC
$f_{OSC}/2$	—	—	—	—
$f_{OSC}/4$	√	√	√	—
$f_{OSC}/16$	√	√	√	—
$f_{OSC}/32$	√	—	—	—
$f_{OSC}/64$	√	√	√	—
$f_{OSC}/128$	—	√	—	—
$f_S/2$	√	√	√	—
$f_S/4$	√	—		—
$f_S/8$	—	√	—	√

5.1.2 Prescalar Block Diagram

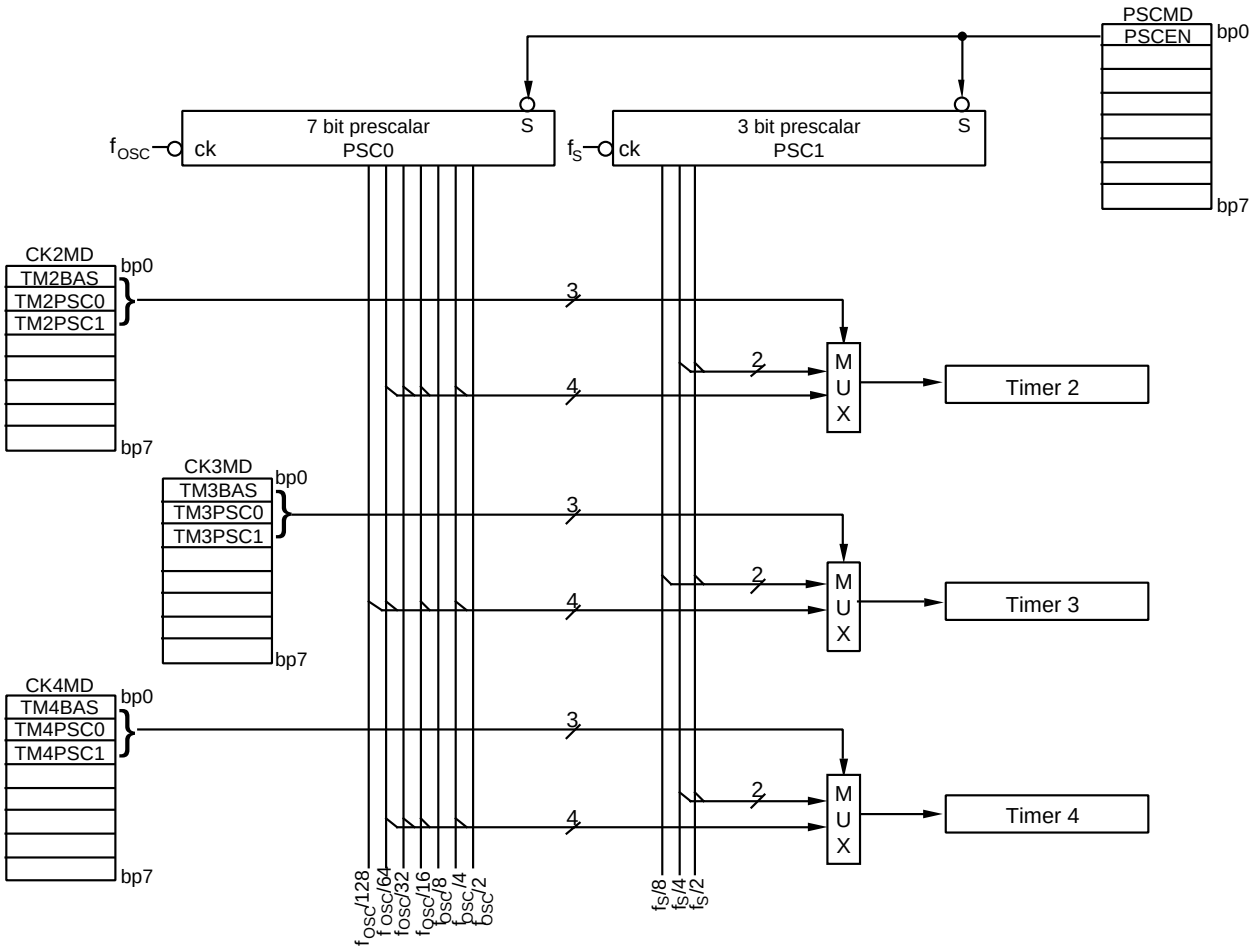


Figure 5-1 Prescalar Block Diagram

5.2 Prescalar Control Registers

5.2.1 Prescalar Control Registers

There are four prescalar control registers.

Table 5-2 Prescalar Control Registers

Register	Address	R/W	Description
PSCMD	x'03F6F'	R/W	Prescalar control register
CK2MD	x'03F5E'	R/W	Timer 2 prescalar select register
CK3MD	x'03F5F'	R/W	Timer 3 prescalar select register
CK4MD	x'03F66'	R/W	Timer 4 prescalar select register

Use the prescalar control register (PSCMD), the timer prescalar select registers (CKnMD), and the serial transfer clocks select registers (SCnCKS) to control prescalar operation and select prescalar output.

PSCMD: Prescalar control register **x'03F6F'**

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	PSCEN
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R/W

The prescalar control register enables or disables prescalar counting.

PSCEN: Controls prescalar 0 and 1 counts

0: Disable

1: Enable

The timer prescalar select registers select the count clocks for the 8-bit timers.

CK2MD: Timer 2 prescalar select register **x'03F5E'**

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	TM2 PSC1	TM2 PCS0	TM2 BAS
Reset:	0	0	0	0	0	X	X	X
R/W:	R	R	R	R	R	R/W	R/W	R/W

Table 5-3 Timer 2 prescalar select register

TM2PSC1	TM2PSC0	TM2BAS	Clock selected
0	0	0	$f_{OSC}/4$
0	1	0	$f_{OSC}/16$
1	0	0	$f_{OSC}/32$
1	1	0	$f_{OSC}/64$
—	0	1	$f_S/2$
—	1	1	$f_S/4$

CK3MD: Timer 3 prescalar select register **x'03F5F'**

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	TM3 PSC1	TM3 PCS0	TM3 BAS
Reset:	0	0	0	0	0	X	X	X
R/W:	R	R	R	R	R	R/W	R/W	R/W

Table 5-4 Timer 3 Prescalar Select Register

TM3PSC1	TM3PSC0	TM3BAS	Clock selected
0	0	0	$f_{OSC}/4$
0	1	0	$f_{OSC}/16$
1	0	0	$f_{OSC}/64$
1	1	0	$f_{OSC}/128$
—	0	1	$f_S/2$
—	1	1	$f_S/8$

CK4MD: Timer 4 prescalar select register **x'03F66'**

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	TM4 PSC1	TM4 PCS0	TM4 BAS
Reset:	0	0	0	0	0	X	X	X
R/W:	R	R	R	R	R	R/W	R/W	R/W

Table 5-5 Timer 4 Prescalar Select Register

TM4PSC1	TM4PSC0	TM4BAS	Clock selected
0	0	0	$f_{OSC}/4$
0	1	0	$f_{OSC}/16$
1	0	0	$f_{OSC}/32$
1	1	0	$f_{OSC}/64$
—	0	1	$f_S/2$
—	1	1	$f_S/4$

5.3 Operation of the Prescalar Function

5.3.1 Prescalar Operation

■ **Prescalars 0 and 1**

Prescalars 0 and 1 are free-running counters of 7 and 3 bits, respectively, that output clocks produced by dividing the reference clocks. Start and stop them incrementing using the PSCEN flag of the prescalar control register (PSCMD).

■ **Count timing (prescalars 0 and 1)**

Prescalar 0 increments on the negative edge of f_{OSC}. Prescalar 1 increments on the negative edge of f_S.

■ **Peripheral functions that use the divided clocks output by the prescalars**

The following functions can use the divided clocks output by the prescalars. The associated control registers for selecting clocks are also shown.

To use a prescalar divided clock, first enable the prescalar count, then start the peripheral function.

Table 5-6 Peripheral Functions that Can Use Prescalar Clocks

Function	Control Register
Timer 2 count clock	CK2MD
Timer 3 count clock	CK3MD
Timer 4 count clock	CK4MD

5.3.2 Example of Prescalar Operation Setup

■ **Selecting a count clock for timer 2**

The following example shows and describes setup procedures for selecting the $f_{OSC}/16$ output by prescalar 0 as the count clock for timer 2.

Table 5-7 Procedures for Setting up a Count Clock for Timer 2

Procedure	Description
(1) Select prescalar output CK2MD (x'03F5D') Bits 2:1—TM2PSC[1:0]=01 Bit 0—TM2BAS=0	(1) Selects $f_{OSC}/16$ as the prescalar output using TM2PSC[1:0] and TM2BAS in the timer 2 prescalar select register (CK2MD).
(2) Enable prescalar output PSCMD (x'03F6F') Bit 0—PSCEN=1	(2) Enables prescalar counting by setting the PSCEN flag in the prescalar control register (PSCMD) to 1.

Enable prescalar counting by setting the PSCEN flag in the prescalar control register (PSCMD). The prescalar starts counting when it is enabled. Start the timer counting after the prescalar is set up. You must also select the prescalar output at the timer using the timer mode register.

6 8-Bit Timers

6.1 Introduction to the 8-Bit Timers

The MN101C46F has three 8-bit timers (timers 2, 3, and 4). Timers 2 and 3 can be cascaded and used as a 16-bit timer. You cannot cascade timer 4.

You can select divided clocks based on f_{OSC} and f_S for the clock sources of the timers by using the output of the prescalars. An IR remote signal receiver output circuit is built in.

6.1.1 8-Bit Timer Function

The table below describes the functions that can use these timers.

Table 6-1 Timer Function

	Timer 2 (8 bits)	Timer 3 (8 bits)	Timer 4 (8 bits)
Interrupt sources	TM2IRQ	TM3IRQ	TM4IRQ
Timer operation	√	√	√
Cascade connection?	√	√	No
Clock sources	f_{OSC} $f_{OSC}/4$ $f_{OSC}/16$ $f_{OSC}/32$ $f_{OSC}/64$ $f_S/2$ $f_S/4$ f_X	f_{OSC} $f_{OSC}/4$ $f_{OSC}/16$ $f_{OSC}/64$ $f_{OSC}/128$ $f_S/2$ $f_S/8$ f_X	f_{OSC} $f_{OSC}/4$ $f_{OSC}/16$ $f_{OSC}/32$ $f_{OSC}/64$ $f_S/2$ $f_S/4$ f_X
f_{OSC}: Machine clock (oscillation for fast operation) f_X: Machine clock (oscillation for slow operation) f_S: System clock. See section 2.13, “Setting the Clock Switch Register.”			

6.1.2 8-Bit Timer Block Diagrams

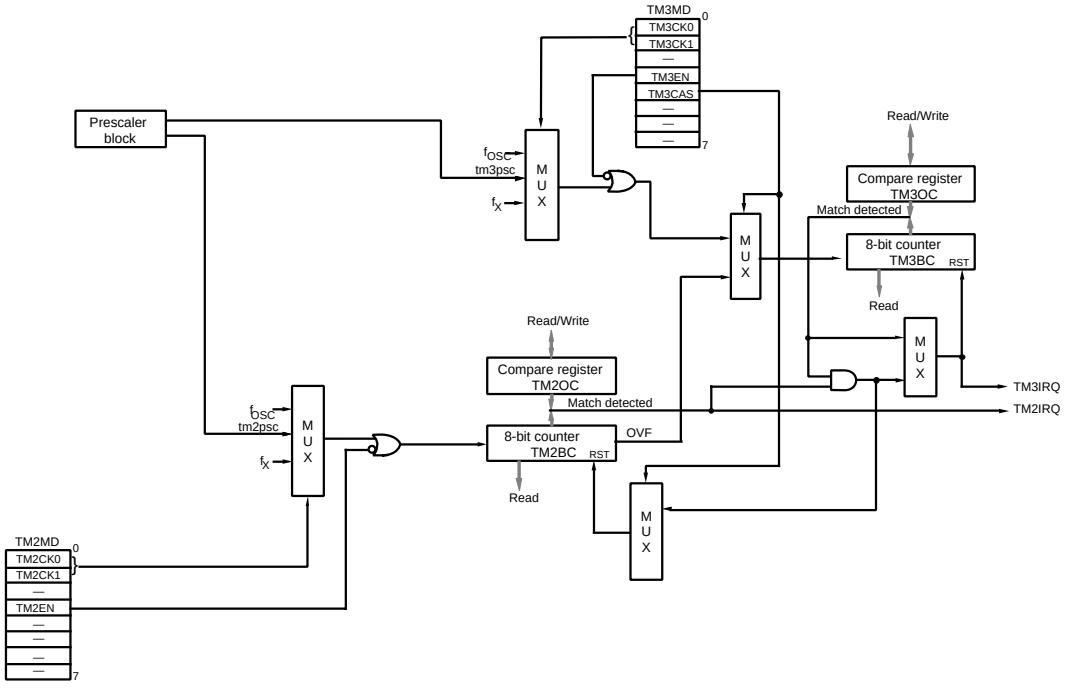


Figure 6-1 Block Diagram of Timers 2 and 3

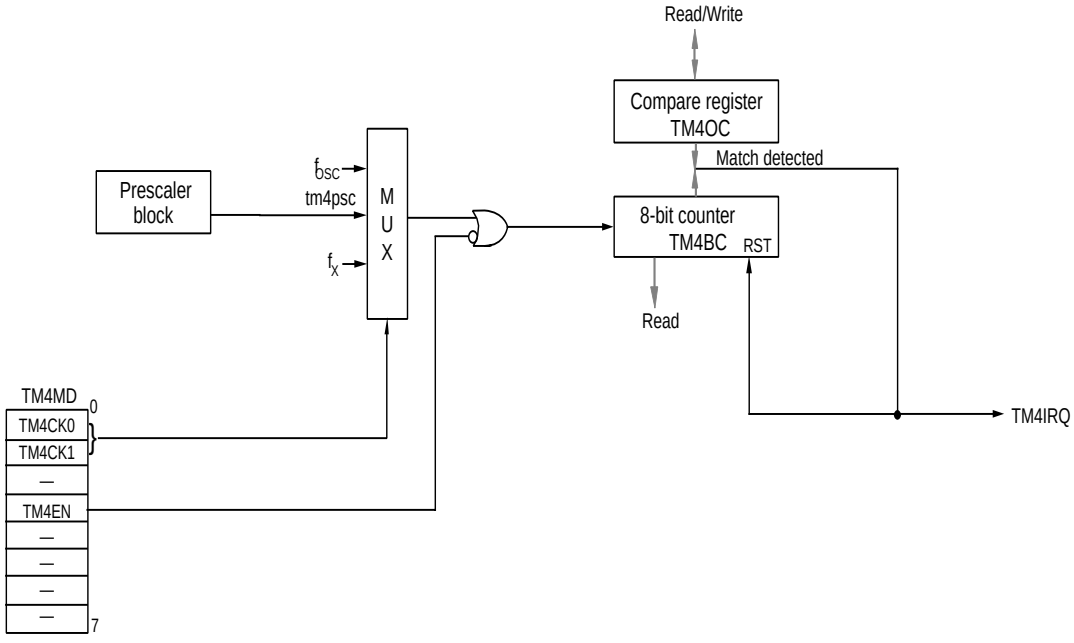


Figure 6-2 Block Diagram of Timer 4

6.2 Operation of the 8-Bit Timers

6.2.1 Operation of the 8-Bit Timers

Timer functions are able to generate interrupts repeatedly at set intervals.

■ **Operation of 8-bit timers 2, 3, and 4**

Timer interrupts are generated at intervals that are preset using the clock source selection and the compare register (TMnOC) setting. When the binary counter (TMnBC) matches the compare register setting, an interrupt request is generated at the next count clock, the binary counter is cleared, and up counting begins again from x'00'. Select from among the following clock sources for different timers.

Table 6-2 Clock Sources with Timers Running (Timers 2, 3, and 4)

Clock Source	Count Period	Timer 2 (8 Bits)	Timer 3 (8 Bits)	Timer 4 (8 Bits)
f_{OSC}	125 ns	√	√	√
$f_{OSC}/4$	500 ns	√	√	√
$f_{OSC}/16$	2	√	√	√
$f_{OSC}/32$	4 µs	√	—	√
$f_{OSC}/64$	8 µs	√	√	√
$f_{OSC}/128$	16 µs	—	√	—
$f_S/2$	500 ns	√	√	√
$f_S/4$	1 µs	√	—	√
$f_S/8$	2 µs	—	√	—
f_X	500 ns	√	√	√
f_{OSC} = 8 MHz, f_X = 2 MHz, $f_S = f_{OSC}/2$ = 4 MHz.				

■ **Count timing for timer operation (timers 2, 3, and 4)**

The binary counter counts up using the selected clock source as the count clock. The operation shown below is the basic sequence for all 8-bit timers.

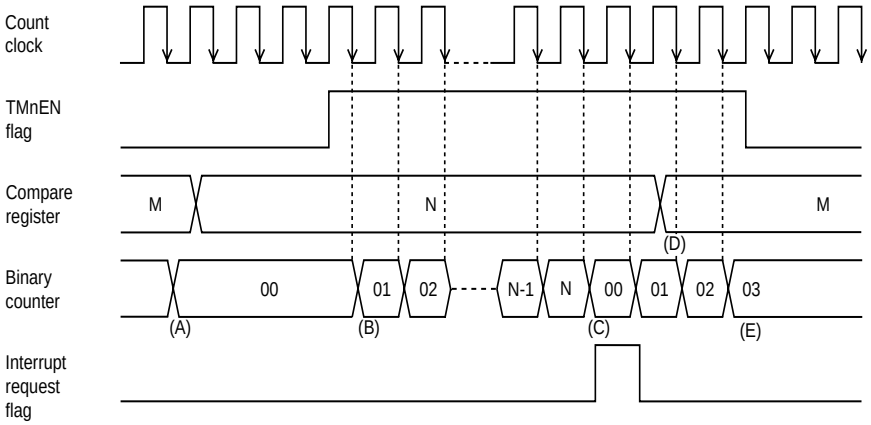


Figure 6-3 Count Timing for Timer Operation (Timers 2, 3, and 4)

(a) When a value is written to a compare register while the TMnEN flag indicates a stop (0), the binary counter is cleared to x'00' during that write cycle.

(b) The binary counters starts counting when the TMnEN flag indicates operation (1). Negative edges of the count clock are counted.

(c) When the binary counter matches the compare register value, the interrupt request flag is set at the next count clock, the binary counter is cleared to x'00', and up counting begins again.

(d) Rewriting the compare register while the TMnEN flag indicates operation (1) does not change the binary counter.

(e) The binary counter stops running when the TMnEN flag indicates a stop (0).

When the binary counter matches the compare register value, the interrupt request flag is set at the next count clock, and the binary counter is cleared, so observe the following relationship:
(Compare register setting)= (Clocks till interrupt request is generated - 1)

Setting a compare register value that is smaller than the binary counter value during a count will cause the binary counter to count up till one end overflows.

When processing by means of interrupts, clear the timer interrupt request flag prior to starting the timer.

The timing pattern when a timer n interrupt request is generated with a setting of TMnOC=x'00' is the same as when the setting is x'01'.

6.2.2 Example of 8-Bit Timer Operation Setup

■ **Timer operation (timers 2, 3, and 4)**

In the following example, a clock function is implemented by using timer 0 to generate interrupts at set intervals. $f_S/4$ is used as the clock source (f_{OSC} = 20 MHz), and interrupts are generated at every 250 divisions (100 µs). The procedures are described below.

Table 6-3 Procedure for Setting up an 8-Bit Timer

Procedure	Description
(1) Check that counter is stopped. TM2MD (x'03F5C') bit 3—TM0EN=0	(1) Set the TM2EN flag of the timer 2 mode register (TM2MD) to 0 to stop the count.
(2) Select count clock source. TM2MD (x'03F5C') bits 1:0—TM2CK[1:0]=01	(2) Use the TM2CK[1:0] flag of the TM2MD register to select prescalar output as the clock source.
(3) Select prescalar output and enable it. CK2MD (x'03F5E') bits 2:1—TM0PSC[1:0]=01, bit 0—TM0BAS=1 PSCMD (x'03F6F') bit 0—PSCEN=1	(3) Use the TM2PSC[1:0] field and the TM2BAS flag of the timer 2 prescalar select register (CK2MD) to select $f_S/4$ as the prescalar output. Also set the PSCEN flag in the prescalar control register (PSCMD) to 1 to enable the prescalar count.
(4) Set period for generating interrupts. TM2OC (x'03F5A')=x'F9'	(4) Set a value for the interrupt generation period in the timer 2 compare register (TM2OC). Since we are using 250 divisions, set to 249 (x'F9'). The timer 2 binary counter (TM2BC) will be initialized to x'00'.
(5) Set the interrupt level. TM2ICR (x'03FEB') bits 7:6—TM2LV[1:0]=10	(5) Set the interrupt level using the TM2LV[1:0] field in the timer 2 interrupt control register (TM2ICR). If the interrupt request flag might have already been set, clear the request flag. See section 3.5, “Setting the Interrupt Flags.”
6) Enable interrupts. TM2ICR (x'03FEB') bit 1—TM2IE=1	(6) Enable interrupts by setting the TM2IE flag of the TM2ICR register to 1.
(7) Start the timer. TM2MD (x'03F5C') bit 3—TM2EN=1	(7) Set the TM2EN flag of the TM2MD register to 1 to start timer 2.

TM2BC starts counting from x'00'. When TM2BC matches the value set in the TM2OC register, the timer 0 interrupt request flag is set at the next count clock and TM2BC starts counting up again from x'00'.

You can switch between binary counters in the count up by changing the TMnEN flag of the TMnMD register at the same time as another bit.

6.3 Operation of the 8-Bit Timer Cascade Connection

The cascade connection links timers 2 and 3 so they can be used together as a single 16-bit timer. The cascaded timer runs on the clock source of timer 2, which corresponds to the eight low-order bits.

Table 6-4 Functions of Cascaded Timers

	Timer 2 + Timer 3 (16 bits)
Interrupt source	TM3IRQ
Timer operation	√
Clock sources	$f_{OSC}/4$ $f_{OSC}/16$ $f_{OSC}/32$ $f_{OSC}/64$ $f_{OSC}/128$ $f_S/2$ $f_S/4$ $f_S/8$ f_X
f_{OSC}: Machine clock (oscillation for fast operation) f_X: Machine clock (oscillation for slow operation) f_S: System clock. See section 2.13, “Setting the Clock Switch Register.”	

When timers are cascaded, both the binary counter and the compare register function as 16-bit registers. Set the TMnEN flags of the mode registers to 1 for both the high order 8-bit timer and the low order 8-bit timer. Use the low order 8-bit timer to select the clock source. All other settings and count timing are the same as when the 8-bit timers are used independently.

When timers 2 and 3 are cascaded, use the interrupt request flag of timer 3. Tie the timer pulse output of timer 2 low. Disable timer 2 interrupts even though no timer 2 interrupt requests will occur.

6.3.1 Example of Cascade Connection Setup

■ **Cascading timers 2 and 3**

In the following setup example, a clock function is implemented by cascading timers 2 and 3 as a 16-bit timer and generating interrupts at set intervals. The selected clock source is $f_S/4$ (f_{OSC} = 20 MHz). It generates an interrupt every 2500 divisions (1 ms). The setup procedures are described below.

Table 6-5 Procedures for Setting up a Cascade Connection

Procedure	Description
(1) Check that counters are stopped. TM2MD (x'03F5C') bit 3—TM2EN=0 TM3MD (x'03F5D') bit 3—TM3EN=0	(1) Set the TM2EN flag of the timer 2 mode register (TM2MD) and the TM3EN flag of the timer 3 mode register (TM3MD) to 0 to stop the count in both timers.
(2) Setup the cascade connection. TM3MD (x'03F5D') bit 4—TM3CAS = 1	(2) Set the TM3CAS flag in the TM3MD register to 1 to cascade timers 2 and 3.
(3) Select count clock source. TM2MD (x'03F5C') bits 1:0—TM2CK[1:0]=01	(3) Use the TM2CK[1:0] flag of the TM2MD register to select prescalar output as the clock source.
(4) Select prescalar output and enable it. CK2MD (x'03F5E') bits 2:1—TM2PSC[1:0]=01 bit 0—TM2BAS=1 PSCMD (x'03F6F') bit 0—PSCEN=1	(4) Use the TM2PSC[1:0] field and the TM2BAS flag of the timer 2 prescalar select register (CK2MD) to select $f_S/4$ as the prescalar output. Also set the PSCEN flag in the prescalar control register (PSCMD) to 1 to enable the prescalar count.
(5) Set the period for generating the interrupts. TMnOC (x'03F5B', x'03F5A')=x'09C3'	(5) Set a value for the interrupt generation period in the timer 1 compare register and timer 0 compare register (TM3OC + TM2OC). Since we are using 2500 divisions, set to x'09C3' (2500 - 1). The timer 3 binary counter and timer 2 binary counter (TM3BC + TM2BC) will be initialized to x'0000'.
(6) Set the low-order timer interrupt level. TM2ICR (x'03FEB') bit 1—TM0IE=0	(6) Set the TM2IE flag in the timer 2 interrupt control register (TM2ICR) to 0 to disable interrupts.
(7) Set the high-order timer interrupt level. TM3ICR (x'03FEC') bits 7:6—TM2LV[1:0]=10	(7) Set the interrupt level using the TM3LV[1:0] field in the timer 3 interrupt control register (TM3ICR). If the interrupt request flag might have already been set, clear the request flag. See section 3.5, “Setting the Interrupt Flags.”
(8) Enable high-order timer interrupts. TM3ICR (x'03FEC') bit 1—TM3IE=1	(8) Enable interrupts by setting the TM3IE flag of the TM3ICR register to 1.
(9) Start the high-order timer. TM3MD (x'03F5D') bit 3—TM3EN=1	(9) Set the TM3EN flag of the TM3MD register to 1 to start timer 3.

Table 6-5 Procedures for Setting up a Cascade Connection (Continued)

Procedure	Description
(10) Start the low-order timer. TM2MD (x'03F5C') bit 3—TM2EN=1	(10) Set the TM2EN flag of the TM2MD register to 1 to start timer 2.

TM3BC and TM2BC start counting from x'0000' as a 16-bit timer. When TM3BC and TM2BC match the value set in the TM3OC and TM2OC registers, the timer 1 interrupt request flag is set at the next count clock and TM3BC and TM2BC start counting up again from x'0000'.

Use 16-bit access instructions when working with the settings for the combined register (TM3OC and TM2OC).

Start the high-order timer before the low-order timer.

6.4 8-Bit Timer Control Registers

Timers 0 through 4 are each composed of a binary counter (TMnBC) and a compare register (TMnOC). They are controlled by mode registers (TMnMD). If you are selecting the count clock sources for timers 0 through 4 by selecting a prescalar output, you must use the prescalar control register (PSCMD) and a prescalar select register (CKnMD) for control. Control the IR remote signal receiver with the IR remote signal receiver carrier output control register (RMCTR).

6.4.1 Control Registers

Table 6-6 lists the registers that control timers 2 through 4.

Table 6-6 8-Bit Timer Control Registers

Register		Address	R/W	Description
Timer 2	TM2BC	x'03F50'	R	Timer 2 binary counter
	TM2OC	x'03F52'	R/W	Timer 2 compare register
	TM2MD	x'03F54'	R/W	Timer 2 mode register
	CK2MD	x'03F56'	R/W	Timer 2 prescalar select register
	PSCMD	x'03F6F'	R/W	Prescalar control register
	TM2ICR	x'03FE9'	R/W	Timer 2 interrupt control register
Timer 3	TM3BC	x'03F51'	R	Timer 3 binary counter
	TM3OC	x'03F53'	R/W	Timer 3 compare register
	TM3MD	x'03F55'	R/W	Timer 3 mode register
	CK3MD	x'03F57'	R/W	Timer 3 prescalar select register
	PSCMD	x'03F6F'	R/W	Prescalar control register
	TM3ICR	x'03FEA'	R/W	Timer 3 interrupt control register
Timer 4	TM4BC	x'03F58'	R	Timer 4 binary counter
	TM4OC	x'03F5A'	R/W	Timer 4 compare register
	TM4MD	x'03F5C'	R/W	Timer 4 mode register
	CK4MD	x'03F5E'	R/W	Timer 4 prescalar select register
	PSCMD	x'03F6F'	R/W	Prescalar control register
	TM4ICR	x'03FEB'	R/W	Timer 4 interrupt control register

6.4.2 Programmable Timer Registers

Timers 0 through 4 all have 8-bit programmable timer registers. The programmable timer registers are composed of compare registers and binary counters. The compare registers are 8-bit registers that hold values to be compared to the binary counter. The binary counters are 8-bit up-counters. Binary counters are cleared to 00 when compare registers are written to during counting.

TM2OC: Timer 2 Compare Register **x'03F5A'**

Bit:	7	6	5	4	3	2	1	0
	TM2 OC7	TM2 OC6	TM2 OC5	TM2 OC4	TM2 OC3	TM2 OC2	TM2 OC1	TM2 OC0
Reset:	X	X	X	X	X	X	X	X
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

TM3OC: Timer 3 Compare Register **x'03F5B'**

Bit:	7	6	5	4	3	2	1	0
	TM3 OC7	TM3 OC6	TM3 OC5	TM3 OC4	TM3 OC3	TM3 OC2	TM3 OC1	TM3 OC0
Reset:	X	X	X	X	X	X	X	X
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

TM4OC: Timer 4 Compare Register **x'03F62'**

Bit:	7	6	5	4	3	2	1	0
	TM4 OC7	TM4 OC6	TM4 OC5	TM4 OC4	TM4 OC3	TM4 OC2	TM4 OC1	TM4 OC0
Reset:	X	X	X	X	X	X	X	X
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

TM2BC: Timer 2 Binary Counter **x'03F58'**

Bit:	7	6	5	4	3	2	1	0
	TM2 BC7	TM2 BC6	TM2 BC5	TM2 BC4	TM2 BC3	TM2 BC2	TM2 BC1	TM2 BC0
Reset:	X	X	X	X	X	X	X	X
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

TM3BC: Timer 3 Binary Counter **x'03F59'**

Bit:	7	6	5	4	3	2	1	0
	TM3 BC7	TM3 BC6	TM3 BC5	TM3 BC4	TM3 BC3	TM3 BC2	TM3 BC1	TM3 BC0
Reset:	X	X	X	X	X	X	X	X
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

TM4BC: Timer 4 Binary Counter **x'03F60'**

Bit:	7	6	5	4	3	2	1	0
	TM4 BC7	TM4 BC6	TM4 BC5	TM4 BC4	TM4 BC3	TM4 BC2	TM4 BC1	TM4 BC0
Reset:	X	X	X	X	X	X	X	X
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

6.4.3 Timer Mode Registers

The timer mode registers are read/write registers that control timers 2 through 4.

TM2MD: Timer 2 Mode Register **x'03F5C'**

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	TM2EN	—	TM2 CK1	TM2 CK0
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R/W	R	R/W	R/W

TM2EN: Timer 2 count control

0: Stop count

1: Count

TM2CK[1:0]: Clock source select

00: f_{OSC} 10: f_X

01: tm2spc (prescalar output) 11: Disabled

TM3MD: Timer 3 Mode Register **x'03F5D'**

Bit:	7	6	5	4	3	2	1	0
	—	—	—	TM3 CAS	TM3EN	—	TM3 CK1	TM3 CK0
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R/W	R	R/W	R/W

TM3CAS: Timer 3 operating mode select

0: Normal counting

1: Cascade connection

TM3EN: Timer 3 count control

0: Stop count

1: Count

TM3CK[1:0]: Clock source select

00: f_{OSC} 10: f_X

01: tm3spc (prescalar output) 11: Disabled

TM4MD: Timer 4 Mode Register **x'03F64'**

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	TM4EN	—	TM4 CK1	TM4 CK0
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R/W	R	R/W	R/W

TM4EN: Timer 4 count control

0: Stop count

1: Count

TM4CK[1:0]: Clock source select

00: f_{OSC} 10: f_X

01: tm4spc (prescalar output) 11: Disabled

7 On-Screen Display

7.1 Description

Table 7-1 shows the OSD functions of the MN101C46F. OSD allows you to display on-screen text characters and graphic tiles of the same size (16 dots × 18 lines) on the same line in any order. You can also modify the ROM space that contains the text characters and the graphic tiles using register settings. This allows you to adjust the memory space to fit your application.

Table 7-1 OSD Functions and Features

Function/Feature	Text Characters	Graphics Tiles
Characters or tiles per line (1) (on a single screen)	Maximum total number of text characters and graphic tiles on the line holding the most characters on the screen: 61 codes.	
	60 characters per line maximum(1)	61 tiles per line
RAM usage	128 bytes per line maximum Line-by-line basis (2048 bytes, 16 lines vertically) Maximum 64 lines	128 bytes per line maximum Line-by-line basis (2048 bytes, 16 lines vertically) Maximum 64 lines
ROM usage	36 bytes per character (9 KB, 256 character types)	8 colors(2): 108 bytes per tile (7 KB, 64 tile types)
Resolution	16 (wide) × 18 (high) pixels *In closed-caption mode:* 16 × 26 (underlining is in the hardware)	16 (wide) × 18 (high) pixels
Color depth(3)	8 of 27 colors (four 8-color palettes) *In closed-caption mode:* 16 of 27 colors (two 16-color palettes)	8 of 27 colors (four 8-color palettes) (Up to 27 colors in one display)
Display position	H: 1 dot resolution, 1024 steps(4) V: 1 H scan line resolution, 1024 steps	H: 1 dot resolution, 1024 steps V: 1 H scan line resolution, 1024 steps
Display size	Four vertical types and four horizontal types on a line-by-line basis, total 16 types(4)	Four vertical types and four horizontal types on a line-by-line basis, total 16 types
Display functions	• Shutter effect • Outlining • Blinking • Shadowing (foreground and background) *In closed-caption mode:* • Italics • Underlining	• Repeated tile or blank (5)

Notes:
1. Maximum 61 characters per line with the default colors. Each color assignment, including outlining and blinking, decreases this total by one. (The maximum is about 44 characters in NTSC interlacing when HSYNC = 63.5 µs.)
2. 8-color mode: 108 bytes per tile (7 KB, 64 tile types)
   4-color mode: 72 bytes per tile (4.5 KB, 64 tile types)
   Multiple modes cannot be used simultaneously; the color mode applies to the entire display.
3. R, G, and B can each be set in three levels giving a total of 27 available display colors (3 × 3 × 3). Selects any eight of these color combinations for one color palette. Intermediate grades for R, G, and B will cause Hi-Z output. Implement intermediate voltages using external resistors.
4. The OSD dot clock frequency controls the horizontal position and size. For details, see section 7.7.4, “Setting Up the OSD Display Position,” on page 134, and section 7.9, “Selecting the OSD Dot Clock,” on page 136.
5. This function can be used for a wallpapering effect or to insert spaces for continuous blanking. One tile code can be repeated up to 16 times. Repeating tiles allows you to use more than 62 tiles per line.

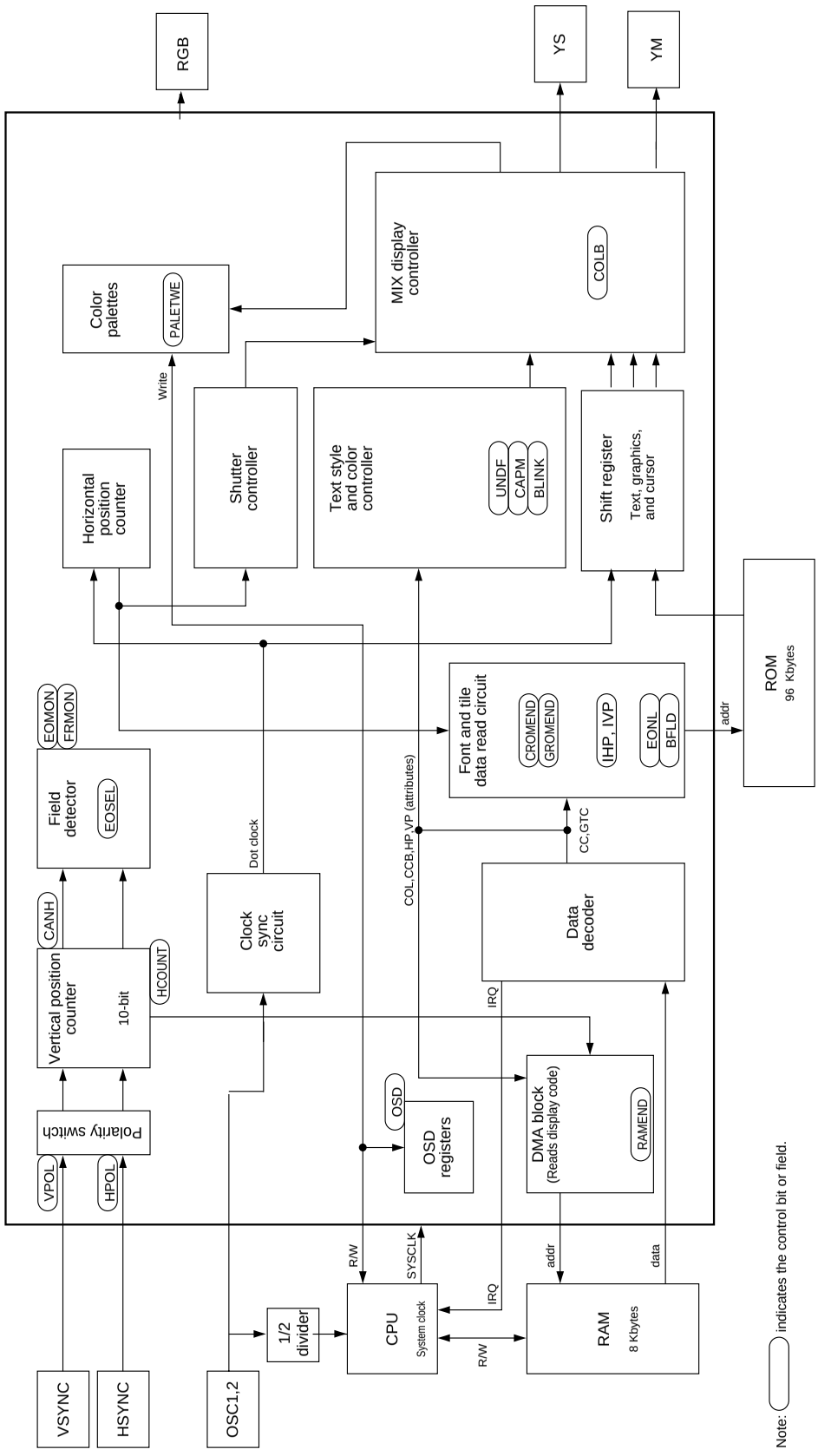


Figure 7-1 OSD Block Diagram

7.2 Power-Saving Considerations in the OSD Block

Table 7-2 shows the two control bits that can decrease the power consumption of the OSD block. This section explains how to use these bits.

Table 7-2 Power-Saving Control Bits for the OSD

Bit Name	Register	Address	Bit	Description	Reset
OSDPOFF	PCNT0	x'03F4A'	7	0: System clock off to OSD 1: System clock on to OSD	0
OSDREGE	PCNT2	x'03F4E'	2	0: R/W disabled for OSD registers 1: R/W enabled for OSD registers	0

■ **Using OSDPOFF to control the system clock supply to the OSD**

The OSDPOFF bit enables or disables the system clock supply to the OSD block. When the OSD is unused, setting this bit to 0 stops the clock supply to the OSD, reducing power dissipation. Setting OSDPOFF to 0 not only disables the OSD display, it disables reads from and writes to the OSD registers. To operate the OSD, set this bit to 1, then set up the OSD registers.

■ **Using OSDREGE to control read/write access to the OSD registers**

The OSDREGE bit enables or disables read/write operations to the OSD registers. (See section 7.12, "OSD Registers," on page 146.) Once you have set the OSD registers, you can write a 0 to this bit to disable furthers reads and writes to them, reducing power dissipation. This bit resets to 0.

Note that when OSDPOFF is 0, you cannot set the OSD registers even if OSDREGE is 1. Table 7-3 shows the combinations of OSDPOFF and OSDREGE settings. Note also that when OSDREGE is 0, the OSD display runs, but the shutter cannot be moved. If your application requires shutter movement, you must enable OSDREGE.

Table 7-3 OSDPOFF and OSDREGE Settings

OSDPOFF	OSDREGE	OSD	Register R/W	Power Dissipation
0	Don't care	Off	Disabled	Less ↕ Greater
1	0	On	Disabled	
1	1	On	Enabled	

7.3 OSD Operation

This section describes the basic operation of the OSD block. Also see the descriptions of register assignments and VRAM assignments for details. Figure 7-1 shows a block diagram of the OSD circuit.

7.3.1 Operating Clock

The source clocks for OSD operation are the external oscillator pins OSC1 and OSC2. (The frequency range is 12 to 14.32 MHz.)

See section 7.9, “Selecting the OSD Dot Clock,” on page 136 for more information on setting the operating clock frequency.

7.3.2 External Input Synchronization Signal

Input a horizontal synchronization signal HSYNC and a vertical synchronization signal VSYNC. Use software to select whether there are any pull-up resistors and to select polarity. For VSYNC, assign an IRQ so the microcontroller can detect the start of the field. Note that REDG5 selects the interrupt edge and VPOL selects the OSD input.

7.3.3 Display Control System

The OSD is able to display text fonts and graphic tiles in the same line.

7.3.3.1 Text Fonts

These are used to display text. Each individual character has a text color and a background color for its display. Colors for outlines and shading are set separately. There is a special closed-caption mode for displaying closed-captions. You cannot use normal character display while you are using closed-caption mode. Also, in using closed-caption mode, characters can be lined up with graphic tiles, but graphic tiles on characters displayed as italics will be shifted by a pixel.

7.3.3.2 Graphic Layer

Do not include graphic tiles in italic display lines in the closed-caption load. Graphic tiles will be shifted by a pixel.

The graphic layer is used primarily for graphics. Eight colors can be displayed in pixel units for one tile (8-color mode). There are four sets of color palettes, so a single tile can be displayed with 4 different colors on one screen by switching the color palette for individual tiles. There is also a 4-color mode. The color mode must be switched per screen, so all tiles displayed on a single screen must be the same mode. (8-color mode tiles and 4-color mode tiles cannot be displayed on the same screen.) The resolution of graphic tiles is 16 dots × 18 dots.

7.3.4 Output Pin Setup

Select OSD or port for the output pins. RGB, YS, and YM are digital outputs. Set the YS polarity.

7.3.5 Microcontroller Interface

The microcontroller writes display data to be sent to the OSD in the control register and the VRAM, which is assigned to internal RAM space. Assign the VRAM by setting its end address in the control register RAMEND.

7.3.6 Basic VRAM Operation

After a reset clears, the system clock supply to the OSD stops. To operate the OSD, you must first set OSDPOFF (bit 7 of PCNT0 (x'03F4A')) to 1.

Display data stored in the VRAM transfers automatically (through a DMA transfer) from the internal RAM to the OSD as the display approaches the position specified by the microcontroller. Since the OSD will have the internal bus during this transfer, the microcontroller stops. See section 7.8, "DMA and Interrupt Timing," on page 135, for more information.

The two MSBs identify the transferred data with the following ID codes:

1.	Display code	CC
2.	Color control code (normal mode)	COL
3.	Color control code (closed-caption mode)	COL
4.	Repeat character/blank code	CB
5.	Horizontal position code	HP
6.	Vertical position code	VP

7.3.7 Conditions for VRAM Writes

1. The lead data for each line must be the color control code (COL) or the character code (CC). Never place the horizontal position (HP), vertical position (VP), RAM address pointer (AP), or repeat (CB) codes at the beginning of a line. (If the lead data is CC, the character will be palette color 1, and the background will be color 2.)

2. Always place data in the following order at the end of a line.
   APCNT = 0: Horizontal position (HP), vertical position (VP)
   APCNT = 1: Horizontal position (HP), vertical position (VP), address pointer (AP)

You do not need to meet condition 3 in the closed-caption mode, since COL carries over.

3. Insert the color control code (COL) before the character code (CC).

4. A character code (CC) must immediately precede a repeat character/blank code (CB), and a color control code (COL) or character code (CC) must follow it.

5. To indicate the last line of a display, make the VP value for the last line smaller than that in the currently displayed line. In addition, write a 1 to the VP's last-line flag (LAST).

6. Lines cannot overlap.

7. If the horizontal sync signal is asserted while the microcontroller is accessing HP and VP, no more lines will be displayed.

7.4 Display Setup Examples

7.4.1 Setting Up the Display without AP

This section shows how to set up the display data in the VRAM without AP.

■ **Register settings**

RAMEND (x'003EB4') = x'BF' (VRAM end address: x'0BFF')

IHPH (x'003ECB') = x'08' (IHP = x'22', IHSZ = x'1')

IHP (x'003ECA') = x'22' (IHP = x'22', IHSZ = x'1')

IVPH (x'003ECD') = x'18' (IVP = x'03', IVSZ = x'3')

IVP (x'003ECC') = x'03' (IVP = x'03', IVSZ = x'3')

OSD2 (x'003EB8') = x'00' (APCNT = 0, 8-color graphics mode, no graphics output)

Table 7-4 Example Graphics VRAM Settings

Line No.	RAM Addr.	RAM Data	Data Type	Description
1	0BFE	2802	CC	CG=1, PLT=1, CH=x'002'
	0BFC	1801	CC	CG=0, PLT=3, CH=x'001'
	0BFA	C004	HP	HSZ=x'0', SHT=0, HP=x'04'
	0BF8	C040	VP	LAST=0, VSZ=x'0', INT=0, VP=x'40'
	...	...		
2	0BAE	0810	CC	CG=0, PLT=1, CH=x'010'
	0BAC	2813	CC	CG=1, PLT=1, CH=x'013'
	0BAA	4003	CB	BF=1, CB=x'3'
	0BA8	1014	CC	CG=0, PLT=2, CH=x'014'
	0BA6	4002	CB	BF=1, CB=x'2'
	0BA4	3016	CC	CG=1, PLT=2, CH=x'016'
	0BA2	D810	HP	HSZ=x'3', SHT=0, HP=x'10'
	0BA0	C858	VP	LAST=0, VSZ=x'1', INT=0, VP=x'58'
	...	...		
3	0B5E	2981	CC	CG=1, PLT=1, CH=x'181'
	0B5C	3182	CC	CG=1, PLT=2, CH=x'182'
	0B5A	C044	HP	HSZ=x'0', SHT=0, HP=x'44'
	0B58	E020	VP	LAST=1, VSZ=x'0', INT=0, VP=x'20'
	...	...		

Notes: 1. Always specify HP and VP, in that order, at the end of each line.

2. Set INT to 1 in the VP setting to generate an OSD interrupt.

3. Set LAST to 1 in the VP setting for the last line in the display. Also, set the VP value to a smaller value than the position of the current line. (In the example in table 7-4, VP = x'20' is smaller than VP = x'58'.)

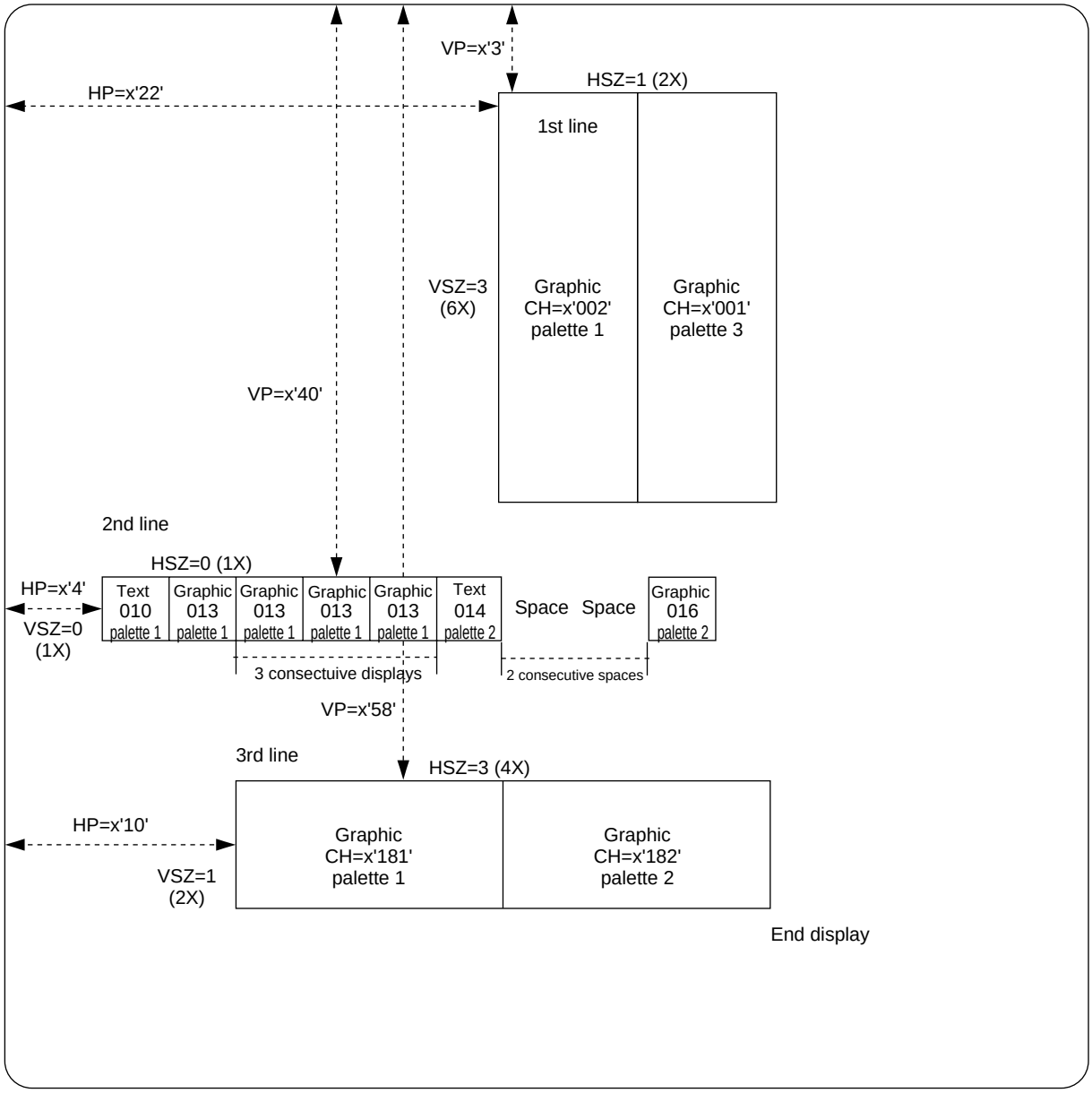


Figure 7-2 Display Example without AP

7.4.2 Setting Up the Display with AP

This section shows how to set up the display data in the VRAM with AP.

■ **Register settings**

RAMEND (x'003EB4') = x'BF' (VRAM end address: x'0BFF')

IAPH (x'003ECF') = x'0B' (IAP = x'BFF')

IAP (x'003ECE') = x'FF' (IAP = x'BFF')

IHPH (x'003ECB') = x'08' (IHP = x'22', IHSZ = x'1')

IHP (x'003ECA') = x'22' (IHP = x'22', IHSZ = x'1')

IVPH (x'003ECD') = x'18' (IVP = x'03', IVSZ = x'3')

IVP (x'003ECC') = x'03' (IVP = x'03', IVSZ = x'3')

OSD2 (x'003EB8') = x'00' (APCNT = 1, 8-color graphics mode, no graphics output)

Table 7-5 Example Text VRAM Settings

Line No.	RAM Addr.	RAM Data	Data Type	Description
1	0BFE	2802	CC	CG=1, PLT=1, CH=x'002'
	0BFC	1801	CC	CG=0, PLT=3, CH=x'001'
	0BFA	C004	HP	HSZ=x'0', SHT=0, HP=x'04'
	0BF8	C040	VP	LAST=0, VSZ=x'0', INT=0, VP=x'40'
	0BF6	CBF4	AP	AP=x'BF4'
2	0BF4	0810	CC	CG=0, PLT=1, CH=x'010'
	0BF2	2813	CC	CG=1, PLT=1, CH=x'013'
	0BF0	4003	CB	BF=1, CB=x'3'
	0BEE	1014	CC	CG=0, PLT=2, CH=x'014'
	0BEC	4002	CB	BF=0, CB=x'2'
	0BEA	3016	CC	CG=1, PLT=2, CH=x'016'
	0BE8	D810	HP	HSZ=x'3', SHT=0, HP=x'10'
	0BE6	C858	VP	LAST=0, VSZ=x'1', INT=0, VP=x'58'
	0BE4	CBE2	AP	AP=x'BE2'
3	0BE2	2981	CC	CG=1, PLT=1, CH=x'181'
	0BE0	3182	CC	CG=1, PLT=2, CH=x'182'
	0BDE	C044	HP	HSZ=x'0', SHT=0, HP=x'44'
	0BDC	E020	VP	LAST=1, VSZ=x'0', INT=0, VP=x'20'
	0BDA	CBFE	AP	AP=X'BFE'

Notes: 1. Always specify HP, VP, and AP, in that order, at the end of each line.

2. Set INT to 1 in the VP setting to generate an OSD interrupt.

3. Set LAST to 1 in the VP setting for the last line in the display. Also, set the VP value to a smaller value than the position of the current line. (In the example in table 7-4, VP = x'20' is smaller than VP = x'58'.)

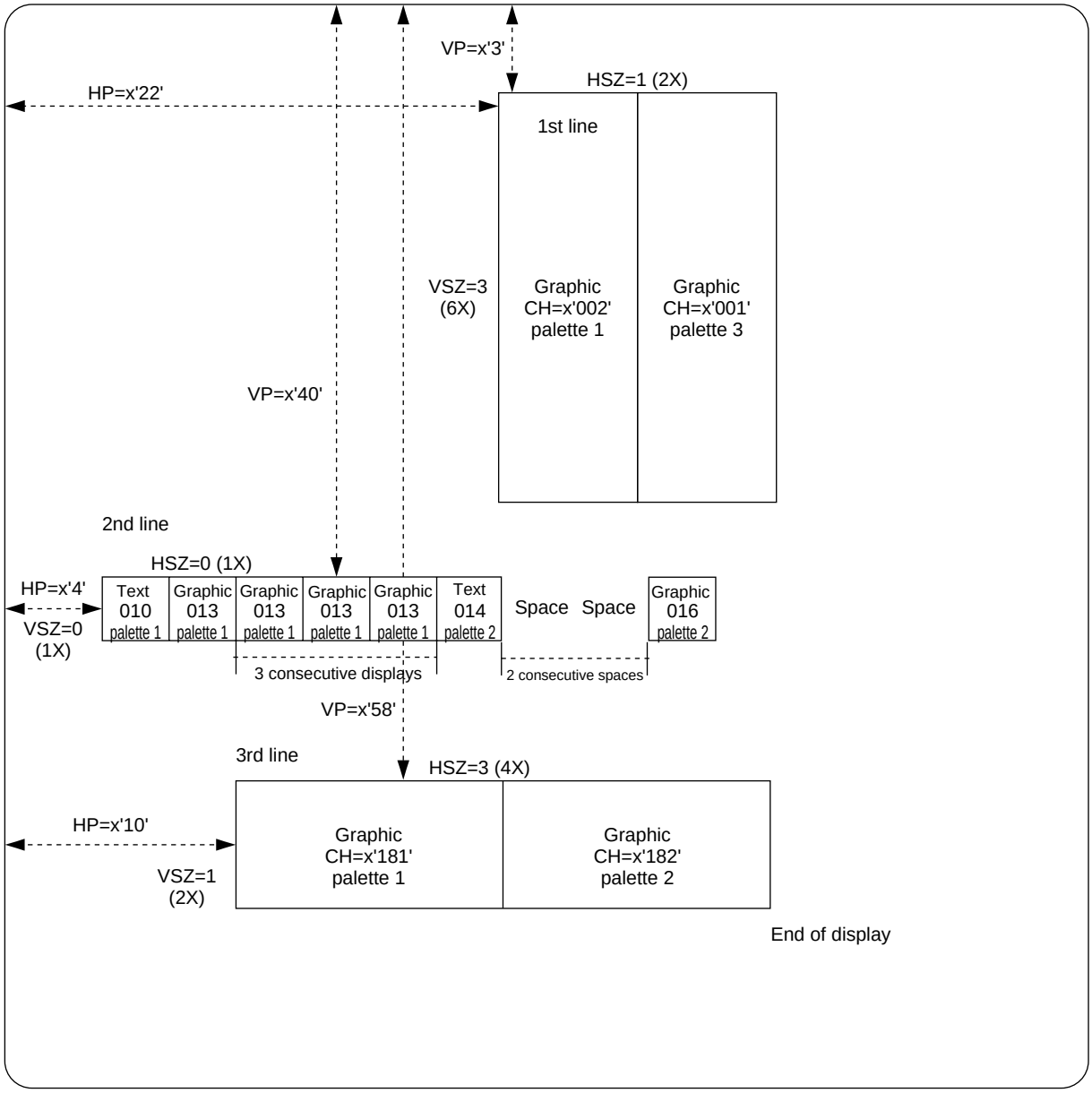


Figure 7-3 Display Example with AP

7.5 VRAM

7.5.1 VRAM Bit Assignments in Internal RAM

Table 7-6 VRAM Bit Assignment

Bits (2 linked odd/ even bytes)	15 (odd. bp7)	14 (odd. bp6)	13 (odd. bp5)	12 (odd. bp4)	11 (odd. bp3)	10 (odd. bp2)	9 (odd. bp1)	8 (odd. bp0)	7 (even. bp7)	6 (even. bp6)	5 (even. bp5)	4 (even. bp4)	3 (even. bp3)	2 (even. bp2)	1 (even. bp1)	0 (even. bp0)
CC	0	0	CGSEL	PLT1	PLT0	*	*	CH8	CH7	CH6	CH5	CH4	CH3	CH2	CH1	CH0
Text code	ID code		C/G select	Palette select				Text character/graphic tile address (512 of each)								
COL (normal mode)	1	0	*	BSHAD1	BSHAD0	CSHAD	FRAME	BLINK	*	BCOL2	BCOL1	BCOL0	*	CCOL2	CCOL1	CCOL0
Character color control code	ID code			Box shadow		Char. shadow	Outline	Blink		Background color select				Text color select		
COL (closed-caption mode)	1	0	*	*	CUNDL	ITALIC	FRAME	BLINK	BCOL3	BCOL2	BCOL1	BCOL0	CCOL3	CCOL2	CCOL1	CCOL0
Character color control code	ID code				Underline	Italics	Outline	Blink	Background color (select)				Character color (select)			
CB	0	1	*	*	*	*	*	*	*	*	*	BF	CB3	CB2	CB1	CB0
Repeat character/blank code	ID code											Blank/ char.	Number of blank/char. repetitions (4 bits)			
HP	1	1	*	HSZ1	HSZ0	SHT	HP9	HP8	HP7	HP6	HP5	HP4	HP3	HP2	HP1	HP0
H position control	ID code			H text size		Shutter	H display start position (1 dot resolution, 1024 steps, start position set in register IHP)									
VP	1	1	LAST	VSZ1	VSZ0	INT	VP9	VP8	VP7	VP6	VP5	VP4	VP3	VP2	VP1	VP0
V position control	ID code		Last line	V text size		Interrupt	V display start position (1H scan line resolution, 1024 steps, start position set in register IVP)									
AP	1	1	*	*	AP11	AP10	AP9	AP8	AP7	AP6	AP5	AP4	AP3	AP2	AP1	AP0
RAM address pointer	ID code			VRAM start code address position for next line (x'00100'- x'00BFF': Specify internal RAM address. Set initial address in the IAP register.)												

* Don't-care bits

7.5.2 VRAM Operation

CC: Character Code — ID Code: 00

CGSEL

Selects text or graphic

0: Text

1: Graphic

PLT[1:0]

Select the palette (PLT0x-PLT3x).

CCH[8:0]

Specify the address of the text characters or graphic tile stored in ROM (512 types each).

COL: Color Control Code, Normal Mode — ID Code: 10

BSHAD[1:0]

Specify shadowing of the character box for a 3D button effect.

00: Disable

01: Disable

10: Upper left white and lower right black shadows

11: Upper left black and lower right white shadows

CSHAD

Specifies character shadowing for a 3D effect.

0: Disable

1: Enable

FRAME

Specifies character outlining (black).

0: Disable
1: Enable

BLINK

Specifies character blinking.

0: Disable
1: Enable

BCOL[2:0]

Specify the background color (1 of 8 colors).

CCOL[2:0]

Specify the foreground (text) color (1 of 8 colors).

COL: Color Control Code, Closed-Caption Mode ID Code: 10

CUNDL

Specifies underlining.

0: Disable
1: Enable

ITALIC

Specifies italicization.

0: Disable
1: Enable

FRAME

Specifies character outlining (black).

0: Disable
1: Enable

BLINK

Specifies character blinking.

0: Disable
1: Enable

BCOL[3:0]

Specifies the background color (1 of 16 colors).

CCOL[4:0]

Specifies the foreground (text) color (1 of 16 colors).

CB: Repeat Blank/Character Code ID Code: 01

BF

Repeat blank/repeat character select.

0: Repeat blank
1: Repeat character

CB[3:0]

This field specifies the number of times (up to 16) a blank space or character is repeated. This function saves RAM space by preventing the VRAM address from incrementing. The program redisplays the preceding character code the specified number of times. This increases the limit beyond 61 characters per line.

HP: Horizontal Position Control Code ID Code: 11

HSZ[1:0]

This field specifies the H size of the display code on the next line.

00: 1 dot = 1 VCLK period
01: 1 dot = 2 VCLK periods
10: 1 dot = 3 VCLK periods
11: 1 dot = 4 VCLK periods

SHT

Specifies shutter operation for the next line. Setting this bit to 1 disables the shuttering function. You can disable and enable shuttering on a line-by-line basis.

0: Enable
1: Disable

HP[9:0]

This field specifies a VCLK indicating the horizontal start position for the next line. 1024 steps are available.

VP: Vertical Position Control Code ID Code: 11

LAST

Specifies the last line in the display. This resets the line pointer for character reads from the internal RAM to the first line.

0: Disable
1: Enable

VSZ[1:0]

This field specifies the V size of the display code on the next line.

00: 1 dot = 1 H scan line
01: 1 dot = 2 H scan lines
10: 1 dot = 4 H scan lines
11: 1 dot = 6 H scan lines

INT

Specifies an OSD interrupt.

0: Disable
1: Enable

VP[9:0]

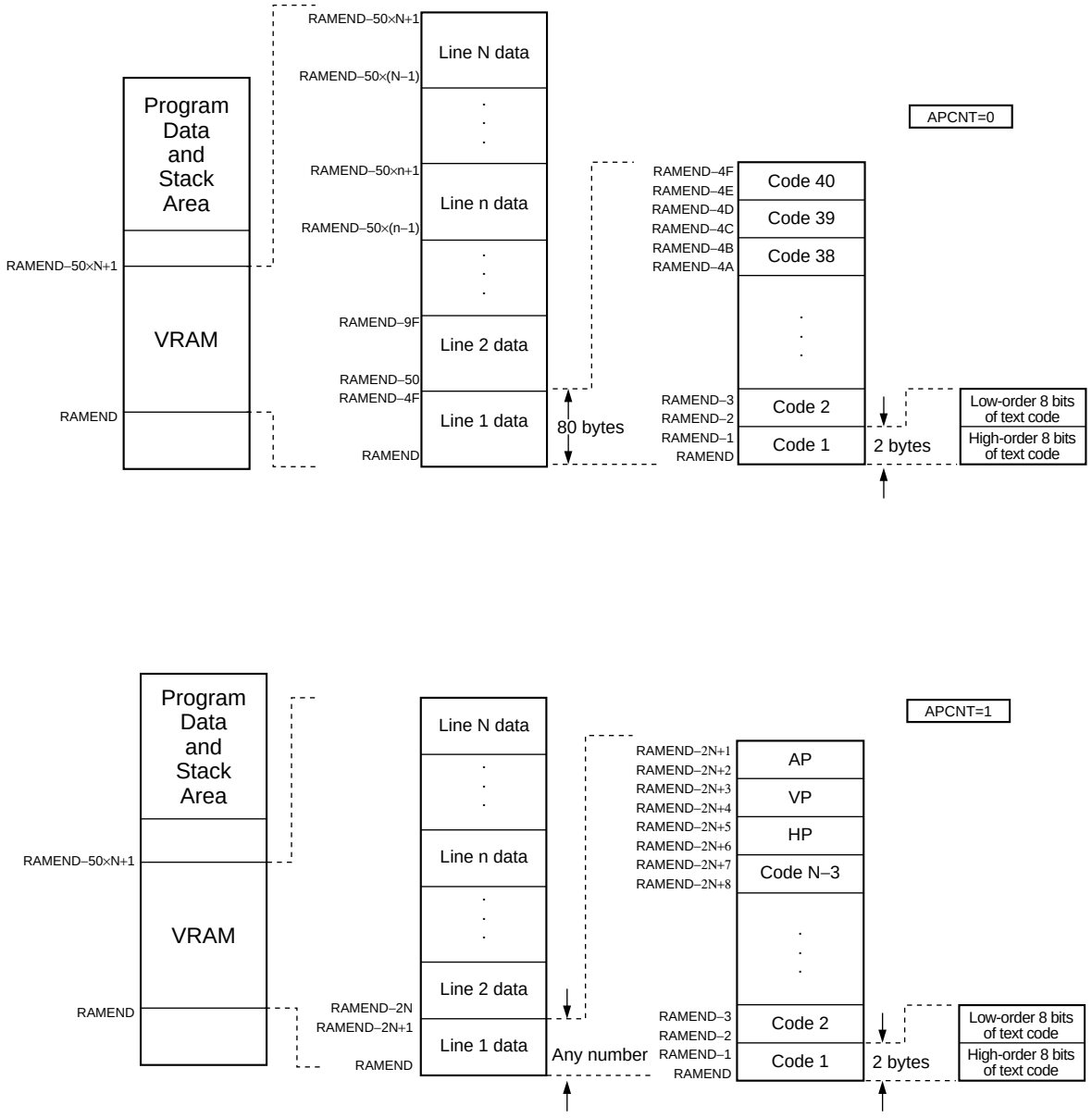
Specifies an H scan line indicating the vertical start position for the next line. 1024 steps are available.

AP: RAM Address Pointer ID Code: 11

AP[11:0]

Specifies the VRAM start code address position of the next line. (Specifies an on-chip RAM address within the 3KB x'0100’-x'0BFF’.) The starting line is specified in the IAP register. (This is valid only when APCNT = 1.)

7.5.3 VRAM Organization

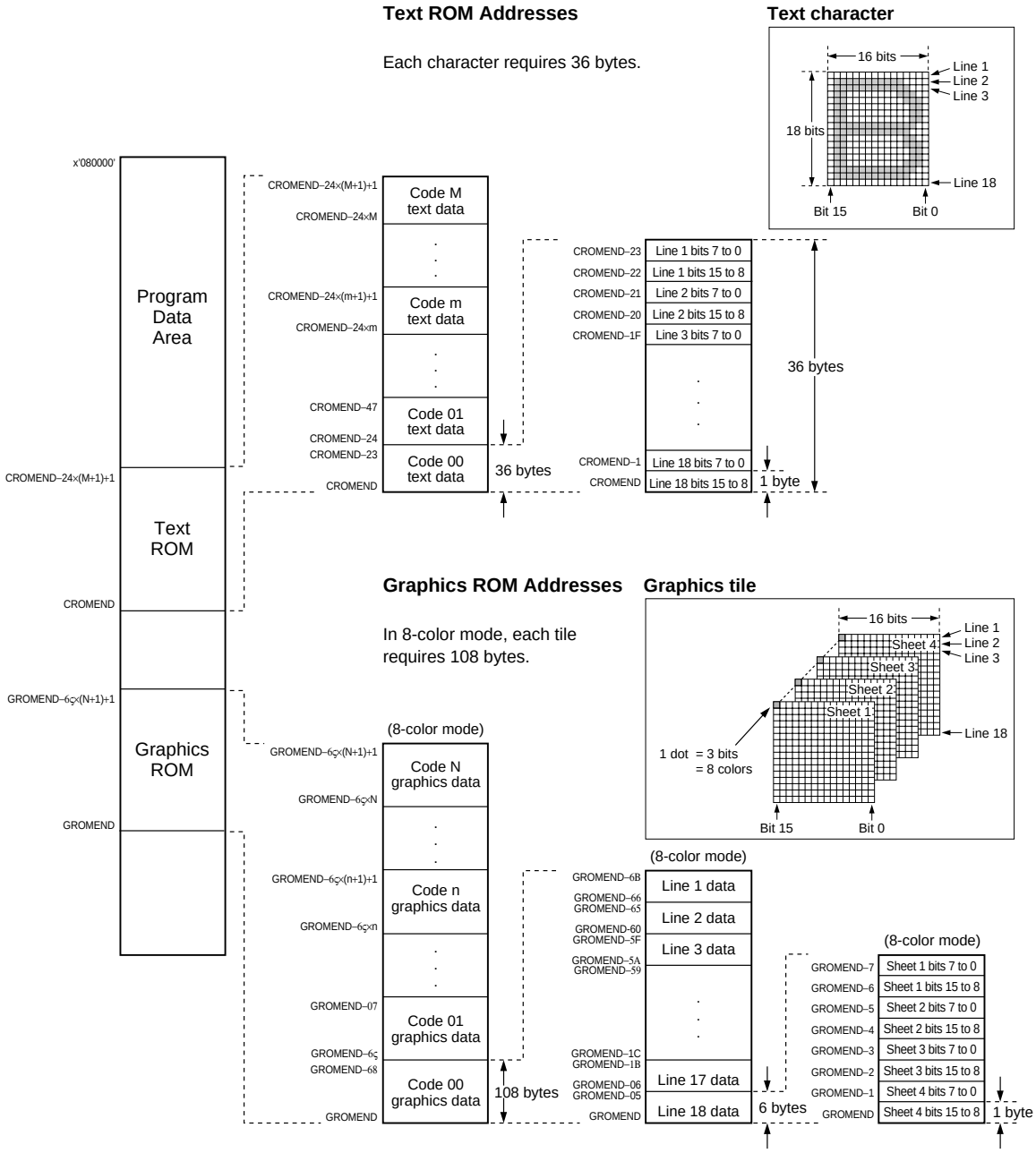


Notes: 1. All addresses are expressed in hex notation. Other values are decimal.
2. RAMEND: RAM end address register (programmable to any address)

Figure 7-4 VRAM Organization

7.6 ROM

7.6.1 ROM Organization



Notes:
1. All addresses are expressed in hex notation. Other values are decimal.
2. GROMEND: Graphics ROM end address register (programmable to any address)
3. CROMEND: Text ROM end address register (programmable to any address)
4. M: Number of text fonts- 1
5. m: 0 and up
6. N: Number of graphic fonts - 1 (8-color mode)
7. n: 0 and up

Figure 7-5 ROM Organization

7.6.2 Graphics ROM Organization in Different Color Modes

The graphics layer supports up to eight colors in the 8-color mode. It also supports a 4-color mode. The smaller the number of colors, the less ROM area required per tile. The figures in this section illustrate the ROM organization for each color mode.

The example in figure 7-6 demonstrates the graphics ROM setup for line 18 of the code 00 data when the graphics layer is in 8-color mode. The three bits of data for each pixel, in sheets 1, 2, and 3, determine the color palette used for that pixel.

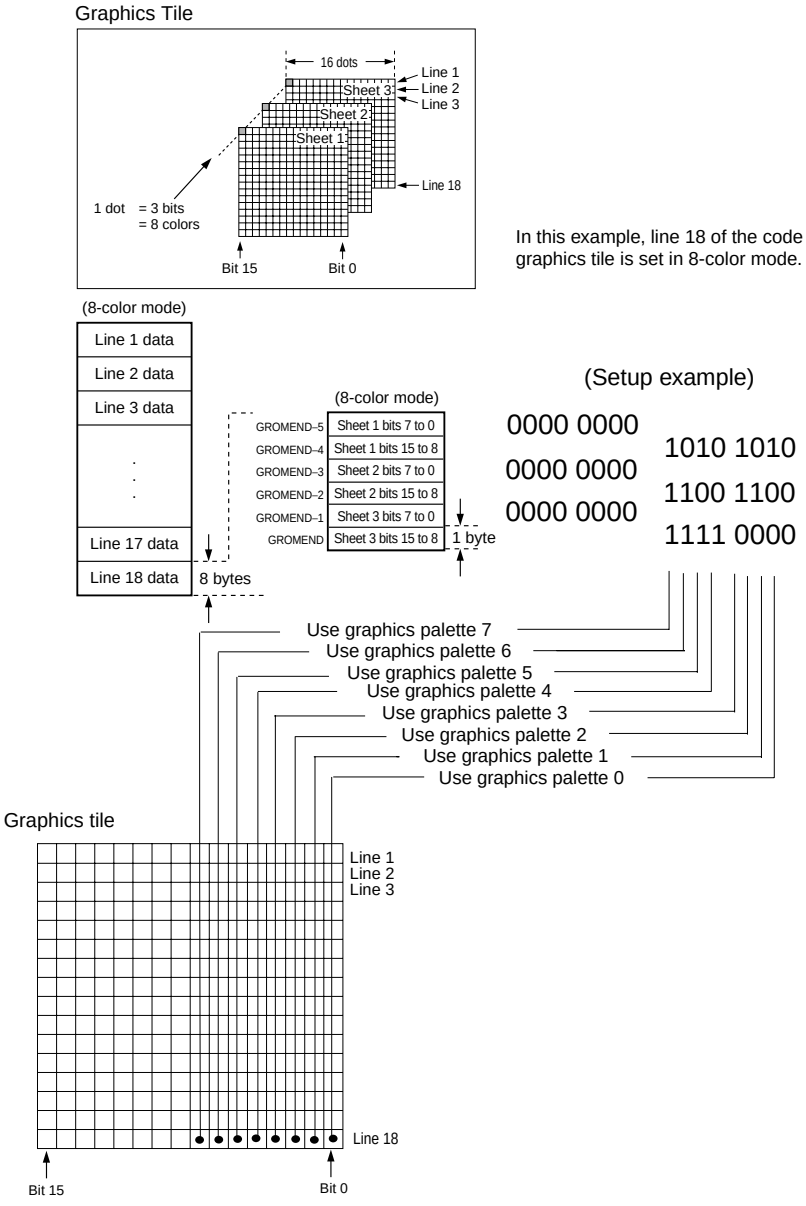


Figure 7-6 Graphics ROM Setup Example for a Single Line

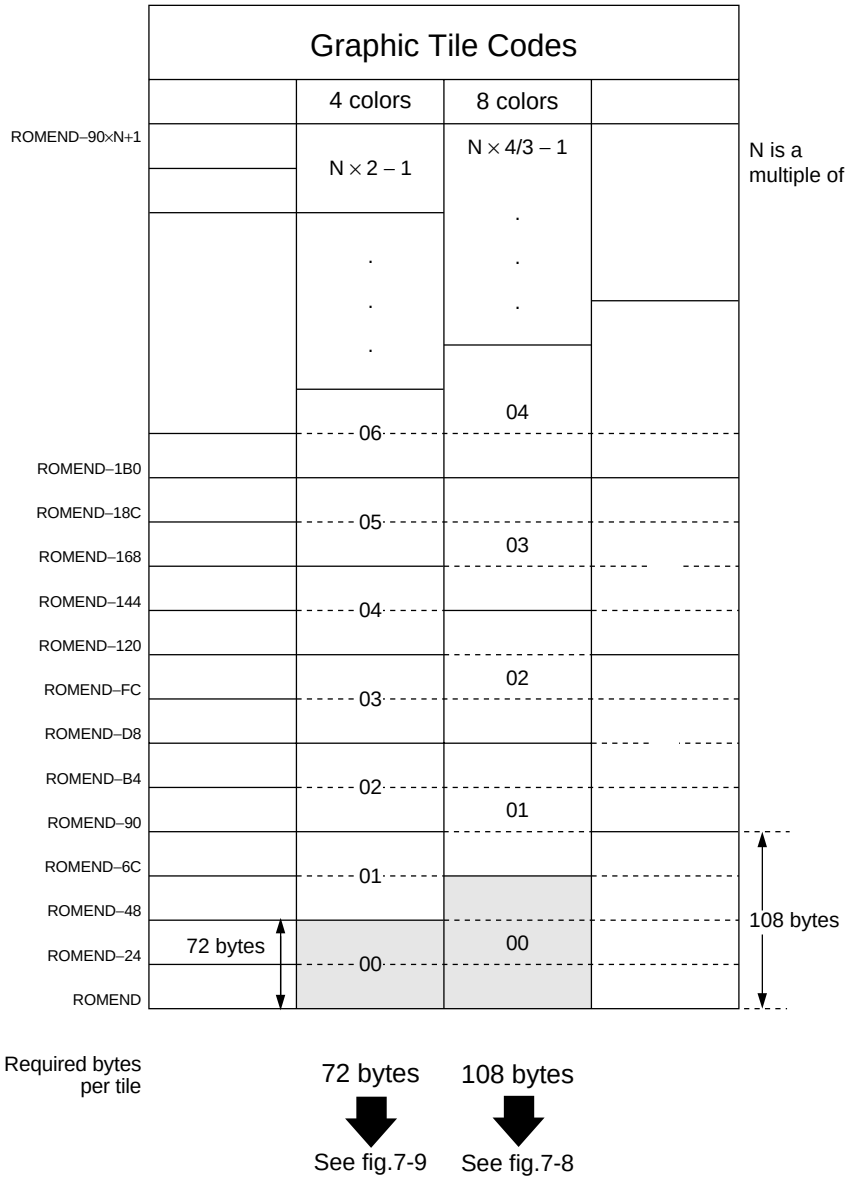


Figure 7-7 Graphics ROM in the Two Color Modes

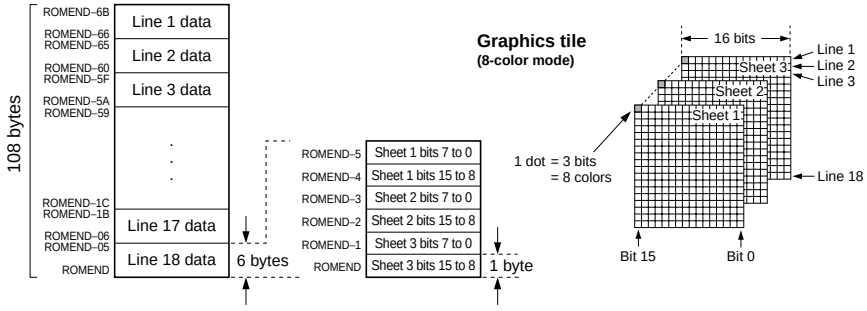


Figure 7-8 Graphics ROM Organization in 8-Color Mode (16W × 18H Tiles)

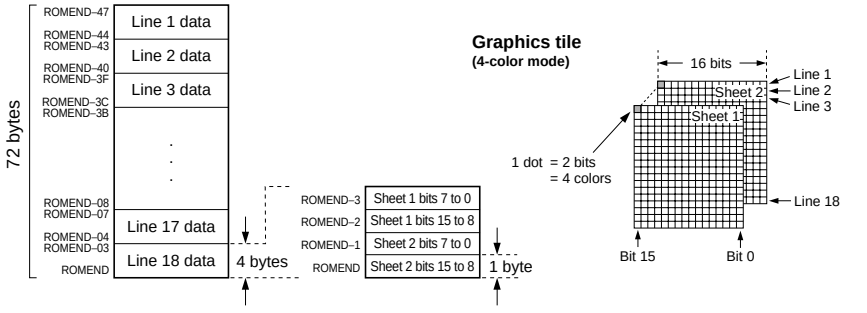


Figure 7-9 Graphics ROM Organization in 4-Color Mode (16W × 18H Tiles)

7.7 Setting up the OSD

7.7.1 Setting OSD Display Colors

This section describes how to set up the display colors for the OSD.

■ **Setting up the color palette**

Table 7-7 Color Palettes

Palette Name	Address	Applications
PLT00-PLT37	x'03EE0'–x'03EFF'	Text foreground and background colors
COLB	x'03ED0'	Color background
FRAME	x'03ED2'	Outlining and character shadowing colors
BBSHD	x'03ED4'	Box shadowing color (black)
WBSHD	x'03ED6'	Box shadowing color (white)

■ **Setting up the graphics display colors**

The following settings determine the graphics display colors:

- **GCOL** (x'03EB8', bit 1) sets the number of colors of the mode (4 or 8).
- **CH[8:0]** (CC bits 8 to 0 in the RAM data) set the code of the tile to be displayed.
- **PLT[1:0]** (CC bits 12 to 11 in the RAM data) select color palette 0, 1, 2, or 3.
- **CGSEL** (CC bit 13 in the RAM data) selects graphic tile display when 1.
- **PLT0x** (x'03EE0'–x'03EE7'), **PLT1x** (x'03EE8'–x'03EEF'), **PLT2x** (x'03EF0'–x'03EF7'), or **PLT3x** (x'03EF0'–x'03EFF') specify the palette color for the tile data stored in ROM.

■ **Setting up the text display colors (normal display)**

The following settings determine the text display colors in normal mode:

- **CAPM** (x'03EBA', bit 1) sets normal display mode when 0.
- **CH[8:0]** (CC bits 8 to 0 in the RAM data) set the code of the font to be displayed.
- **PLT[1:0]** (CC bits 12 to 11 in the RAM data) select color palette 0, 1, 2, or 3.
- **CGSEL** (CC bit 13 in the RAM data) selects text font display when 0.
- **CCOL[2:0]** (COL bits 2 to 0 in the RAM data) set the color of the character (8 colors). This value is in reference to the selected color palette (PLTx0-PLTx7).

- **BCOL[2:0]** (COL bits 6 to 4 in the RAM data) set the background color (8 colors). As with CCOL, this value is in reference to the selected color palette (PLTx0-PLTx7).
- **CSHAD** (COL bit 10 in the RAM data) enables character shadowing when set to 1. Set the shadowing color in the FRAME of the color palette. This function is unavailable in the closed-caption mode.
- **BSHAD[1:0]** (COL bits 12 to 11 in the RAM data) enable character box shadowing when BSHAD1 is 1. This function is unavailable in the closed-caption mode.

  00 and 01: No box shadowing
  10: Upper left white and lower right black shadows
  11: Upper left black and lower right white shadows
- **PLT0x** (x'03EE0'–x'03EE7'), **PLT1x** (x'03EE8'–x'03EEF'), **PLT2x** (x'03EF0'–x'03EF7'), or **PLT3x** (x'03EF0'–x'03EFF') specify the palette color for the font data stored in ROM.
- **BBSHD** (x'03ED4') specifies the "black" color for box shadowing.
- **WBSHD** (x'03ED6') specifies the "white" color for box shadowing.

■ **Setting up the text display colors (closed-caption display)**

The following settings determine the text display colors in closed-caption mode:

- **CAPM** (x'03EBA', bit 0) sets closed-caption display mode when 1.
- **CH[8:0]** (CC bits 8 to 0 in the RAM data) set the code of the font to be displayed.
- **PLT[1]** (CC bit 12 in the RAM data) selects the color palette. 0 selects the 16 colors of PLT00-PLT17 while 1 selects the 16 colors of PLT20-37.
- **CGSEL** (CC bit 13 in the RAM data) selects text font display when 0.
- **CCOL[3:0]** (COL bits 3 to 0 in the RAM data) set the color of the character (16 colors). This value is in reference to the selected color palette (PLT00-PLT17 or PLT20-PLT37).
- **BCOL[3:0]** (COL bits 7 to 4 in the RAM data) set the background color (16 colors). As with CCOL, this value is in reference to the selected color palette (PLT00-PLT17 or PLT20-PLT37).
- **ITALIC** (COL bit 10 in the RAM data) enables character italicization when set to 1.
- **CUNDL** (COL bit 11 in the RAM data) enable character underlining when set to 1.
- **PLT0x** (x'03EE0'–x'03EE7') and **PLT1x** (x'03EE8'–x'03EEF') or **PLT2x** (x'03EF0'–x'03EF7') and **PLT3x** (x'03EF0'–x'03EFF') specify the palette color for the font data stored in ROM.

■ **Setting up functions for all text**

This section describes settings for text display that are used for both normal display and closed-caption display.

- **BLINK** (COL bit 8 in the RAM data) enables character blinking when set to 1.
- **FRAME** (COL bit 9 in the RAM data) enables character outlining when set to 1. Set the outline color using FRAME in the color palette.
- **FRAME** (x'03ED2') specifies the outline color or character shading.

■ **Setting up functions for all the display colors**

Color background function

The color background function allows you to fill the television screen areas that are not covered by the OSD display (text or graphics) with any color.

- **COLB** (x'03ED8', bit 0) enables the color background function when set to 1.
- **COLB** (x'03ED0') specifies the color of the background.

Transparency

- **PLTx0-PLTx7** (bit 6) controls the YS pin output.

Translucency

- **PLTx0-PLTx7** (bit 7) controls the YM pin output.

See figure 7-10.

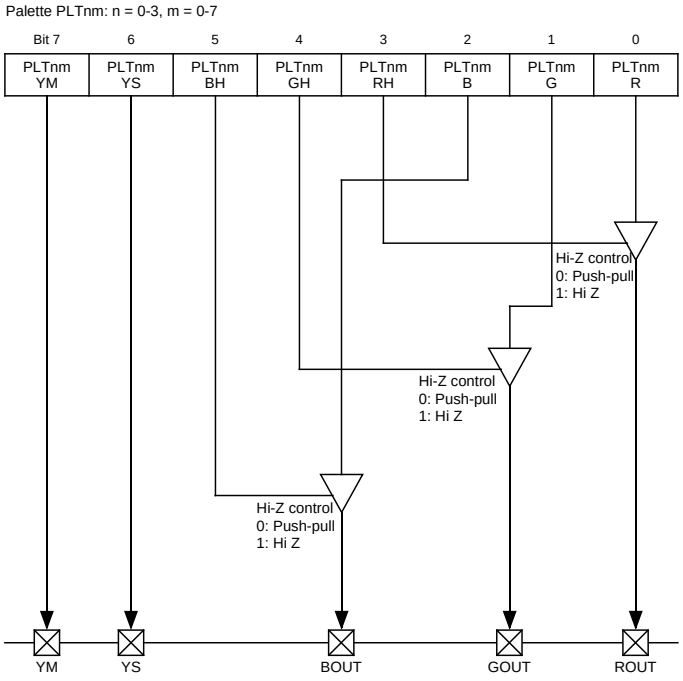


Figure 7-10 OSD Signal Output Control

7.7.2 Text Layer Functions

This section describes the character enhancement functions available in the text layer.

■ **Outlining**

In both normal and closed-caption modes, writing a 1 to bit 9 (FRAME) of the VRAM's COL field causes outlines to appear around all characters following that COL. You can specify the color of the outline in the color palette's FRAME (x'03ED2'). Figure 7-11 shows an example of character outlining. As shown in the figure, if a character contains dots in the left or right borders of its field, the outlining for those dots appear in the adjacent character field.

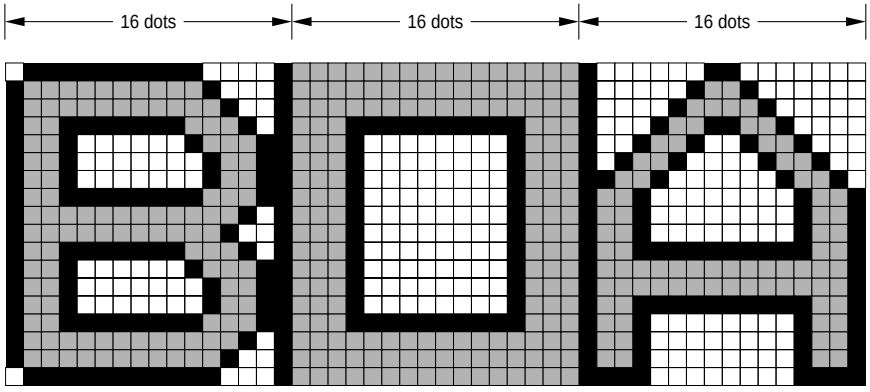


Figure 7-11 Character Outlining Example

■ **Character shadowing**

In normal mode, writing a 1 to bit 10 (CSHAD) of the VRAM's COL field causes shadows to appear behind all characters following that COL. You can specify the color of the shadow in the color palette's FRAME register (x'03ED2'). Figure 7-12 shows an example of character shadowing. As shown in the figure, if a character contains dots in the right border of its field, the shadowing for those dots appear in the character field to the right.

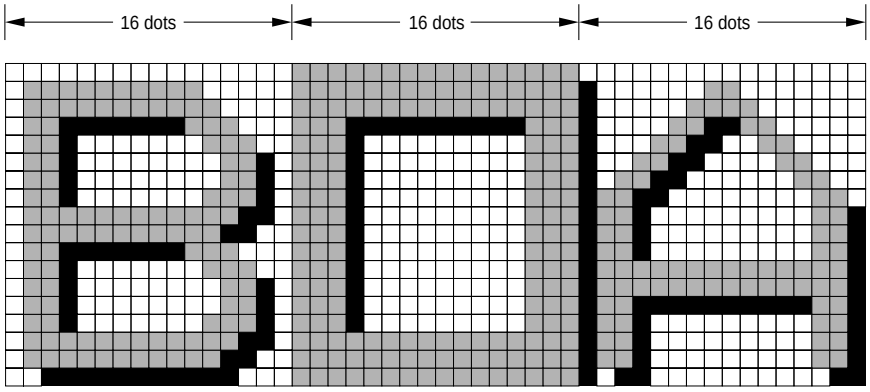


Figure 7-12 Character Shadowing Example

■ **Box shadowing**

In normal mode, writing a 1 to bit 12 (BSHAD1) of the VRAM's COL field causes boxes to appear around all characters following that COL. If COL bit 11 (BSHAD0) is 0, the color specified in the WBSHD register (x'03ED6') appears on the top and left sides of the box and the color specified in the BBSHD register (x'03ED4') appears on the bottom and right sides of the box. These positions are reversed if BSHAD0 is 1. Figure 7-13 shows an example of box shadowing. As shown in the figure, the right-hand border of the shadow box appears in the character field to the right of the shadowed text.

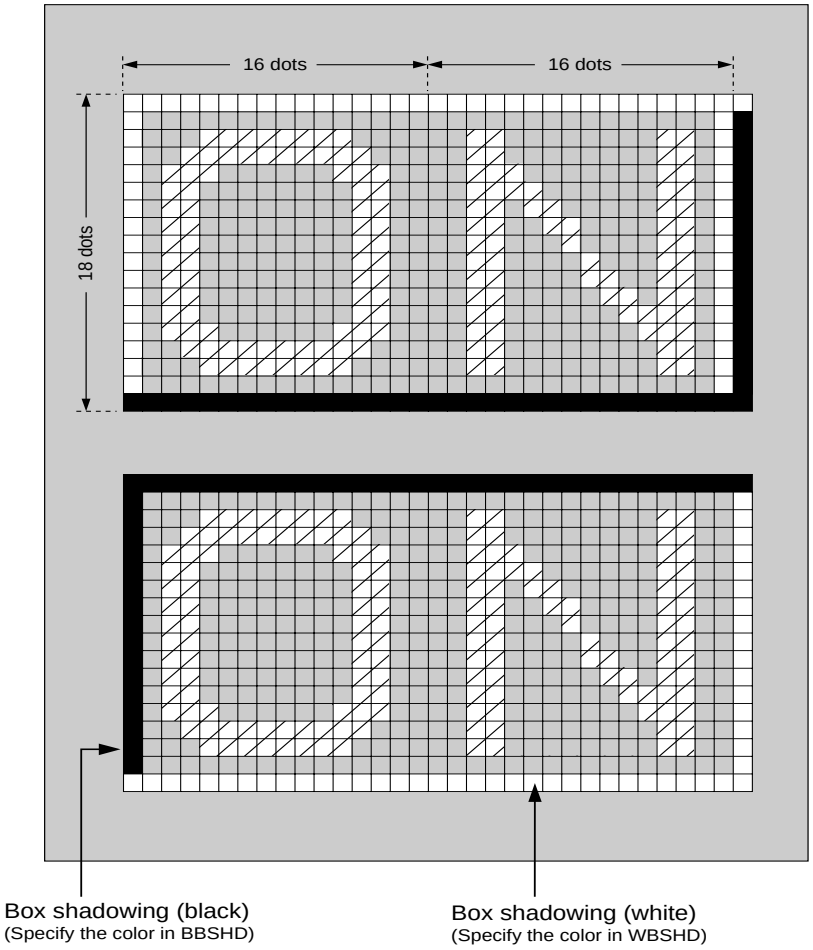


Figure 7-13 Box Shadowing Example

■ **Italicizing**

In closed-caption mode, writing a 1 to bit 10 (ITALIC) of the VRAM's COL field italicizes all characters following that COL. Figure 7-14 shows an example of an italicized character.

■ **Underlining**

In closed-caption mode, writing a 1 to bit 11 (CUNDL) of the VRAM's COL field underlines all characters following that COL. Figure 7-14 shows an example of an underlined character.

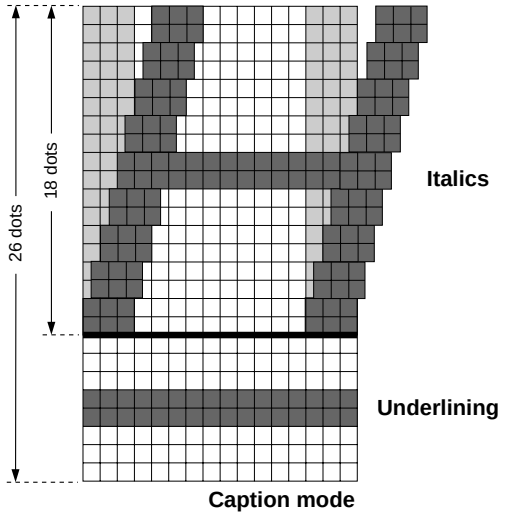


Figure 7-14 Italicizing and Underlining Example

■ **Blinking**

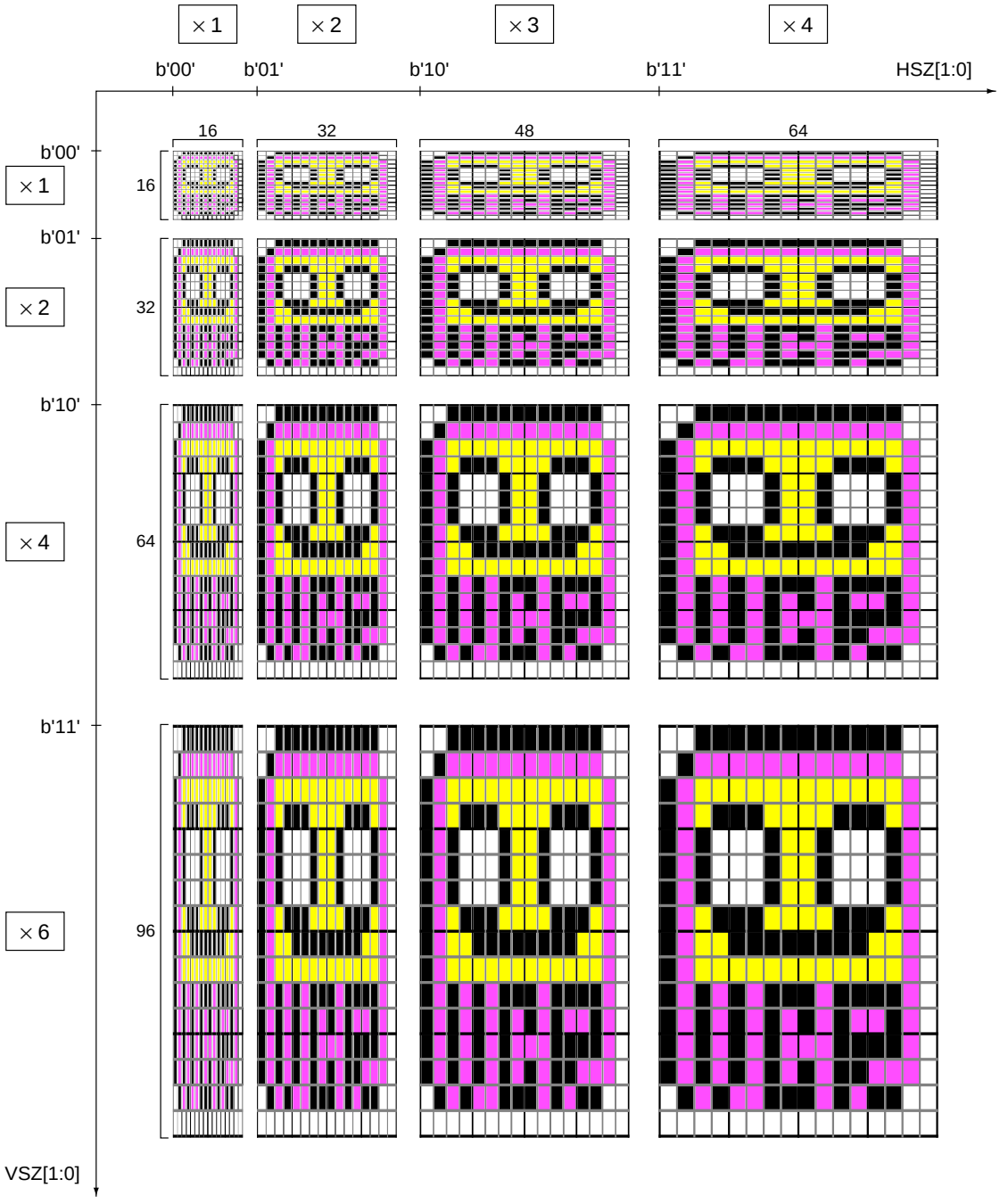
In both normal and closed-caption modes, writing a 1 to bit 8 (BLINK) of the VRAM's COL field causes all characters following that COL to blink. To use this function, you must enable blinking by writing a 1 to bit 5 (BLINK) of the OSD3 register (x'03EBA').

In closed-caption mode, you can specify whether or not the underlines blink on underlined, blinking characters. Set bit 1 (UNDF) of the OSD3 register to 0 to disable underline blinking or set it to 1 to enable underline blinking.

The blink cycle lasts for 128 VSYNC pulses (about 2 seconds). The characters display for 96 VSYNCs (about 1.5 seconds) and turn off for 32 VSYNCs (about 0.5 seconds).

7.7.3 Display Sizes

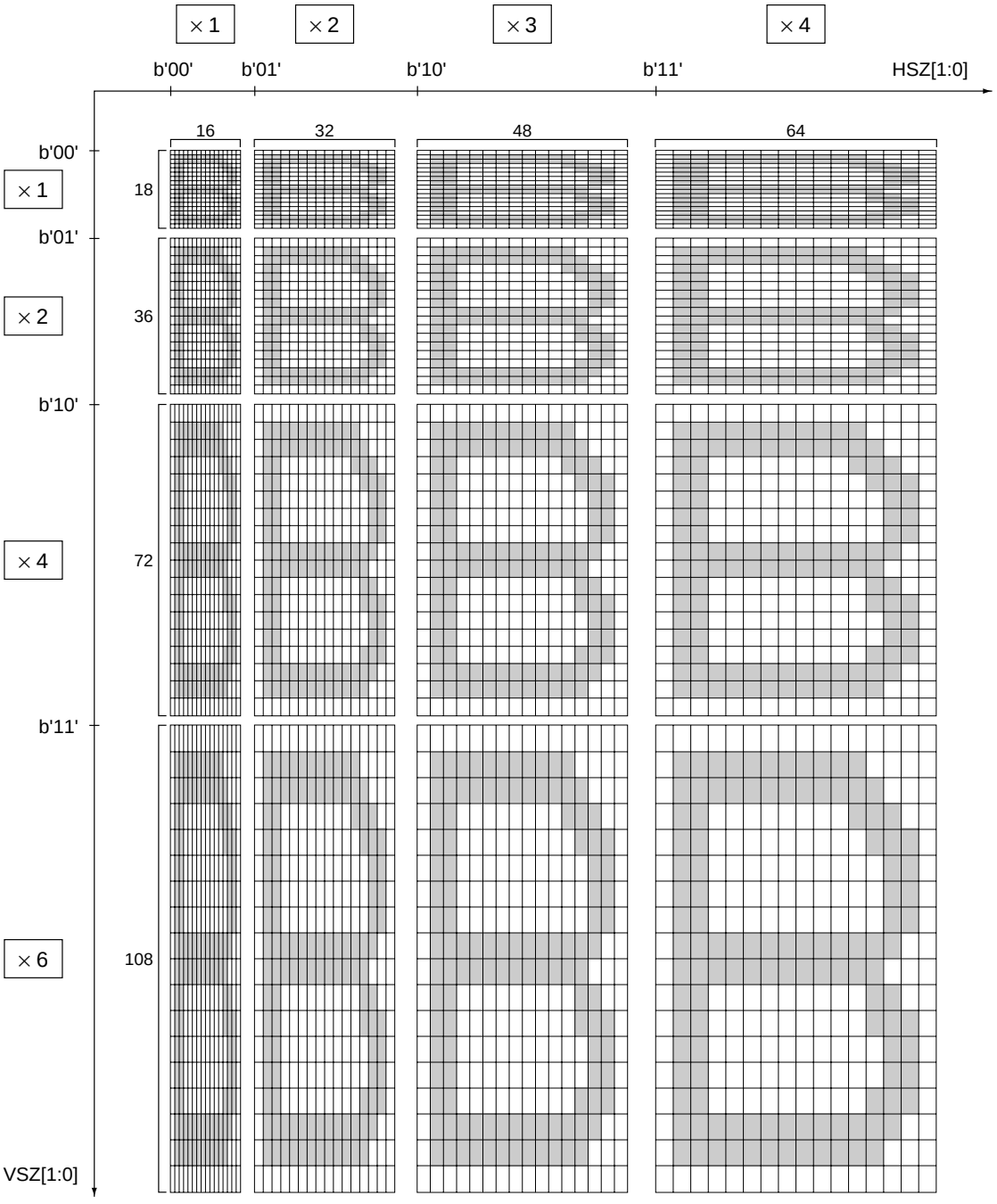
■ **Graphic tile sizes**



The settings shown are for interlaced displays. In progressive displays, the vertical size settings (VSZ[1:0]) are as follows: 01 = 1x, 10 = 2x, and 11 = 3x. The 00 setting is invalid.

Figure 7-15 Graphic Size Combinations

■ **Character sizes**



The settings shown are for interlaced displays. In progressive displays, the vertical size settings (VSZ[1:0]) are as follows: 01 = 1x, 10 = 2x, and 11 = 3x. The 00 setting is invalid.

Figure 7-16 Character Size Combinations

7.7.4 Setting Up the OSD Display Position

This section describes how to control the positioning of the OSD.

■ **To set the horizontal position of the display:**

- Write the position of the first line in the display to the IHP field (x'03ECA' and x'03ECB', bits 9 to 0).
- Write the position of the second and all following lines in the HP[9:0] field within the text display RAM data of the preceding line.
- Permissible ranges: IHP ≥ x'0C' and HP ≥ x'0C'

■ **About the horizontal start position on the screen**

The horizontal position, or HP settings determine where the left side of the display starts on the screen. You can set this value in 1-pixel units.

■ **To set the vertical position of the display:**

- Write the position of the first line in the display to the IVP[9:0] field (x'03ECC' and x'03ECD).
- Write the position of the second and all following lines in the VP[9:0] field within the text display RAM data of the preceding line.
- Permissible range: x'3F0' − (no. of H scan lines) ≥ IVP or VP ≥ x'03'

Sample VP range calculation

The base graphics line height is 18 dots, or H scan lines. If the graphics line you are positioning displays at 2× the base height, the number of H scan lines is:

$$18 \times 2 = 36 = x'24' \text{ H scan lines}$$

The permissible range of settings for VP is:

$$x'3F0' - x'24' = x'3CC' \geq VP \geq x'03'$$

When you write new values to the GIVP and CIVP fields, the settings take effect on the next VSYNC pulse. This means that changes are reflected in the next display screen rather than the current one.

■ **About the vertical start position on the screen**

The vertical position (VP) settings determine where the upper edge of the display starts on the screen. You can set this value in H scan line units.

7.8 DMA and Interrupt Timing

This section describes how the MN101C46F handles the timing of direct memory access (DMA) transfers of OSD data and OSD interrupts

■ Direct Memory Access

The microcontroller reads the line 1 data from the RAM as it scans line 1 onto the display. For line 2 and following lines, it reads the data as it scans the display start for the preceding line. The RAM read starts 13 system clock cycles ($13T_S$) after the leading edge of the HSYNC pulse. The DMA transfer takes $4T_S$ for each display data word.

If a DMA transfer occurs at the same time as the leading edge of a VSYNC pulse, the screen flickers. To avoid this, do not set a display position in the last line.

To prevent error, program data to meet the restrictions for the number of characters used, outlined in section 7.1, “Description,” on page 109.

■ Interrupts

The microcontroller processes the INT interrupt request bit of the display data’s VP field during the DMA transfer. If INT is set to 1, when the associated VP transfer ends the OSD generates an interrupt request.

Note that if the interrupt bit is set to 1 in the line 1 display data, the interrupt occurs at the first scan line. If the interrupt bit is set to 1 in the line 2 display data, the interrupt occurs at the first display line.

If an unexpected VSYNC or too many HSYNCs come in during a DMA transfer, the DMA transfer address may move to a different line and the microcontroller may shut down. If this occurs, add a dummy line to the VRAM area and set HP and VP.

■ IVP write timing

The IVP register determines the vertical positions of the L1 line shown in figure 7-17. Mistimed writes to this register can cause the OSD display to flicker. If, for instance, the write to IVP occurs above line L1, the display jumps immediately to the new L1 setting. If it occurs below line L1, however, the new display position (line L1) does not display until the next screen (or field). To prevent flickering,

always write to the IVP register at a point after L1 has displayed.

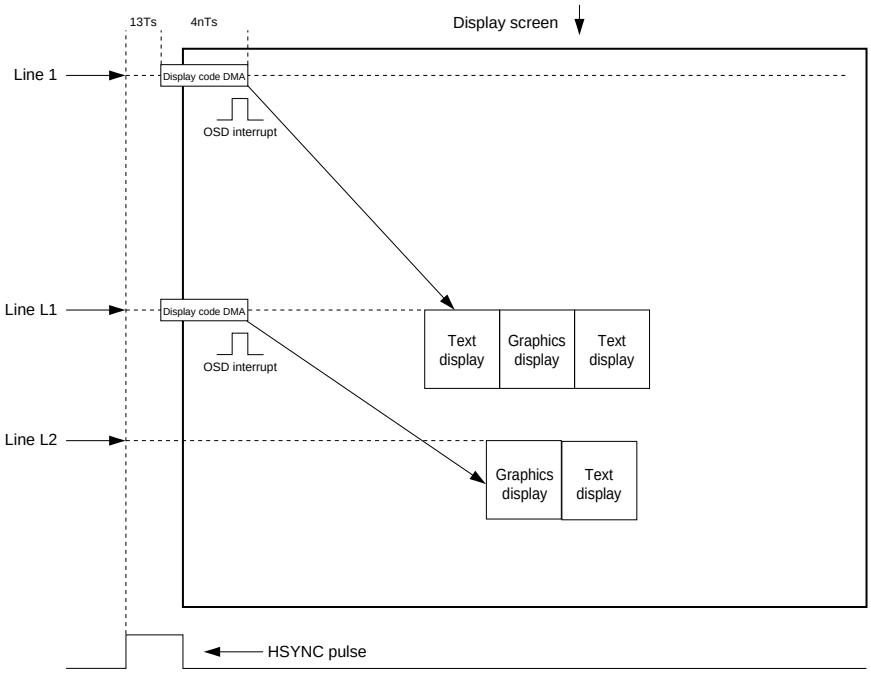


Figure 7-17 DMA and Interrupt Timing for the OSD

7.9 Selecting the OSD Dot Clock

This section describes how to set up the OSD dot clock.

The source for the OSD dot clock is programmable to either the 12- or 14.32-MHz clock supplied through the OSC1 and OSC2 pins then synchronized internally to the HSYNC pulse.

7.10 Controlling the Shuttering Effect

The MN101C46F OSD achieves a shuttering effect using four programmable shutters—two vertical and two horizontal. With this feature, you can shutter any portion of the OSD display, or you can combine shuttering with a wipe-out effect to create a smooth appearing and disappearing effect.

To prevent flickering and shadows on the display, only write to the registers during the VSYNC cycle.

7.10.1 Controlling the Shuttered Area

The register settings for the two vertical shutters (VSHT0 and VSHT1) and two horizontal shutters (HSHT0 and HSHT1) control which area of the screen is shuttered. Table 7-8 shows the register settings required for this function, and figure 7-18 shows four setup examples.

Table 7-8 Bit Settings for Controlling the Shuttered Area

Function	VSHT0 Bit	VSHT1 Bit	HSHT0 Bit	HSHT1 Bit	Description
Shutter enable/disable	VSON0	VSON1	HSON0	HSON1	0: Disable shutter (Acts as though there are no shutter lines.) 1: Enable shutter
Shutter position	VST00–VST09	VST10–VST19	HST00–HST09	HST10–HST19	For vertical shutters, this is the number of H scan lines from the top of the screen. For horizontal shutters, it is the number of pixels from the left of the screen.
Shuttering direction	VSP0	VSP1	HSP0	HSP1	0: Shutter below (vertical shutters) or to the right (horizontal shutters) 1: Shutter above (vertical shutters) or to the left (horizontal shutters)
Shuttering mode control	SHTRAD (shared bit)				0: AND the shuttered areas of all the shutters 1: OR the shuttered areas of all the shutters

■ **Determining the vertical shutter positions (VST0 and VST1)**

The top edge of the television screen is x'000'. Each integer higher brings the shutter position down one H scan line.

■ **Determining the horizontal shutter positions (HST0 and HST1)**

The left edge of the television screen is x'000'. Each integer higher brings the shutter position right one pixel. (One pixel, or one dot, is the smallest display unit in the OSD.)

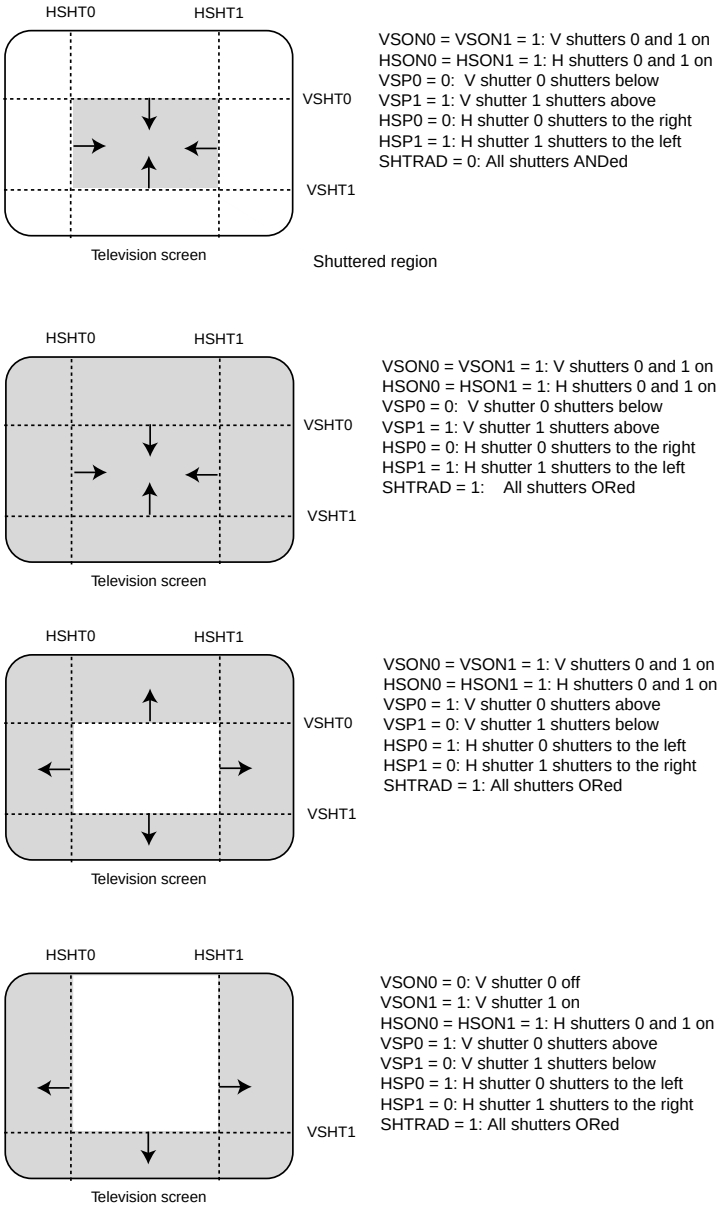


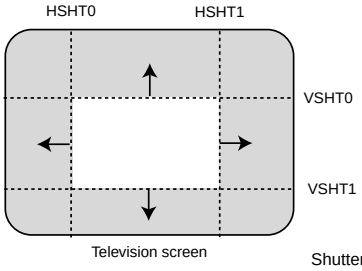
Figure 7-18 Shuttered Area Setup Examples

7.10.2 Controlling Shutter Movement

Enabling the shutter movement function in the registers allows the shuttered area to expand or contract over time, producing a wipe-in or wipe-out effect. This allows the OSD display to appear or disappear without an abrupt transition. Table 7-9 shows the register settings required for this function, and figure 7-19 shows four setup examples. There is no repeat operation for shutter movement, so you must reset the bits each time.

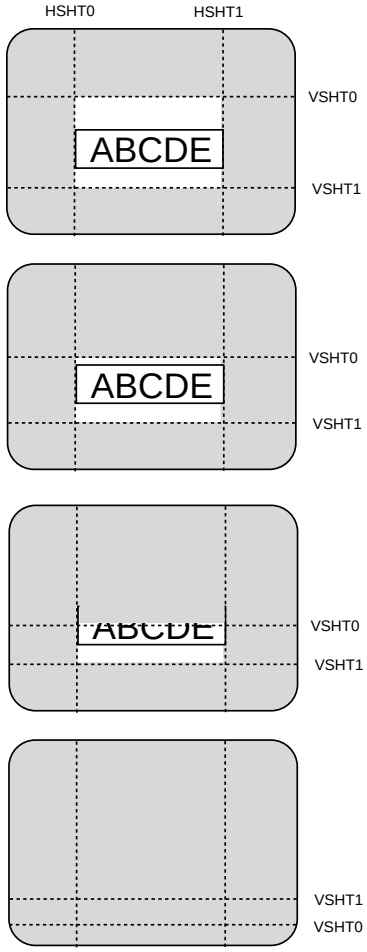
Table 7-9 Bit Settings for Controlling Shutter Movement

Function	VSHT0 Bit	VSHT1 Bit	HSHT0 Bit	HSHT1 Bit	Description
Shutter movement enable/ disable	VSM0	VSM1	HSM0	HSM1	0: Move shutter 1: Don't move shutter
Shuttering movement direction	VSMP0	VSMP1	HSMP0	HSMP1	0: Move from top to bottom (vertical shutters) or from left to right (horizontal shutters) 1: Move from bottom to top (vertical shutters) or from right to left (horizontal shutters)
Shuttering movement speed control	SHSP0, SHTSP1 (shared bits)				00: Move every VSYNC 01: Move every 2 VSYNCs 10: Move every 3 VSYNCs 11: Move every 4 VSYNCs



VSON0 = VSON1 = 1: V shutters 0 and 1 on
HSON0 = HSON1 = 1: H shutters 0 and 1 on
VSP0 = 1: V shutter 0 shutters above
VSP1 = 0: V shutter 1 shutters below
HSP0 = 1: H shutter 0 shutters to the left
HSP1 = 0: H shutter 1 shutters to the right
SHTRAD = 1: All shutters ORed

This example shows V shutter 0 moving downward.
It shutters both the text and the background color in the text layer.



VSM0 = 1: V shutter 0 movement enabled
VSM1 = 0: V shutter 1 movement disabled
HSM0 = HSM1 = 1: Movement enabled for H shutters 0 and 1
VSMP0 = 0: V shutter 0 moves downward
SHTSP0 = SHTSP1 = 0: Shutter moves 1 HSYNC each VSYNC

You must set OSDREGE = x'1', or the shutters will not move (PCNT2 register, bit 2).

Vertical shutter 0 stops x'3FF' HSYNC lines from the top of the screen.

Figure 7-19 Shutter Movement Setup Examples

7.10.3 Controlling Shuttering Effects

Through register settings, you can independently control shuttering for text, text background, graphics, and color background. You can also output blanks to the shuttered area.

You cannot shutter the cursor layer.

Table 7-9 shows the register settings required for these effects. There are three types of shuttering: shuttering of text, text background, and graphics, shuttering of the color background, and shutter blanking. The sections below describe how to control each of these.

Table 7-10 Bit Settings for Controlling Shuttering Effects

Function	Bit Name	Description
Text shuttering	CCSHT	0: Shutter text-layer characters 1: Don't shutter text-layer characters
Text background shuttering	BCSHT	0: Shutter text-layer background 1: Don't shutter text-layer background
Graphics shuttering	GSHT	0: Shutter graphics layer 1: Don't shutter graphics layer
Color background shuttering	COLBSHT	0: Shutter color background 1: Don't shutter color background
Shutter blanking	SHTBLK	0: Don't output blanks to the shuttered area 1: Output blanks to the shuttered area

■ **To shutter text-layer characters, text-layer background, and graphics layer:**

Text

Set the text shutter control bit, CCSHT, of the shutter control register, SHTC (x'03EC8') to 1.

Text background

Set the text background shutter control bit, BCSHT, of SHTC to 1.

Graphics

Set the text background shutter control bit, GSHT, of SHTC to 1.

Figure 7-20 shows three setup examples of text-layer shuttering.

Do not allow the horizontal shuttering boundaries to overlap any italicized portion of a closed-caption display. This distorts the italicized characters.

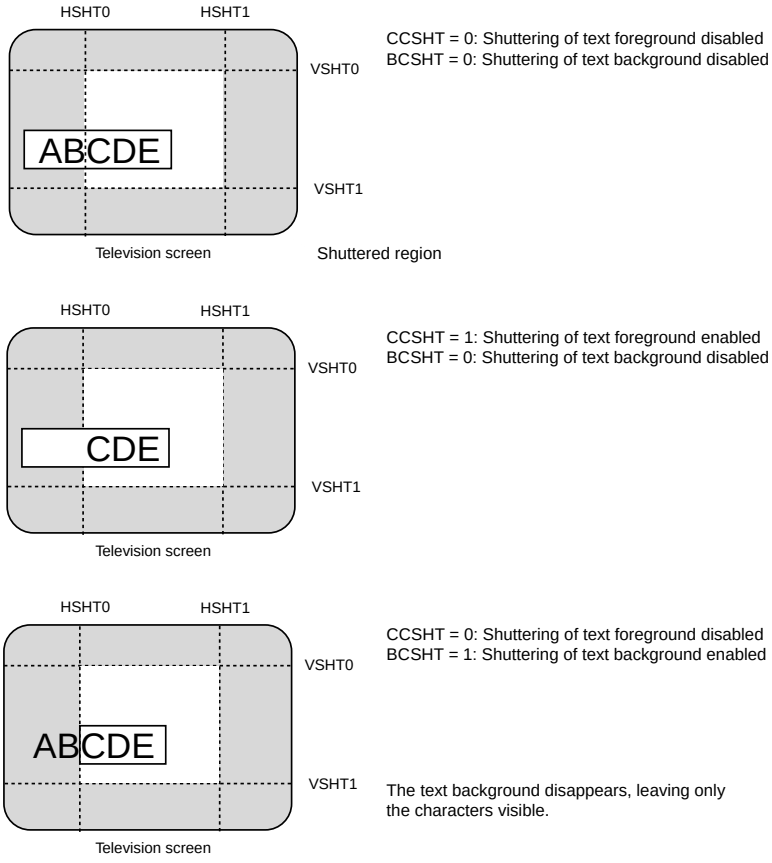


Figure 7-20 Text-Layer Shuttering Setup Examples

■ **To shutter the color background:**
Set the color background shutter control bit, COLBSHT, of the shutter control register, SHTC (x'03EC8') to 1.

This function exists only when the program enables a color background. It allows you to limit the area covered by the color background.

■ **To blank out the shuttered area:**
Set the shutter blanking control bit, SHTBLK, of SHTC to 1.

Shutter blanking outputs black to the entire shuttered area. To output blanking to a display that uses a color background, enable the color background shutter (COLBSHT = 1) so that the color background will also be blanked in the shuttered area. Figure 7-21 shows two setup examples.

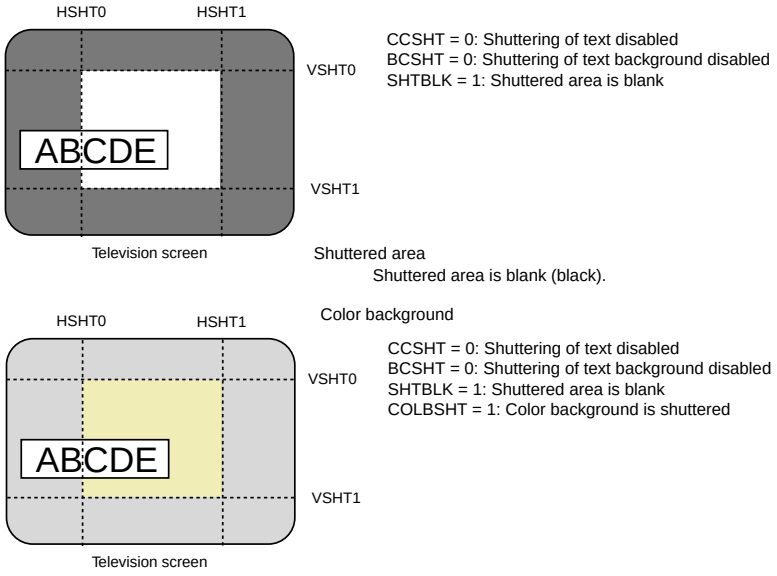


Figure 7-21 Shutter Blanking Setup Examples

7.10.4 Controlling Line Shuttering

It is possible to cancel shuttering of individual lines on the text and graphics layers so that they will be displayed on both shuttered and non-shuttered regions.

■ **To disable shuttering on the next line:**
Set the SHT bit (bit 10 of HP in the RAM data) to 1.

■ **To disable shuttering on the first line:**
Set the ISHT bit of the IHPH register (x'03ECB') to 1.

Figure 7-22 shows a setup example for the text layer.

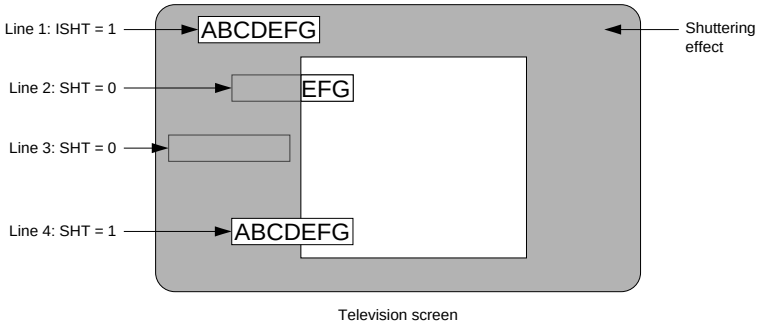


Figure 7-22 Line Shuttering Setup Example

7.11 Field Detection Circuit

7.11.1 Block Diagram

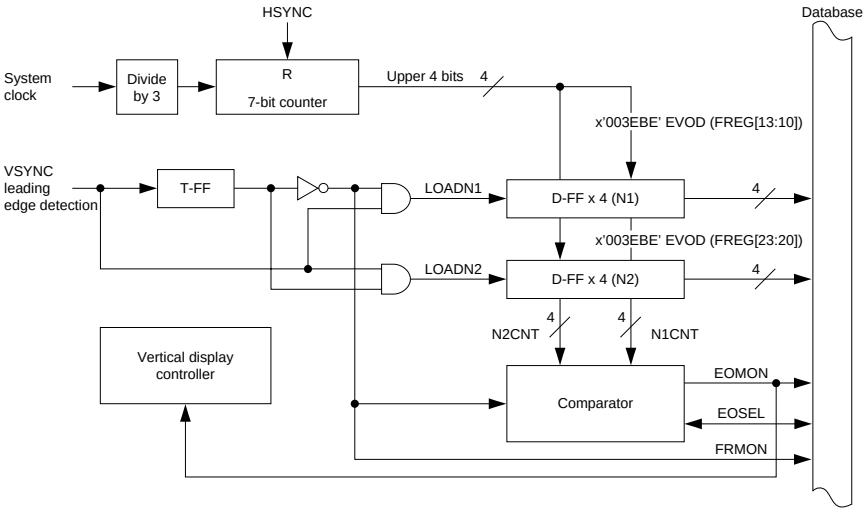


Figure 7-23 Field Detection Circuit Block Diagram

7.11.2 Description

The 7-bit field counter in this block resets every HSYNC interval to count the system clock. At each VSYNC interval, the four MSBs of the 7-bit counter are alternately loaded (made readable) to bits 7 to 4 (N2) and 3 to 0 (N1) of the EVOD register (x'03EBE'). The comparator compares the N1 and N2 values and outputs the results to the EOMON bit of EVODH (x'03EBF). The OSD identifies the field that sets EOMON to 1 as the display start field. Table 7-11 shows the criteria that the comparator uses.

By reading the FRMON bit of EVODH, the OSD can determine which register the four MSBs will load to on the next VSYNC input.

To ensure that the display starts at the right field, you must also set the EOSEL bit of EVODH so that EOMON becomes 1 at the display start field.

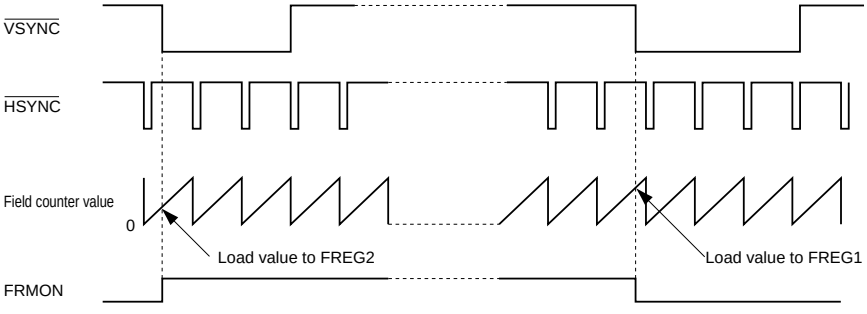


Figure 7-24 Field Detection Timing

Table 7-11 EOMON Output Criteria

FRMON	N1/N2 Relationship	EOMON Output	
		EOSEL = 0	EOSEL = 1
0 Load to N2 next	N1 > N2	0	1
	N1 < N2	1	0
	N1 = N2	Complement previous	Complement previous
1 Load to N1 next	N1 > N2	1	0
	N1 < N2	0	1
	N1 = N2	Complement previous	Complement previous

7.11.3 Considerations for Interlaced Displays

■ **Switching the display start field**

The OSD is constructed so the display start position is the field (field 1) where the EOMON bit is 1; however, interlaced displays may require that the start position be a field (field 2) where the EOMON bit is 0. In this case, merely complementing the EOSEL bit will not result in a correct display. You must set the following two bits to have the display start at field 2.

- **CANH** (x'03EBA', bit 4): Set to 1.
  0: Normal display
  1: Slide the field 1 display position down 1 line
- **EOSEL** (x'03EBF', bit 2): Complement the value EOSEL has for field 1 in a normal display.

■ **Scrolling in the closed-caption mode**

To implement text-layer scrolling in the closed-caption mode, the program must constantly switch the text display fields. This can cause the text lines to display incorrectly. To prevent this, set the following bits to fix the text lines to the even or odd field characters while scrolling.

- **BFLD** (x'03EBA', bit 2): Set to 1 to enable scrolling.
  0: Normal display
  1: Display the same characters in fields 1 and 2
- **EONL** (x'03EBA', bit 3): Set to 0 or 1.
  0: Fix to characters in field 1
  1: Fix to characters in field 2

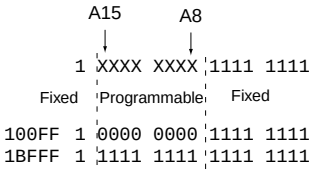
7.12 OSD Registers

CROMEND: Text ROM End Address Register **x'03EB0'**

Bit:	7	6	5	4	3	2	1	0
	A15	A14	A13	A12	A11	A10	A9	A8
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

A[15:8] hold the programmable portion of the text ROM end address. The low-order eight bits of the address are always x'FF' and the MSB is always b'1'. The available address range is x'1_00FF' to x'1_BFFF', with a programmable range from x'00' to x'BF'.

ROM data will be displayed incorrectly if CROMEND is set to x'C0' or higher.

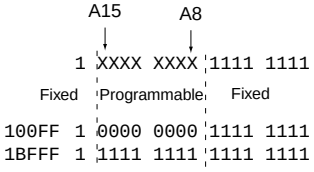


GROMEND: Graphics ROM End Address Register **x'03EB2'**

Bit:	7	6	5	4	3	2	1	0
	A15	A14	A13	A12	A11	A10	A9	A8
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

A[15:8] hold the programmable portion of the graphics ROM end address. The low-order eight bits of the address are always x'FF' and the MSB is always b'1'. The available address range is x'1_00FF' to x'1_BFFF', with a programmable range from x'00' to x'BF'.

ROM data will be displayed incorrectly if GROMEND is set to x'C0' or higher.

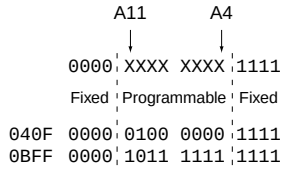


RAMEND: VRAM Address Register **x'03EB4'**

Bit:	7	6	5	4	3	2	1	0
	A11	A10	A9	A8	A7	A6	A5	A4
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

A[11:4] hold the VRAM end address RAMEND. The low-order four bits of the address are always x'F' and the high-order four bits are always x'0'. The available address range is x'040F' to x'0BFF', with a programmable range from x'40' to x'BF'.

Do not set outside x'40'-x'BF'. The microcontroller will not operate correctly.



IHPH: Initial Horizontal Position Register High **x'03ECB'**
IHP: Initial Horizontal Position Register **x'03ECA'**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	--	--	--	IHSZ1	IHSZ0	ISHT	IHP9	IHP8	IHP7	IHP6	IHP5	IHP4	IHP3	IHP2	IHP1	IHP0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	--	--	--	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

IHSZ[1:0]: Initial horizontal size

00: 1 dot = 1 VCLK period
01: 1 dot = 2 VCLK periods
10: 1 dot = 3 VCLK periods
11: 1 dot = 4 VCLK periods

ISHT: Initial shutter control

0: Shutter control on
1: Shutter control off

IHP[9:0]: Initial horizontal position

IVPH: Initial Vertical Position Register High **x'03ECD'**
IVP: Initial Vertical Position Register **x'03ECC'**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	--	--	--	IVSZ1	IVSZ0	--	IVP9	IVP8	IVP7	IVP6	IVP5	IVP4	IVP3	IVP2	IVP1	IVP0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	--	--	--	R/W	R/W	--	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

IVSZ[1:0]: Initial vertical size

IVSZ[1:0] Setting	1 Dot Size	
	Interlaced Displays	Progressive Displays
00	1 H scan line	Reserved
01	2 H scan lines	1 H scan line
10	4 H scan lines	2 H scan lines
11	6 H scan lines	3 H scan lines

IVP[9:0]: Initial vertical position

Set APCNT (bit 7 of x'03EB8') to 1 to make this setting valid.

IAPH: Initial RAM Address Pointer Register High **x'03ECF'**
IAP: Initial RAM Address Pointer Register **x'03ECE'**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	--	--	--	--	IAP11	IAP10	IAP9	IAP8	IAP7	IAP6	IAP5	IAP4	IAP3	IAP2	IAP1	IAP0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	--	--	--	--	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

IAP[11:0]: Initial RAM address (start of second line) pointer

Set APCNT (bit 7 of x'03EB8') to 1 to make this setting valid.

EVODH: Field ID Control Register HIgh **x'03EBF'**
EVOD: Field ID Control Register **x'03EBE'**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	--	--	--	--	--	EO SEL	FR MON	EO MON	FREG 23	FREG 22	FREG 21	FREG 20	FREG 13	FREG 12	FREG 11	FREG 10
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	--	--	--	--	--	R/W	R	R	R	R	R	R	R	R	R	R

EOSEL: Even/odd field select (R/W)

0: Select the smaller counter value as the display start field

1: Select the larger counter value as the display start field

FRMON: Field counter load monitor (R)

Monitors which field register (FREG) loads the counter value on the leading edge of VSYNC.

0: Loaded to FREG2[3:0]

1: Loaded to FREG1[3:0]

EOMON: Even/odd field monitor (R)

Flagged during display field interval.

0: No display start field detected

1: Display start field detected

FREG2[3:0]: Field register 2

4-bit register storing field counter value (R).

FREG1[3:0]: Field register 1

4-bit register storing field counter value (R).

HCOUNTH: Vertical Display Position Counter **x'03EBD'**
HCOUNT: Vertical Display Position Counter **x'03EBC'**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	--	--	--	--	--	--	HCNT 9	HCNT 8	HCNT 7	HCNT 6	HCNT 5	HCNT 4	HCNT 3	HCNT 2	HCNT 1	HCNT 0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	--	--	--	--	--	--	R	R	R	R	R	R	R	R	R	R

These registers hold the HSYNC count. They are cleared to 0 on the rising edge of VSYNC.

OSD1: OSD Register 1 **x'03EB6'**

Bit:	7	6	5	4	3	2	1	0
	--	HPOL	VPOL	YS POL	--	OSD	--	--
Reset:	0	0	0	0	0	0	0	0
R/W:	--	R/W	R/W	R/W	--	R/W	--	--

HPOL and VPOL: HSYNC and VSYNC input polarity select

HPOL–HSYNC and VPOL–VSYNC selects.

0: Active low

1: Active high

YSPOL: YS output polarity

0: Active high

1: Active low

OSD: OSD function switch

0: Off

1: On

The value written to bit 2 of OSD1 is valid from the leading edge of the next VSYNC. When the OSD function is on, the OSD starts operating from the VSYNC after this bit is set to 1. When the OSD function is off, the OSD stops operating at the next VSYNC after 0 is written.

OSD2: OSD Register 2 **x'03EB8'**

Bit:	7	6	5	4	3	2	1	0
	AP CNT	--	--	--	--	--	GCOL	COLB
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	--	--	--	--	--	R/W	R/W

APCNT: Address pointer enable

0: Address pointer off (AP of VRAM line disabled. The maximum number of horizontally displayable characters is 38.)

1: Address pointer on (AP of VRAM line enabled. The maximum number of horizontally displayable characters is 61.)

GCOL[1:0]: Graphics color mode

0: 8-color mode

1: 4-color mode

COLB: Color background control

0: Don't output

1: Output

When you stop the OSD function to lower current consumption, 0 is written to OSDPOFF to stop the OSD clock immediately after 0 is written to OSD. OSD-function-off will not be valid, so current consumption may not decrease. If this occurs, stop the OSD clock after writing 0 to OSD after VSYNC input.

OSD3: OSD Register 3 **x'03EBA'**

Bit:	7	6	5	4	3	2	1	0
	--	--	BLINK	CANH	EONL	BFLD	UNDF	CAPM
Reset:	0	0	0	0	0	0	0	0
R/W:	--	--	R/W	R/W	R/W	R/W	R/W	R/W

BLINK: Character blinking control

Controls blinking for characters with BLINK set in the COL code.

0: Don't blink

1: Blink

CANH: Vertical alignment control for closed captions

Active when interlacing is selected.

0: Normal position

1: Add 1 to V position of even fields

EONL and BFLD: Closed-caption scrolling control

Use when required for smoother scrolling.

00: Normal display

01: Fix to font of odd field during scrolling

10: Fix to font of even field during scrolling

11: Normal display

UNDF: Underline blinking control

0: Don't blink

1: Blink

CAPM: Closed-caption mode setting

0: Normal display mode

1: Closed-caption mode

VSHT0H: Vertical Shutter 0 Register High **x'03EC1'**
VSHT0: Vertical Shutter 0 Register **x'03EC0'**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	--	--	VSON0	VSP0	VSMP0	VSM0	VST09	VST08	VST07	VST06	VST05	VST04	VST03	VST02	VST01	VST00
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	--	--	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

VSON0: Vertical shutter 0 on/off

0: Off

1: On

VSP0: Vertical shutter 0 shuttering direction

0: Shutter below

1: Shutter above

VSMP0: Vertical shutter 0 movement direction

0: Bottom to top

1: Top to bottom

VSM0: Vertical shutter 0 movement control

0: Don't move

1: Move

VST0[9:0]: Vertical shutter 0 position

VSHT1H: Vertical Shutter 1 Register High **x'03EC3'**
VSHT1: Vertical Shutter 1 Register **x'03EC2'**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	--	--	VSON1	VSP1	VSMP1	VSM1	VST19	VST18	VST17	VST16	VST15	VST14	VST13	VST12	VST11	VST10
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	--	--	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

VSON1: Vertical shutter 1 on/off
- 0: Off
- 1: On

VSP1: Vertical shutter 1 shuttering direction
- 0: Shutter below
- 1: Shutter above

VSMP1: Vertical shutter 1 movement direction
- 0: Bottom to top
- 1: Top to bottom

VSM1: Vertical shutter 1 movement control
- 0: Don't move
- 1: Move

VST1[9:0]: Vertical shutter 1 position

HSHT0H: Horizontal Shutter 0 Register High **x'03EC5'**
HSHT0: Horizontal Shutter 0 Register **x'03EC4'**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	--	--	HSON0	HSP0	HSMP0	HSM0	HST09	HST08	HST07	HST06	HST05	HST04	HST03	HST02	HST01	HST00
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	--	--	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

HSON: Horizontal shutter 0 on/off
- 0: Off
- 1: On

HSP0: Horizontal shutter 0 shuttering direction
- 0: Shutter right side
- 1: Shutter left side

HSMP0: Horizontal shutter 0 movement direction
- 0: Right to left
- 1: Left to right

HSM0: Horizontal shutter 0 movement control
- 0: Don't move
- 1: Move

HST0[9:0]: Horizontal shutter 0 position

HSHT1H: Horizontal Shutter 1 Register High **x'03EC7'**
HSHT1: Horizontal Shutter 1 Register **x'03EC6'**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	--	--	HSON 1	HSP1	HSMP 1	HSM1	HST1 9	HST1 8	HST1 7	HST1 6	HST1 5	HST1 4	HST1 3	HST1 2	HST1 1	HST1 0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	--	--	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

HSON: Horizontal shutter 1 on/off
- 0: Off
- 1: On

HSP1: Horizontal shutter 1 shuttering direction
- 0: Shutter right side
- 1: Shutter left side

HSMP1: Horizontal shutter 1 movement direction
- 0: Right to left
- 1: Left to right

HSM1: Horizontal shutter 1 movement control
- 0: Don't move
- 1: Move

HST1[9:0]: Horizontal shutter 1 position

SHTC: Shutter Control Register **x'03EC8'**

Bit:	7	6	5	4	3	2	1	0
	SHT BLK	COLB SHT	SHT SP1	SHT SP0	SHT RAD	GSHT	BC SHT	CC SHT
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SHTBLK: Shutter blank function control.
- 0: Shutter blank off
- 1: Shutter blank on

COLBSHT: Color background shutter control
- 0: Disable
- 1: Enable

SHTSP[1:0]: Shutter speed control
Four speeds

SHTRAD: Shutter mode control
- 0: AND mode
- 1: OR mode

GSHT: Graphics shutter control
- 0: Disable
- 1: Enable

BCSHT: Text background shutter control
- 0: Disable
- 1: Enable

CCSHT: Character shutter control
- 0: Disable
- 1: Enable

COLB: Color Background Register **x'03ED0'**

Bit:	7	6	5	4	3	2	1	0
	COLBYM	COLBYS	COLBBH	COLBGH	COLBRH	COLBB	COLBG	COLBR
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Sets color when background is colored.

COLBYM: YM output

COLBYS: YS output

COLBBH: BOUT pin high impedance control
0: Push-pull control
1: High impedance

COLBGH: GOUT pin high impedance control
0: Push-pull control
1: High impedance

COLBRH: ROUT pin high impedance control
0: Push-pull control
1: High impedance

COLBB: Blue digital output (push-pull)

COLBG: Green digital output (push-pull)

COLBR: Red digital output (push-pull)

FRAME: Outlining and Character Shadowing Color Register **x'03ED2'**

Bit:	7	6	5	4	3	2	1	0
	FRAMEYM	FRAMEYS	FRAMEBH	FRAMEGH	FRAMERH	FRAMEB	FRAMEG	FRAMER
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Sets color of outlines and shading.

FRAMEYM: YM output

FRAMEYS: YS output

FRAMEBH: BOUT pin high impedance control
0: Push-pull control
1: High impedance

FRAMEGH: GOUT pin high impedance control
0: Push-pull control
1: High impedance

FRAMERH: ROUT pin high impedance control
0: Push-pull control
1: High impedance

FRAMEB: Blue digital output (push-pull)

FRAMEG: Green digital output (push-pull)

FRAMER: Red digital output (push-pull)

BBSHD: Black Box Shadowing Register **x'03ED4'**

Bit:	7	6	5	4	3	2	1	0
	BBSHDYM	BBSHDYS	BBSHDBH	BBSHDGH	BBSHDRH	BBSHDB	BBSHDG	BBSHDR
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Sets color of black box shadowing.

BBSHDYM: YM output

BBSHDYS: YS output

BBSHDBH: BOUT pin high impedance control
0: Push-pull control
1: High impedance

BBSHDGH: GOUT pin high impedance control
0: Push-pull control
1: High impedance

BBSHDRH: ROUT pin high impedance control
0: Push-pull control
1: High impedance

BBSHDB: Blue digital output (push-pull)

BBSHDG: Green digital output (push-pull)

BBSHDR: Red digital output (push-pull)

WBSHD: White Box Shadowing Register **x'03ED6'**

Bit:	7	6	5	4	3	2	1	0
	WBSHDYM	WBSHDYS	WBSHDBH	WBSHDGH	WBSHDRH	WBSHDB	WBSHDG	WBSHDR
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Sets color of white box shadowing.

WBSHDYM: YM output

WBSHDYS: YS output

WBSHDBH: BOUT pin high impedance control
0: Push-pull control
1: High impedance

WBSHDGH: GOUT pin high impedance control
0: Push-pull control
1: High impedance

WBSHDRH: ROUT pin high impedance control
0: Push-pull control
1: High impedance

WBSHDB: Blue digital output (push-pull)

WBSHDG: Green digital output (push-pull)

WBSHDR: Red digital output (push-pull)

PLT00–07: Palette 0 Colors 0–7 Register **x'03EE0'–x'03EE7'**

Bit:	7	6	5	4	3	2	1	0
	PLT00 YM	PLT00 YS	PLT00 BH	PLT00 GH	PLT00 RH	PLT00 B	PLT00 G	PLT00 R
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Color 0 of palette 0

PLT00YM: YM output

PLT00YS: YS output

PLT00BH: BOUT pin high impedance control
0: Push-pull control
1: High impedance

PLT00GH: GOUT pin high impedance control
0: Push-pull control
1: High impedance

PLT00RH: ROUT pin high impedance control
0: Push-pull control
1: High impedance

PLT00B: Blue digital output (push-pull)

PLT00G: Green digital output (push-pull)

PLT00R: Red digital output (push-pull)

Bit:	7	6	5	4	3	2	1	0
	PLT07 YM	PLT07 YS	PLT07 BH	PLT07 GH	PLT07 RH	PLT07 B	PLT07 G	PLT07 R
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Color 7 of palette 0

PLT07YM: YM output

PLT07YS: YS output

PLT07BH: BOUT pin high impedance control
0: Push-pull control
1: High impedance

PLT07GH: GOUT pin high impedance control
0: Push-pull control
1: High impedance

PLT07RH: ROUT pin high impedance control
0: Push-pull control
1: High impedance

PLT07B: Blue digital output (push-pull)

PLT07G: Green digital output (push-pull)

PLT07R: Red digital output (push-pull)

PLT10–17: Palette 1 Colors 0–7 Register **x'03EE8'–x'03EEF'**

Bit:	7	6	5	4	3	2	1	0
	PLT10 YM	PLT10 YS	PLT10 BH	PLT10 GH	PLT10 RH	PLT10 B	PLT10 G	PLT10 R
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Color 0 of palette 1

PLT10YM: YM output

PLT10YS: YS output

PLT10BH: BOUT pin high impedance control
0: Push-pull control
1: High impedance

PLT10GH: GOUT pin high impedance control
0: Push-pull control
1: High impedance

PLT10RH: ROUT pin high impedance control
0: Push-pull control
1: High impedance

PLT10B: Blue digital output (push-pull)

PLT10G: Green digital output (push-pull)

PLT10R: Red digital output (push-pull)

Bit:	7	6	5	4	3	2	1	0
	PLT17 YM	PLT17 YS	PLT17 BH	PLT17 GH	PLT17 RH	PLT17 B	PLT17 G	PLT17 R
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Color 7 of palette 1

PLT17YM: YM output

PLT17YS: YS output

PLT17BH: BOUT pin high impedance control
0: Push-pull control
1: High impedance

PLT17GH: GOUT pin high impedance control
0: Push-pull control
1: High impedance

PLT17RH: ROUT pin high impedance control
0: Push-pull control
1: High impedance

PLT17B: Blue digital output (push-pull)

PLT17G: Green digital output (push-pull)

PLT17R: Red digital output (push-pull)

PLT20–27: Palette 2 Colors 0–7 Register **x'03EF0'–x'03EF7'**

Bit:	7	6	5	4	3	2	1	0
	PLT20 YM	PLT20 YS	PLT20 BH	PLT20 GH	PLT20 RH	PLT20 B	PLT20 G	PLT20 R
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Color 0 of palette 2

PLT20YM: YM output

PLT20YS: YS output

PLT20BH: BOUT pin high impedance control
0: Push-pull control
1: High impedance

PLT20GH: GOUT pin high impedance control
0: Push-pull control
1: High impedance

PLT20RH: ROUT pin high impedance control
0: Push-pull control
1: High impedance

PLT20B: Blue digital output (push-pull)

PLT20G: Green digital output (push-pull)

PLT20R: Red digital output (push-pull)

Bit:	7	6	5	4	3	2	1	0
	PLT27 YM	PLT27 YS	PLT27 BH	PLT27 GH	PLT27 RH	PLT27 B	PLT27 G	PLT27 R
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Color 7 of palette 2

PLT27YM: YM output

PLT27YS: YS output

PLT27BH: BOUT pin high impedance control
0: Push-pull control
1: High impedance

PLT27GH: GOUT pin high impedance control
0: Push-pull control
1: High impedance

PLT27RH: ROUT pin high impedance control
0: Push-pull control
1: High impedance

PLT27B: Blue digital output (push-pull)

PLT27G: Green digital output (push-pull)

PLT27R: Red digital output (push-pull)

PLT30–37: Palette 3Colors 0–7 Register **x'03EF8'–x'03EFF'**

Bit:	7	6	5	4	3	2	1	0
	PLT30 YM	PLT30 YS	PLT30 BH	PLT30 GH	PLT30 RH	PLT30 B	PLT30 G	PLT30 R
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Color 0 of palette 3

PLT30YM: YM output

PLT30YS: YS output

PLT30BH: BOUT pin high impedance control
0: Push-pull control
1: High impedance

PLT30GH: GOUT pin high impedance control
0: Push-pull control
1: High impedance

PLT30RH: ROUT pin high impedance control
0: Push-pull control
1: High impedance

PLT30B: Blue digital output (push-pull)

PLT30G: Green digital output (push-pull)

PLT30R: Red digital output (push-pull)

Bit:	7	6	5	4	3	2	1	0
	PLT37 YM	PLT37 YS	PLT37 BH	PLT37 GH	PLT37 RH	PLT37 B	PLT37 G	PLT37 R
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Color 7 of palette 3

PLT37YM: YM output

PLT37YS: YS output

PLT37BH: BOUT pin high impedance control
0: Push-pull control
1: High impedance

PLT37GH: GOUT pin high impedance control
0: Push-pull control
1: High impedance

PLT37RH: ROUT pin high impedance control
0: Push-pull control
1: High impedance

PLT37B: Blue digital output (push-pull)

PLT37G: Green digital output (push-pull)

PLT37R: Red digital output (push-pull)

8 Analog/Digital Converter

8.1 Description

The MN101C46F contains an analog/digital converter (ADC) pair with a 5-bit resolution. It contains a sample hold circuit and can be programmed to switch among eight channels (ADIN7-ADIN0) of analog input. When the ADC is inactive, you can reduce power consumption by turning the internal ladder resistors off.

8.1.1 ADC Functions

The table below shows the A/D conversion functions.

Table 8-1 A/D Conversion Functions

A/D input pins	Eight
Pin names	ADIN7-ADIN0
Interrupt	ADIRQ
Resolution	5 bits
Conversion time (minimum)	13.4 µs (when t_{AD} = 1.12 µs)
Power down function	Internal ladder resistors can be turned on and off

8.2 ADC Block Diagram

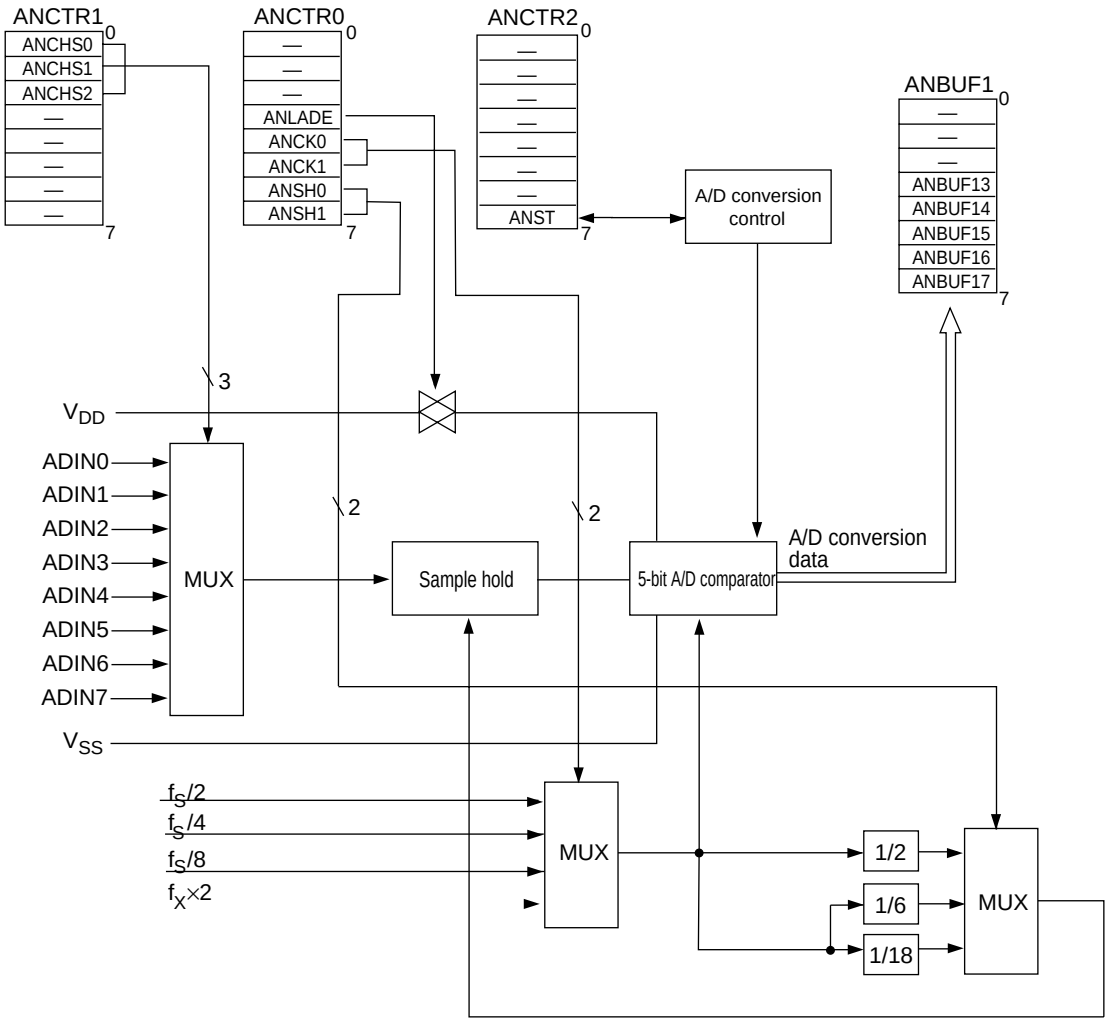


Figure 8-1 ADC Block Diagram

8.3 Analog-to-Digital Conversion Operation

The operation of the analog-to-digital conversion circuit is described below.

1. Set up the Analog Pins
   Use the port input mode register to set the pins that will be set up as analog input pins in step 2 as special function pins. (Complete the port input mode register setup before applying an analog voltage to the pins.)

2. Select the Analog Input Pin
   Select the analog input pin from among ADIN7-ADIN0 using the ANCHS[2:0] field of A/D control register 1 (ANCTR1).

3. Select the A/D Conversion Clock
   Select the A/D conversion clock using the ANCK[1:0] field of A/D control register 0 (ANCTR0). Set so that the oscillator you are using does not result in a conversion clock (t_{AD}) of 800 ns or below.

4. Set the Sample Hold Time
   Set the sample hold time using the ANSH[1:0] field of ANCTR0. Select a value for the sample hold time that is appropriate to the impedance of the analog input.

5. Set up the A/D Ladder Resistors
   Set the ANLADE flag of ANCTR0 to 1 to send current to the ladder resistors, and put A/D conversion in standby. (Steps 2 through 5 do not need to be done in order. Steps 3, 4, and 5 may even be done simultaneously.)

6. Select the A/D Conversion Start Source and Start A/D Conversion
   Set the ANST flag of A/D control register 2 (ANCTR2) to 1 to start A/D conversion.

7. A/D Conversion
   The ADC samples for the sample hold time set in step 3 and sequentially compares and determines the bits starting from the MSB.

8. End of A/D Conversion
   When A/D conversion is complete, the ADC clears the ANST flag and sends the results to the A/D buffer (ANBUF1). It also generates an A/D end interrupt request (ADIRQ) at this time.

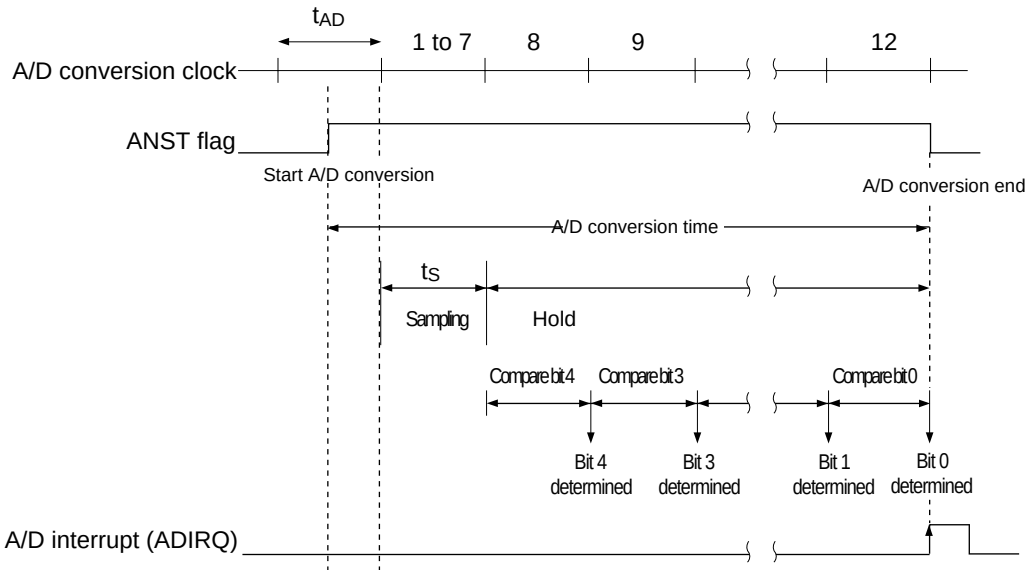


Figure 8-2 The A/D Conversion Operation

To read A/D conversion values, either run multiple A/D conversions and check that levels match in the program or find average values and remove noise.

8.3.1 Setting Up A/D Conversion

8.3.1.1 Setting up the A/D Conversion Input Pins

Select the pins to be used for A/D conversion input using the ANCHS[2:0] flag of ANCTR1.

Table 8-2 Setting up the A/D Conversion Input Pins

ANCHS2	ANCHS1	ANCHS0	A/D Pin
0	0	0	ADIN0 pin
		1	ADIN1 pin
	1	0	ADIN2 pin
		1	ADIN3 pin
1	0	0	ADIN4 pin
		1	ADIN5 pin
	1	0	ADIN6 pin
		1	ADIN7 pin

8.3.1.2 Setting up the A/D Conversion Clock

Set up the A/D conversion clock with the ANCK[1:0] field of the ANCTR1 register. Set the field so the A/D conversion clock (t_{AD}) does not fall below 800 ns. The table below illustrates the relationship between the machine clocks (f_{OSC}, f_S, and f_X) and the A/D conversion clock (t_{AD}). (Calculated with $f_S = f_{OSC}/2$ and $f_X/2$

Table 8-3 A/D Conversion Clocks and Cycles

ANCK1	ANCK0	A/D Conversion Clock	A/D Conversion Cycles (t_{AD})	
			f_{OSC} = 7.16 MHz	f_X = 1.79 MHz
0	0	$f_S/2$	559 ns (not settable)	2.24 µs
	1	$f_S/4$	1.12 µs	4.48 µs
1	0	$f_S/8$	2.24 µs	8.96 µs
	1	$f_X \times 2$	(Reserved)	(Reserved)

See section 2.13, “Setting the Clock Switch Register,” on page 42 for more information on the system clock (f_S).

8.3.1.3 Setting up the A/D Conversion Sampling Time

Setup the A/D conversion sampling time with the ANSH[1:0] field of the ANCTR0 register. Set the sampling time to an appropriate value for the analog input impedance, since it is changed by external circuits.

Table 8-4 A/D Conversion Sampling Time and Conversion Time

ANSH1	ANSH0	Sampling Time (t_S)	A/D Conversion Time	
			t_{AD} = 1.12 µs	t_{AD} = 8.96 µs
0	0	$t_{AD} \times 2$	13.4 µs	107.5 µs
	1	$t_{AD} \times 6$	17.9 µs	143.3 µs
1	0	$t_{AD} \times 18$	31.3 µs	250.8 µs
	1	Reserved	—	—

8.3.1.4 Controlling the Internal Ladder Resistors

Set the ANLADE flag in the ANCTR0 register to 1 to send current to the ladder resistor and put the ADC in standby. When the ADC is stopped, set ANLADE to 0 to reduce power consumption.

Table 8-5 Controlling the A/D Ladder Resistors

ANLADE	A/D Ladder Resistor Control
0	A/D ladder resistors OFF (A/D conversion stopped)
1	A/D ladder resistors ON (A/D conversion on standby)

8.3.1.5 Setting the Start of A/D Conversion

Set the ANST flag of the ANCTR2 to 1 to start A/D conversion. The ANST flag stays at 1 throughout A/D conversion and is cleared to 0 when the A/D conversion end interrupt is generated.

Table 8-6 Starting A/D Conversion

ANST	Starting A/D Conversion
0	Start A/D conversion, or conversion underway
1	Stop A/D conversion, or conversion halted

8.3.2 Setup Example of A/D Conversion

8.3.2.1 Operating the ADC with Register Settings

You can start A/D conversion with register settings. In the following example of setup procedures, the analog input pin is ADIN0, the conversion clock is $f_S/4$, and the sample hold time is $t_{AD} \times 6$. Conversion ends with an interrupt.

Table 8-7 ADC Setup Procedures

Procedure	Description
(1) Set up analog input pins. P0MD (x'03F28') bit 0—P0MD0 = 1 P0PLU (x0'3F40') bit 0—P0PLU0 = 0	(1) Use the port input mode register to set the analog input pins to be set in step 2 as special function pins. Use the port pullup/pulldown resistor control register to set up pins without pullup or pulldown resistors.
(2) Select analog input pin. ANCTR1 (x'03FB1') bits 2:0—ANCHS[2:0] = 000	(2) Select the analog input pins from among ADIN7-0 using the ANCHS[2:0] field of A/D control register 1 (ANCTR1).
(3) Select the A/D conversion clock. ANCTR0 (x'03FB0') bits 5:4—ANCK[1:0] = 01	(3) Select the A/D conversion clock with the ANCK[1:0] field of the ANCTR0 register.
(4) Set the sample hold time. ANCTR0 (x'03FB0') bits 7:6—ANSH[1:0] = 01	(4) Set the sample hold time using the ANSH[1:0] field of ANCTR0.
(5) Set the interrupt level. ADICR (x'03FFA') bits 7:6—ADLV[1:0] = 00	(5) Set the interrupt level with the ADLV[1:0] field of the A/D conversion end interrupt control register (ADICR). If the interrupt request flag may already be set, clear it. See section 3.5.2, “Setting the Interrupt Flags,” on page 58.
(6) Enable interrupts. ADICR (x’03FFA’) bit 1—ADIE = 1	(6) Set the ADIE flag in the ADICR register to 1 to enable interrupts.
(7) Set the A/D ladder resistor. ANCTR0 (x’03FB0’) bit 3—ANLADE = 1	(7) Set the ANLADE flag of ANCTR0 to 1 to send current to the ladder resistors, and put A/D conversion in standby.
(8) Start A/D conversion. ANCTR2 (x’03FB2’) bit 7—ANST = 1	(8) Set the ANST flag of A/D control register 2 (ANCTR2) to 1 to start A/D conversion.
(9) End A/D conversion. ANBUF1 (x’03FB4’)	(9) When A/D conversion is complete, the ADC generates an A/D conversion end interrupt and clears the ANST flag in ADCTR2. The results are stored in the A/D buffer (ANBUF1).

Note: Steps 3 and 4 can be done simultaneously.

8.3.3 Cautions on A/D Conversion

A/D conversion is susceptible to noise, so take preventative measures.

8.3.3.1 Noise Prevention

Place capacitors near the microcontroller's V_{SS} pins for the A/D input (analog input pins).

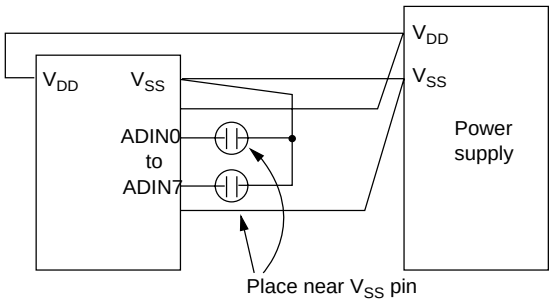


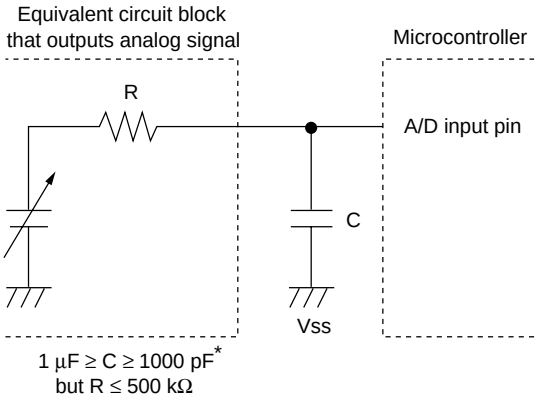
Figure 8-3 Recommended ADC Connection

To ensure A/D conversion precision, observe the following precautions when using the ADC.

1. Keep the input impedance R of the A/D input pins to 500 kΩ or less* and connect an external capacitor of between 1000 pF and 1 μF*.

2. Keep the time constants R and C in mind when setting the A/D conversion interval.

3. When running A/D conversion, A/D precision cannot always be guaranteed if the output levels of the microcontroller are changed or peripheral added circuits are turned on and off, since these actions cause fluctuations in analog input pins and current pins. When evaluating a set, check the waveform of the analog input pins.



Note: Asterisked figures are reference values.

Figure 8-4 Recommended Circuit for ADC

8.4 ADC Control Registers

A/D conversion employs control registers (ANCTRn) and a data storage buffer (ANBUF1).

8.4.1 ADC Control Registers

The following registers control A/D conversion

Table 8-8 ADC Control Registers

Register	Address	R/W	Description
ANCTR0	x'03FB0'	R/W	A/D control register 0
ANCTR1	x'03FB1'	R/W	A/D control register 1
ANCTR2	x'03FB2'	R/W	A/D control register 2
ANBUF1	x'03FB4'	R	A/D conversion data storage buffer 1
ADICR	x'03FFA'	R/W	A/D conversion interrupt control register

ANCTR0: A/D Control Register 0 **x'03FB0'**

Bit:	7	6	5	4	3	2	1	0
	ANSH1	ANSH0	ANCK1	ANCK0	ANLADE	—	—	—
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R	R	R

ANSH[1:0]: Sample hold time

00: $t_{AD} \times 2$

01: $t_{AD} \times 6$

11: $t_{AD} \times 18$

11: Do not use

ANCK[1:0]: A/D conversion clock ($f_{tAD} = 1/t_{AD}$)

00: $f_S/2$

01: $f_S/4$

10: $f_S/8$

11: $f_X \times 2$

ANLADE: A/D ladder resistor control

0: A/D ladder resistor OFF

1: A/D ladder resistor ON

ANCTR1: A/D Control Register 1 **x'03FB1'**

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	Reserved	ANCHS 2	ANCHS 1	ANCSH 0
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R/W	R/W	R/W

Reserved: Set to 0.

ANSH[2:0]: Analog input channel

000: ADIN0 (PA0)
001: ADIN1 (PA1)
010: ADIN2 (PA2)
011: ADIN3 (PA3)
100: ADIN4 (PA4)
101: ADIN5 (PA5)
110: ADIN6 (PA6)
111: ADIN7 (PA7)

ANCTR2: A/D Control Register 2 **x'03FB2'**

Bit:	7	6	5	4	3	2	1	0
	ANST	Reserved	—	—	—	—	—	—
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R	R	R	R	R	R	R

ANST: A/D Conversion Status

0: End or stopped
1: Start or converting

Reserved: Set to 0.

8.4.2 A/D Buffer

ANBUF1: *A/D Conversion Data Storage Buffer* **x'03FB4'**

Bit:	7	6	5	4	3	2	1	0
	ANBUF 17	ANBUF 16	ANBUF 15	ANBUF 14	ANBUF 13	—	—	—
Reset:	X	X	X	X	X	0	0	0
R/W:	R	R	R	R	R	R	R	R

This register stores the results of the A/D conversion.

9 Watchdog Timer

9.1 Description

The MN101C46F contains a watchdog timer function. The watchdog timer is used to detect software faults. It is controlled by the watchdog timer control register (WDTCR) and generates a watchdog interrupt (WDIRQ) when the watchdog timer overflows. Two consecutive watchdog interrupts indicate a software fault that cannot be recovered by software and requires a forcible reset by hardware.

9.1.1 Watchdog Timer Block Diagram

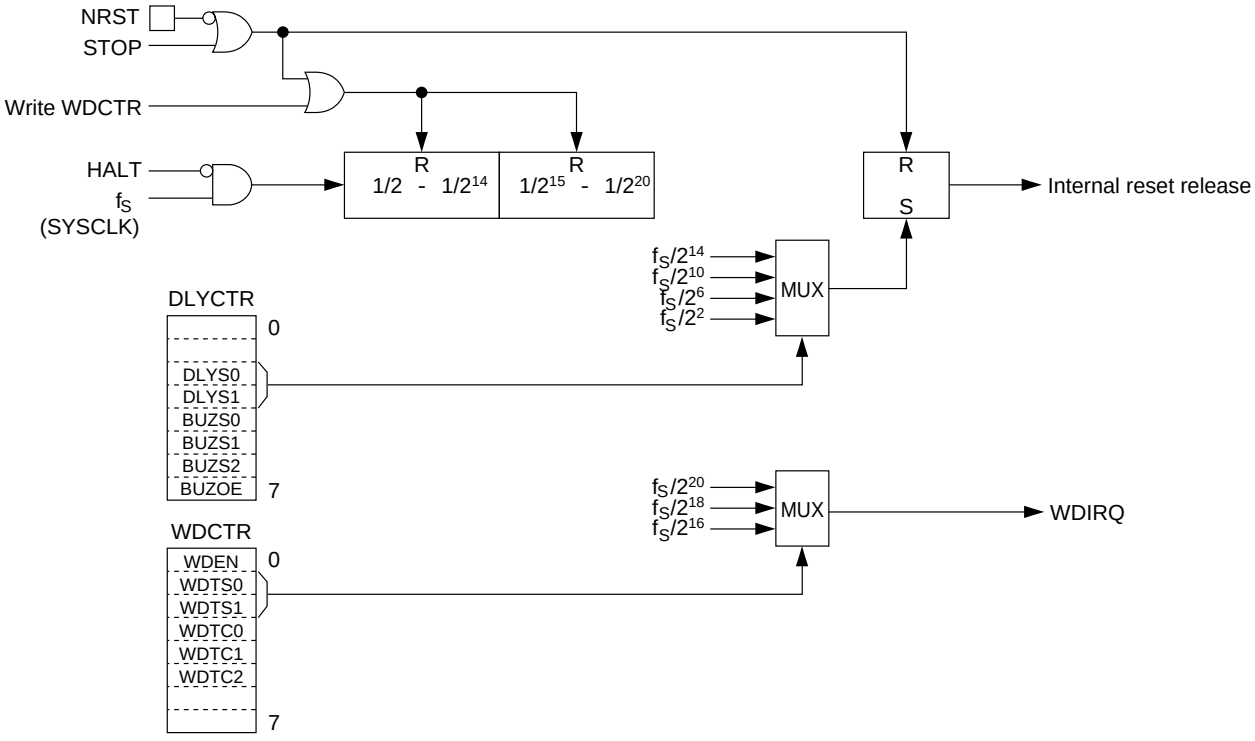


Figure 9-1 Watchdog Timer Block Diagram

The watchdog timer is also used to count the oscillation stabilization wait. It functions as a time-out timer except when a reset is cleared or when returning from STOP mode. For resets and in the STOP mode, the watchdog timer is initialized and starts counting from its initial value (x'0000') using the system clock (f_S) as its clock source. Set the oscillation stabilization wait in the oscillation stabilization wait control register (DLYCTR). After oscillation has stabilized, it continues counting as a watchdog timer. (See section 2.14, "Resets," on page 43.)

9.2 Operation of the Watchdog Timer

9.2.1 Watchdog Timer Function

The watchdog timer counts using system clock f_S as its clock source. When the timer overflows, it generates the watchdog interrupt (WDIRQ) as a nonmaskable interrupt (NMI). The watchdog timer stops at reset, but once it is activated, it cannot be stopped except by another reset. Use the watchdog timer control register (WDCTR) to clear the timer and to set the time-out period.

The watchdog timer can also detect software faults that repeatedly clear the watchdog timer in short periods. When the watchdog timer is cleared in a period shorter than the set time (the lower limit at which the watchdog timer can be cleared), it is considered a software fault and a watchdog interrupt (WDIRQ) is generated.

Once started, the watchdog timer cannot be stopped.

Two consecutive watchdog interrupts will be interpreted as a software fault that cannot be recovered by software and requires a forcible reset by hardware.

9.2.1.1 Using Watchdog Timer Functions

Periodically clear the watchdog timer during your program to avoid timer overflows when you are using the watchdog timer function. When the microcontroller experiences a software fault for any reason, the program will not be able to execute correctly, the watchdog timer will overflow, and a software fault will be detected.

Programming of watchdog timer functions should generally be performed at the last stage of program debugging.

9.2.1.2 Methods of Software Fault Detection

When the program is running correctly, it is expected that the watchdog timer function is cleared at set intervals. The MN101C46F's watchdog timer has two ways it detects software faults.

1. When the watchdog timer overflows.

2. When the watchdog timer clears at an interval shorter than the value set in the WDCTR register for the lower limit at which the timer can be cleared.

When a software fault is detected, a watchdog interrupt (WDIRQ) is generated as a nonmaskable interrupt (NMI).

9.2.1.3 Clearing the Watchdog Timer

The watchdog timer can be cleared by writing in the WDCTR. The timer is cleared regardless of the value written. We recommend that you use the bit set (BSET) instruction, which does not change the value.

9.2.1.4 Time-Out Period

The time-out period is determined by the system clock (f_S) and bits 2 and 1 (WDTS[1:0]) of the WDCTR. If the watchdog timer cannot be cleared before this value is reached, it is considered a software fault and the watchdog interrupt (WDIRQ) is generated as a nonmaskable interrupt (NMI).

Table 9-1 Time-Out Periods

WDTS1	WDTS0	Time-Out Period
0	0	2^{16} system clocks
0	1	2^{18} system clocks
1	X	2^{20} system clocks

Note: The system clock is determined by the CPU mode control register (CPUM). See section 2.13, “Setting the Clock Switch Register,” on page 42.

The time-out period is generally determined by the run time of the program’s main routine. Set a time-out period that is longer than the main routine’s run time divided by a given natural number (1, 2, ...). Insert watchdog timer clear commands in the main routine such that their number makes it an equivalent period.

9.2.1.5 Lower Limit at which Watchdog Can Be Cleared

The lower limit value at which the watchdog timer can be cleared is determined by bits 5, 4, and 3 (WDTC2, WDTC1, and WDTC0) of the WDCTR register.

Table 9-2 Lower Limit at which Watchdog Can Be Cleared

WDTC2	WDTC1	WDTC0	Lower Limit Value at which Watchdog Can Be Cleared
0	0	0	None
		1	2^7 system clocks
	1	0	2^9 system clocks
		1	2^{11} system clocks
1	0	0	2^{13} system clocks
		1	2^{15} system clocks
	1	0	2^{17} system clocks
		1	2^{19} system clocks

9.2.1.6 Relationship between Watchdog Timer and CPU Mode

The watchdog timer has the following relationship to the CPU mode.

1. In NORMAL, IDLE, and SLOW modes, the watchdog timer counts the system clock.

2. The watchdog timer count continues regardless of switching between the NORMAL, IDLE, and SLOW modes.

3. HALT mode stops the watchdog timer.

4. In STOP mode, the watchdog timer clears automatically.

5. No watchdog interrupts are generated in STOP mode.

6. After a reset is cleared and after returning from STOP mode, the count continues only for the oscillation stabilization wait.

In systems that use STOP mode, the program will usually branch depending on whether STOP is invoked or not. When this happens, the count value of the watchdog timer will vary, so consider the danger of triggering a watchdog interrupt when you are setting the lower limit at which the watchdog timer can be cleared.

9.2.2 Setup Examples for the Watchdog Timer

In this example, the watchdog timer is used to detect software faults. The detection period is 2^{18} system clocks and the lower limit for clearability is 2^9 system clocks. The procedure is described below.

Table 9-3 Initialization Program (Example Setup that Initializes the Watchdog Timer)

Procedure	Description
(1) Set time-out period. WDCTR (x0'3F02') bits 2:1—WDTS[1:0] = 01	(1) Set the WDTS[1:0] field of the WDCTR to 01 to select 2^{18} system clocks as the time-out period.
(2) Set lower limit of clearability. WDCTR (x'03F02') bits 5:3—WDTC[2:0] = 010	(2) Set the WDTC[2:0] field of the WDCTR to 010 to select 2^9 system clocks as the lower limit for clearability.
(3) Start watchdog timer running. WDCTR (x'03F02') bit 0—WDEN = 1	(3) Set the WDEN flag in the WDCTR to 1 to start the watchdog timer running.

Place the command that sets the WDEN flag to 1 at the end of the initialization. If the WDCTR register is changed after operation begins, a watchdog interrupt will be generated by the setting for lower limit of clearability.

Table 9-4 Program Main Routine (Example Setup that Periodically Clears the Watchdog Timer)

Procedure	Description
(1) Periodically clear the watchdog timer. Write to WDCTR (x'03F02') (c.f.) BSET (WDCTR) WDEN bit 0—WDEN = 1	(1) Clears the watchdog timer with a period of at least 2^9 system clocks and no more than 2^{18} system clocks. Insert watchdog timer clears within the main routine so periods are equal and the set period is achieved. We recommend an instruction that does not change the value, such as a bit set (BSET), for the clear.

Table 9-5 Setup of Interrupt Service Routine

Procedure	Description
(1) Watchdog interrupt service. NMICR (x'03FE1') TBNZ (NMICR) WDIR, WDPRO	(1) A nonmaskable interrupt is generated when the watchdog timer overflows. Check that the WDIR flag in the nonmaskable interrupt control register (NMICR) is set to 1 with the interrupt service routine and perform service appropriate to the system.

Operation just prior to when a watchdog interrupt is engaged cannot be guaranteed. When a watchdog interrupt occurs, run a program that initializes the system.

9.3 Watchdog Timer Control Register

The watchdog timer is controlled by the watchdog timer control register (WDCTR).

Register	Address	R/W	Description
WDTCR	x'03F02'	R/W	The watchdog timer control register

WDTCR: Watchdog TImer Control Register **x'03F02'**

Bit:	7	6	5	4	3	2	1	0
	—	—	WDTC2	WDTC1	WDTC0	WDTS1	WDTS0	WDEN
Reset:	0	0	0	0	0	1	1	0
R/W:	R	R	R/W	R/W	R/W	R/W	R/W	R/W

WDTC[2:0]: Watchdog Timer Counter

This field sets the lower limit value at which the watchdog timer can be cleared.

000: None
001: 2^7 system clocks
010: 2^9 system clocks
011: 2^{11} system clocks
100: 2^{13} system clocks
101: 2^{15} system clocks
110: 2^{17} system clocks
111: 2^{19} system clocks

WDTS[1:0]: Time-out Period Setting

00: 2^{16} system clocks
01: 2^{18} system clocks
1X: 2^{20} system clocks

WDEN: Watchdog Timer Enable

0: Disable
1: Enable

10 Closed-Caption Decoder

10.1 Description

The MN101C46F contains two identical closed-caption decoder circuits, CCD0 and CCD1. The decoders extract encoded captions from composite video signals. Figure 10-1 provides a block diagram of the decoders, and section 10.3, "Functional Description," on page 175, describes the circuit's main blocks. Note that this section describes CCD0, but all descriptions also apply to CCD1. Table 10-1 provides the pin names for each decoder.

Table 10-1 Pins Used for CCD0 and CCD1

Closed-Caption Decoder	Pin Name			
CCD0	CVBS0	VREFH0	CLH0	CLL0
CCD1	CVBS1	VREFH1	CLH0	CLL0

10.2 Block Diagram

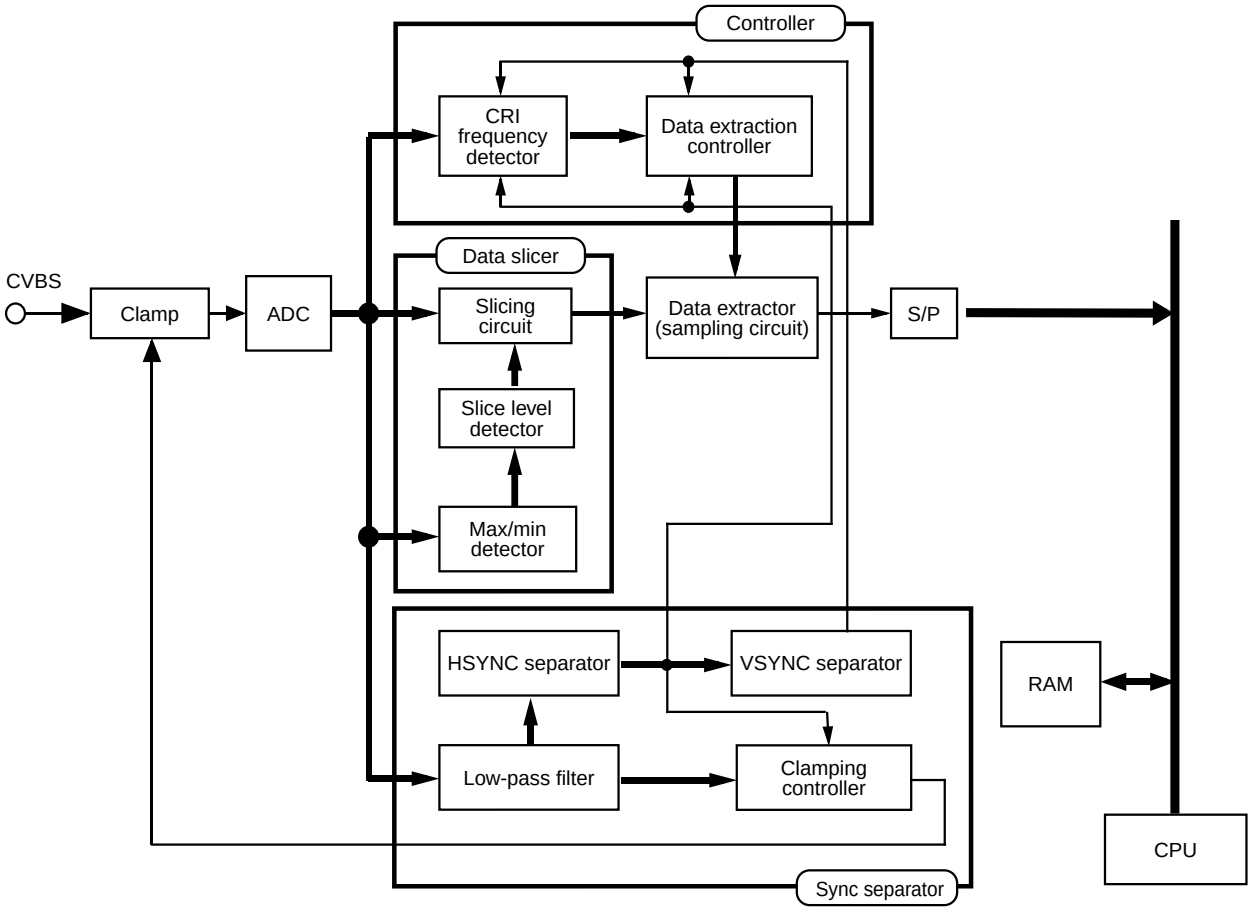


Figure 10-1 Closed-Caption Decoder Block Diagram

10.3 Functional Description

10.3.1 Analog-to-Digital Converter

The analog-to-digital converter (ADC) converts the clamped video signal to 8-bit digital data using a 14.32 MHz sampling clock. Figure 10-2 shows an example configuration using the recommended external pin connections. In this example, both caption decoders are used. Figure 10-3 shows the recommended connection when neither decoder is used, and figure 10-4 shows that when only CCD0 is used.

The constants shown in figures 10-2 to 10-4 are recommended values only. Operation at these values is not guaranteed.

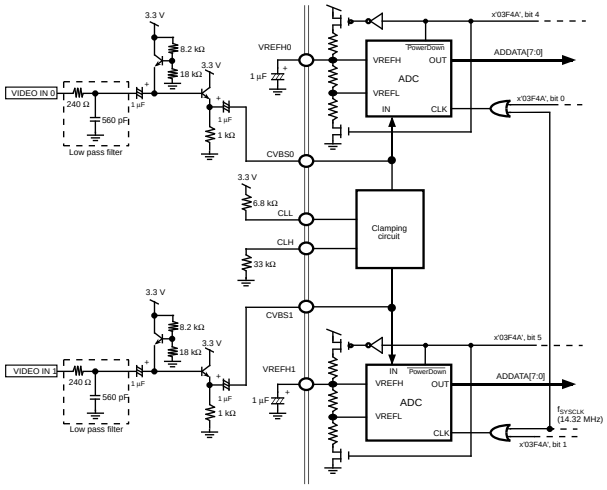


Figure 10-2 Recommended ADC Configuration

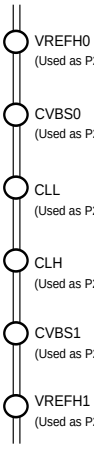


Figure 10-3 External Connection with Both CCD0 and CCD1 Unused

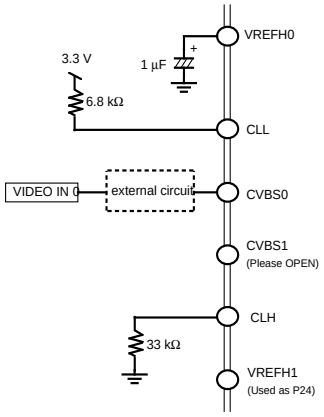


Figure 10-4 External Connection with Only CCD0 Unused

Always set the same value to bits 6 and 7 of P2MD. Otherwise CCD's don't work correctly.

Table 10-2 provides the register setting for caption terminals in cases of Figure 10-2, Figure 10-3, and figure 10-4. And always set both bits 6 and 7 of P2MD to 1, whether using two CCD's or only one CCD. In not using any CCD, always set these two bits to 0.

Table 10-2 Caption decoder register setting

		VBI control	ADC control	Clamp control
Use two caption decoders	caption 0	ON(PCNT0.bp0=0)	ON(PCNT0.bp3=1)	ON(P2MD.bp6=1)
	caption 1	ON(PCNT0.bp1=0)	ON(PCNT0.bp4=1)	ON(P2MD.bp5=1)
Use one caption decoder	caption 0	ON(PCNT0.bp0=0)	ON(PCNT0.bp3=1)	ON(P2MD.bp6=1)
	no caption 1	OFF(PCNT0.bp1=1)	OFF(PCNT0.bp4=0)	ON(P2MD.bp5=1)
No use caption decoder	no caption 0	OFF(PCNT0.bp0=1)	OFF(PCNT0.bp3=0)	OFF(P2MD.bp6=0)
	no caption 1	OFF(PCNT0.bp1=1)	OFF(PCNT0.bp4=0)	OFF(P2MD.bp5=0)

10.3.2 Clamping Circuit

This block clamps the input video signal (CVBS0, CVBS1).

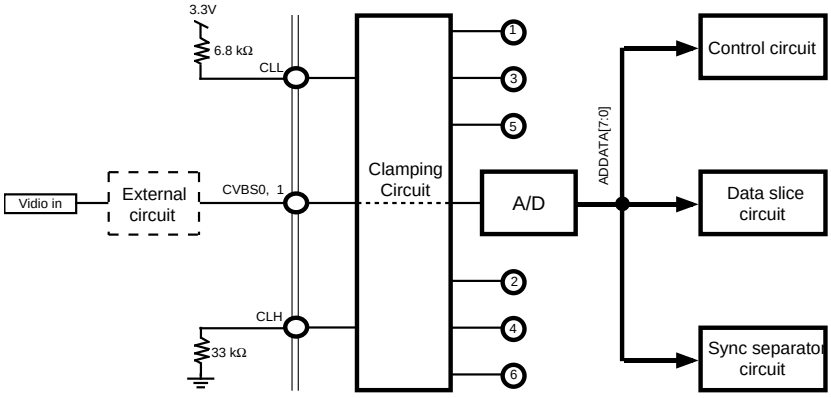


Figure 10-5 Clamping Circuit

The clamping circuit internal to the MN101C46F provides three current sources—high, medium, and low. You can modify these current sources using external resistors R1 and R2. Within the clamping circuit, you can turn each of the current sources on and off in steps. The control bits for these currents are the same for sync tip and pedestal clamping, but the reference and compare levels are different. Table 10-3 provides these values for the two types of clamping, and table 10-4 shows how to control the three current levels so that the video signal matches the reference level.

Table 10-3 Clamping Reference and Compare Levels

Clamping Type	Reference Level		Compare Level	
	CCD0	CCD1	CCD0	CCD1
Sync tip clamping	16 (dec)	16 (dec)	Output from minimum detection circuit (value in SYNCMIN, x'03E48')	Output from minimum detection circuit (value in SYNCMIN, x'03E68')
Pedestal clamping	Value in PCLV, x'03E4D'	Value in PCLV, x'03E6D'	Value in BPLV, x'03E49'	Value in BPLV, x'03E69'

Table 10-4 Current Level Control

Control Conditions	Current Source					
	Low Current		Medium Current		High Current	
	(1)	(2)	(3)	(4)	(5)	(6)
10 ≤ A	Off	On	Off	On	Off	On
4 ≤ A ≤ 9	Off	On	Off	On	Off	Off
1 ≤ A ≤ 3	Off	On	Off	Off	Off	Off
A= 0	Off	Off	Off	Off	Off	Off
-3 ≤ A ≤ -1	On	Off	Off	Off	Off	Off
-9 ≤ A ≤ -4	On	Off	On	Off	Off	Off
A ≤ -10	On	Off	On	Off	On	Off

Notes: 1. A = compare level - reference level

2. The numbers (1) to (6) correspond to the numbers in figure 10-5.

Table 10-5 provides the registers used to control and monitor the clamping circuit. See the page number indicated for register and bit descriptions.

Table 10-5 Control Registers for Clamping Circuit

Register	Page	CCD0 Address	CCD1 Address	Description
Register for selecting the low-pass filter				
NFSELH	191	x'03E41'	x'03E61'	Noise filter select register high
NFSEL	191	x'03E40'	x'03E60'	Noise filter select register
Registers for controlling clamping				
SCMINGH	192	x'03E45'	x'03E65'	Minimum sync level detection interval set register high
SCMING	192	x'03E44'	x'03E64'	Minimum sync level detection interval set register
SYNCMIN	193	x'03E48'	x'03E68'	Minimum sync level register
BPPSTH	192	x'03E47'	x'03E67'	Backporch position register high
BPPST	192	x'03E46'	x'03E66'	Backporch position register
BPLV	193	x'03E49'	x'03E69'	Pedestal level register
CLAMPH	194	x'03E4D'	x'03E6D'	Clamping control register high
CLAMP	194	x'03E4C'	x'03E6C'	Clamping control register
CLPCND1	197	x'03E5C'	x'03E7C'	Clamping control signal status register 1

10.3.3 Sync Separator Circuit

A low-pass filter and a sync separator comprise this block. The sync separator extracts HSYNC and VSYNC from the composite video signal. Figure 10-6 shows a block diagram of the circuit, and table 10-6 provides the registers used to control and monitor it. See the page number indicated for register and bit descriptions.

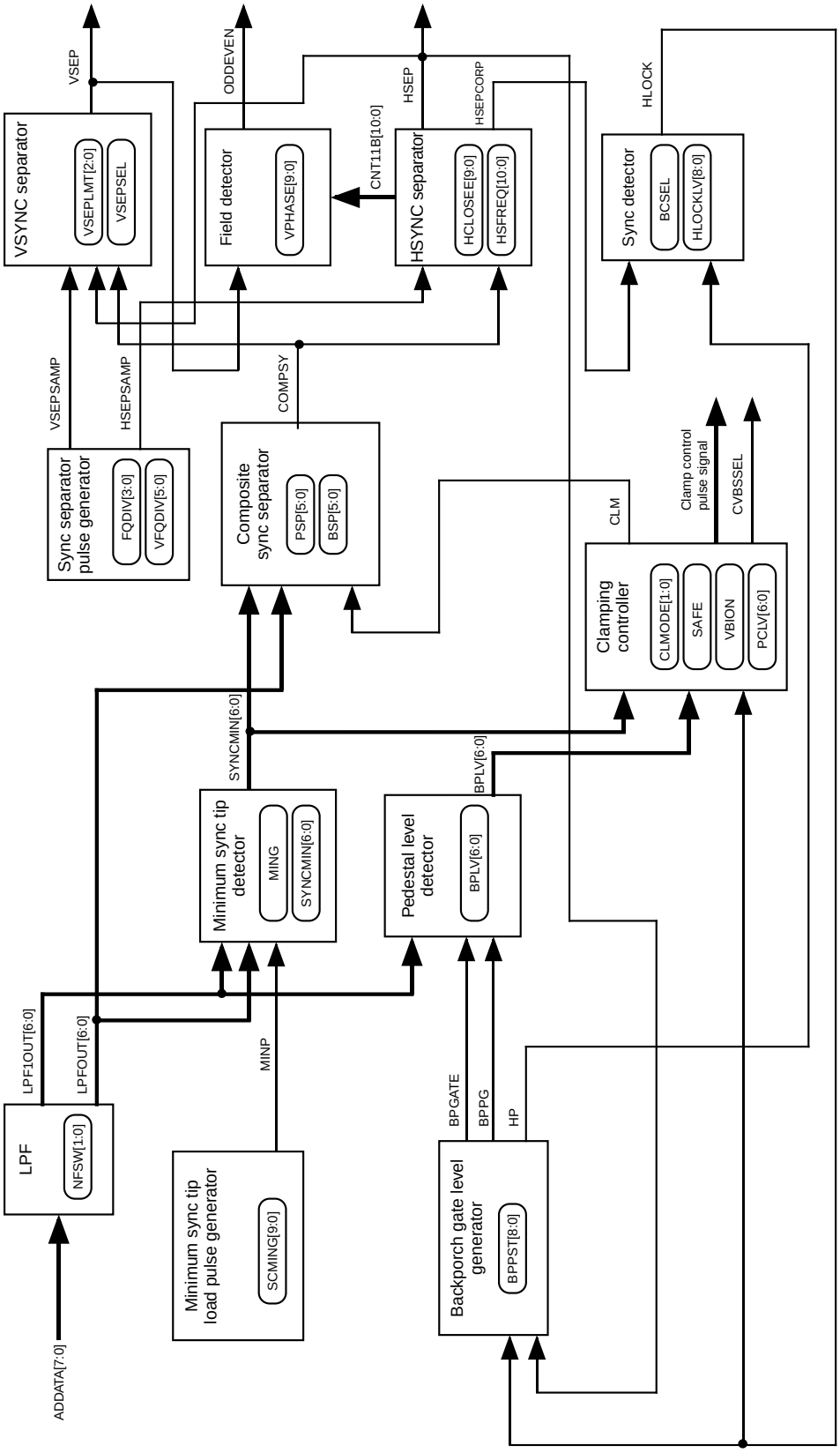


Figure 10-6 Sync Separator Circuit Block Diagram

Table 10-6 Control Registers for Sync Separator Circuit

Register	Page	CCD0 Address	CCD1 Address	Description
Register for setting the sync separator level				
SPLVH	193	x'03E4B'	x'03E6B'	Sync separator level set register high
SPLV	193	x'03E4A'	x'03E6A'	Sync separator level set register
Register for controlling the sync separator clock				
FQSELH	191	x'03E43'	x'03E63'	Frequency select register high
FQSEL	191	x'03E42'	x'03E62'	Frequency select register
Registers for controlling the HSYNC separator				
HSEP1H	195	x'03E4F'	x'03E6F'	HSYNC separator control register 1 high
HSEP1	195	x'03E4E'	x'03E6E'	HSYNC separator control register 1
HSEP2H	195	x'03E51'	x'03E71'	HSYNC separator control register 2 high
HSEP2	195	x'03E50'	x'03E70'	HSYNC separator control register 2
HLOCKLVH	196	x'03E55'	x'03E75'	Sync separator detection control register 1 high
HLOCKLV	196	x'03E54'	x'03E74'	Sync separator detection control register 1
HDISTWH	196	x'03E57'	x'03E77'	Sync separator detection control register 2 high
HDISTW	196	x'03E56'	x'03E76'	Sync separator detection control register 2
Register for controlling the VSYNC separator				
VCNTH	196	x'03E59'	x'03E79'	VSYNC separator control register high
VCNT	196	x'03E58'	x'03E78'	VSYNC separator control register
Register for controlling the field detection				
FIELDH	195	x'03E53'	x'03E73'	Field detection control register high
FIELD	195	x'03E52'	x'03E72'	Field detection control register
Register for monitoring the sync separator status				
HVCOND	196	x'03E5A'	x'03E7A'	Sync separator status register

10.3.3.1 HSYNC Separator

The HSYNC separator extracts the HSYNC signal from the composite sync signal using the sampling clock generated by the sync separator clock pulse generator. This circuit also secures and interpolates the HSYNC signal.

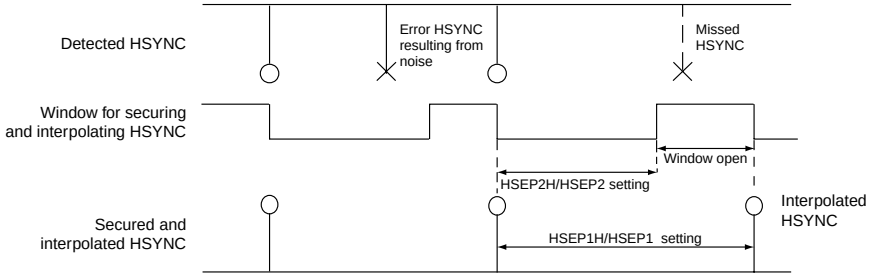


Figure 10-7 HSYNC Securement and Interpolation

As shown in figure 10-7, noise can cause the HSYNC detection circuit to both miss HSYNC pulses and add erroneous ones. The HSYNC separator contains a window circuit to correct these errors. The open and close timing for this window is set in the HSEP1H, HSEP1, HSEP2H, and HSEP2 registers. The unit used for the setting is the sampling clock for the HSYNC separator.

The circuit counts a corrected and interpolated HSYNC signal. If the count within the interval set in the HDISTWH and HDISTW registers is greater than that set in the HLOCKLVH and HLOCKLV registers, the HLOCK bit of HVCOND sets to 0, indicating an asynchronous state. This allows the device to determine the quality of the signal.

10.3.3.2 VSYNC Separator

The VSYNC separator extracts the VSYNC signal from the composite signal. Like the HSYNC separator, it contains programmable methods for eliminating noise. The VCNTH and VCNT registers contain these settings. Masking the 0H to 127H range (by setting the VSEPSEL bit of VCNTH to 0) prevents VSYNC errors due to noise. See figure 10-8.

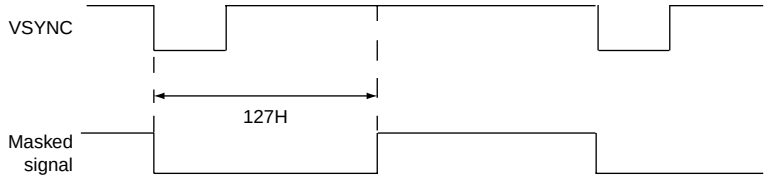


Figure 10-8 VSYNC Masking

10.3.3.3 Field Detection Circuit

The field detection circuit detects the phase difference between VSYNC and HSYNC, based on the setting in the VPHASE[9:0] field of the FIELDH and FIELD registers. This setting is in units of the sampling clock for the HSYNC separator. The results of the field detection are stored in the ODDEVEN bit of FIELDH.

10.3.4 Data Slicer

The data slicer contains the maximum and minimum detection circuits, the slice level calculator, and the slicer. The circuit compares the 8-bit digital values output from the ADC to the slice level, which can be calculated by the hardware or set in the software. It then outputs the results in serial 0s and 1s.

The data slicer calculates the slice level (the level above which a signal is 1 and below which it is 0) from the maximum and minimum clock run-in (CRI) pulses occurring in the interval between the settings in the CRI1SH/CRI1S and CRI1EH/CRI1E registers.

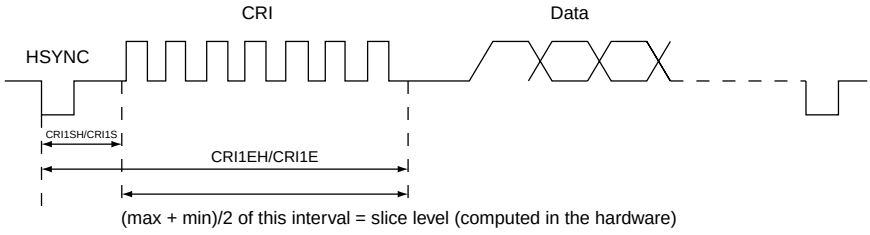


Figure 10-9 Data Slice Level Calculation

Table 10-7 provides the registers used to control and monitor the data slicer.

Table 10-7 Control Registers for Data Slicer

Register	Page	CCD0 Address	CCD1 Address	Description
CRI1SH	189	x'03E11'	x'03E31'	CRI capture start timing control register 1 high
CRI1S	189	x'03E10'	x'03E30'	CRI capture start timing control register 1
CRI1EH	189	x'03E13'	x'03E33'	CRI capture stop timing control register 1 high
CRI1E	189	x'03E12'	x'03E32'	CRI capture stop timing control register 1
MAX	186	x'03E03'	x'03E23'	CRI interval maximum register
MIN	186	x'03E02'	x'03E22'	CRI interval minimum register
SLSF	186	x'03E05'	x'03E25'	VBI data slice level register (software setting)
SLHD	186	x'03E04'	x'03E24'	VBI data slice level register (hardware calculated)
SLCNT	185	x'03E01'	x'03E21'	Slice level calculation control register

10.3.5 Controller and Sampling Circuit

The control circuit contains the CRI window generator and the caption data window generator. The sampling circuit extracts the 16-bit caption data (503 kHz) from the serial data output from the data slicer at the 14.32-MHz ADC sampling rate.

Table 10-8 provides the registers used to control and monitor these two blocks.

Table 10-8 Control Registers for Controller and Sampling Circuit

Register	Page	CCD0 Address	CCD1 Address	Description
Registers for Detecting CRI and Generating Sampling Clock				
CRI2SH	189	x'03E15'	x'03E35'	CRI capture start timing control register 2 high
CRI2S	189	x'03E14'	x'03E34'	CRI capture start timing control register 2
CRI2EH	189	x'03E17'	x'03E37'	CRI capture stop timing control register 2 high
CRI2E	189	x'03E16'	x'03E36'	CRI capture stop timing control register 2
CRI2FQW	188	x'03E0D'	x'03E2D'	CRI frequency width register A high
CRI1FQW	188	x'03E0C'	x'03E2C'	CRI frequency width register A low
CRI4FQW	188	x'03E0F'	x'03E2F'	CRI frequency width register B high
CRI3FQW	188	x'03E0E'	x'03E2E'	CRI frequency width register B low
Registers for Controlling Data Capture				
DATASH	190	x'03E19'	x'03E39'	Data capture start timing control register high
DATAS	190	x'03E18'	x'03E38'	Data capture start timing control register
DATAEH	190	x'03E1B'	x'03E3B'	Data capture stop timing control register high
DATAE	190	x'03E1A'	x'03E3A'	Data capture stop timing control register
CAPDATAH	188	x'03E0B'	x'03E2B'	Caption data capture register high
CAPDATA	188	x'03E0A'	x'03E2A'	Caption data capture register
HNUM	187	x'03E06'	x'03E26'	HSYNC count register
VBIIRQ	187	x'03E07'	x'03E27'	VBI interrupt timing control register

10.3.5.1 CRI Detection for Sampling Clock Generation

The decoder captures the caption data on the rising edge of the CRI pulse. To achieve this, it contains a circuit to accurately detect the CRI pulse rises and to generate a data sampling clock.

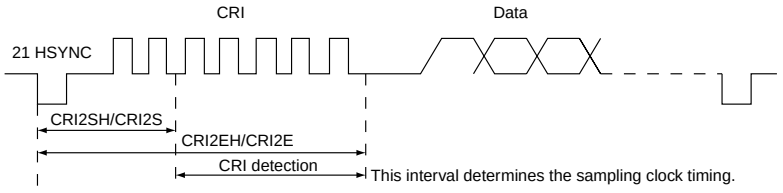


Figure 10-10 Sampling Clock Timing Determination

10.3.5.2 Data Capture Control

The DATASH/DATAS and DATAEH/DATAE registers control the data capture timing, and the CAPDATAH/CAPDATA registers store the caption data captured on the sampling clock generated through CRI detection.

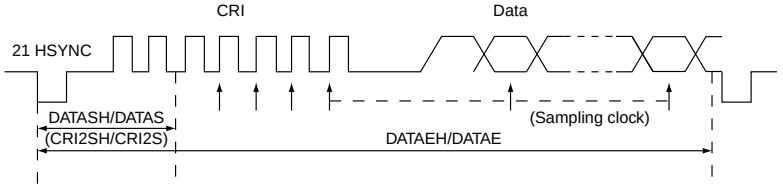


Figure 10-11 Caption Data Capture Timing

10.4 Closed-Caption Decoder Registers

Table 10-9 Closed-Caption Decoder Registers

Register	CCD0 Address	CCD1 Address	R/W	Description
FC	x'03E00'	x'03E20'	R/W	VBI decoding format select register
SLCNT	x'03E01'	x'03E21'	R/W	Slice level calculation control register
MAX	x'03E03'	x'03E23'	R	CRI interval maximum register
MIN	x'03E02'	x'03E22'	R	CRI interval minimum register
SLSF	x'03E05'	x'03E25'	R/W	VBI data slice level register (software)
SLHD	x'03E04'	x'03E24'	R/W	VBI data slice level register (hardware)
HNUM	x'03E06'	x'03E26'	R	HSYNC count register
VBIIRQ	x'03E07'	x'03E27'	R/W	VBI interrupt timing control register
ACQ1H	x'03E09'	x'03E29'	R/W	ACQ capture timing control register 1 high
ACQ1	x'03E08'	x'03E28'	R/W	ACQ capture timing control register 1
CAPDATAH	x'03E0B'	x'03E2B'	R	Caption data capture register high
CAPDATA	x'03E0A'	x'03E2A'	R	Caption data capture register
CRI2FQW	x'03E0D'	x'03E2D'	R	CRI frequency width register A high
CRI1FQW	x'03E0C'	x'03E2C'	R	CRI frequency width register A low
CRI4FQW	x'03E0F'	x'03E2F'	R	CRI frequency width register B high
CRI3FQW	x'03E0E'	x'03E2E'	R	CRI frequency width register B low
CRI1SH	x'03E11'	x'03E31'	R/W	CRI capture start timing control register 1 high
CRI1S	x'03E10'	x'03E30'	R/W	CRI capture start timing control register 1
CRI1EH	x'03E13'	x'03E33'	R/W	CRI capture stop timing control register 1 high
CRI1E	x'03E12'	x'03E32'	R/W	CRI capture stop timing control register 1
CRI2SH	x'03E15'	x'03E35'	R/W	CRI capture start timing control register 2 high
CRI2S	x'03E14'	x'03E34'	R/W	CRI capture start timing control register 2
CRI2EH	x'03E17'	x'03E37'	R/W	CRI capture stop timing control register 2 high
CRI2E	x'03E16'	x'03E36'	R/W	CRI capture stop timing control register 2
DATASH	x'03E19'	x'03E39'	R/W	Data capture start timing control register high
DATAS	x'03E18'	x'03E38'	R/W	Data capture start timing control register
DATAEH	x'03E1B'	x'03E3B'	R/W	Data capture stop timing control register high
DATAE	x'03E1A'	x'03E3A'	R/W	Data capture stop timing control register
STAPH	x'03E1D'	x'03E3D'	R/W	Sampling start position register (software setting) high
STAP	x'03E1C'	x'03E3C'	R/W	Sampling start position register (software setting)
FCPNUMH	x'03E1F'	x'03E3F'	R	Sampling start position register (hardware calculation) high
FCPNUM	x'03E1E'	x'03E3E'	R	Sampling start position register (hardware calculation)

Table 10-9 Closed-Caption Decoder Registers

Register	CCD0 Address	CCD1 Address	R/W	Description
NFSELH	x'03E41'	x'03E61'	R/W	Noise filter select register high
NFSEL	x'03E40'	x'03E60'	R/W	Noise filter select register
FQSELH	x'03E43'	x'03E63'	R/W	Frequency select register high
FQSEL	x'03E42'	x'03E62'	R/W	Frequency select register
SCMINGH	x'03E45'	x'03E65'	R/W	Minimum sync level detection interval set register high
SCMING	x'03E44'	x'03E64'	R/W	Minimum sync level detection interval set register
BPPSTH	x'03E47'	x'03E67'	R/W	Backporch position register high
BPPST	x'03E46'	x'03E66'	R/W	Backporch position register
SYNCMIN	x'03E48'	x'03E68'	R	Minimum sync level register
BPLV	x'03E49'	x'03E69'	R	Pedestal level register
SPLVH	x'03E4B'	x'03E6B'	R/W	Sync separator level set register high
SPLV	x'03E4A'	x'03E6A'	R/W	Sync separator level set register
CLAMPH	x'03E4D'	x'03E6D'	R/W	Clamping control register high
CLAMP	x'03E4C'	x'03E6C'	R/W	Clamping control register
HSEP1H	x'03E4F'	x'03E6F'	R/W	HSYNC separator control register 1 high
HSEP1	x'03E4E'	x'03E6E'	R/W	HSYNC separator control register 1
HSEP2H	x'03E51'	x'03E71'	R/W	HSYNC separator control register 2 high
HSEP2	x'03E50'	x'03E70'	R/W	HSYNC separator control register 2
FIELDH	x'03E53'	x'03E73'	R/W	Field detection control register high
FIELD	x'03E52'	x'03E72'	R/W	Field detection control register
HLOCKLVH	x'03E55'	x'03E75'	R/W	Sync separator detection control register 1 high
HLOCKLV	x'03E54'	x'03E74'	R/W	Sync separator detection control register 1
HDISTWH	x'03E57'	x'03E77'	R/W	Sync separator detection control register 2 high
HDISTW	x'03E56'	x'03E76'	R/W	Sync separator detection control register 2
VCNTH	x'03E59'	x'03E79'	R/W	VSYNC separator control register high
VCNT	x'03E58'	x'03E78'	R/W	VSYNC separator control register
HVCOND	x'03E5A'	x'03E7A'	R	Sync separator status register
CLPCND1H	x'03E5D'	x'03E7D'	R	Clamping control signal status register 1 high
CLPCND1	x'03E5C'	x'03E7C'	R	Clamping control signal status register 1
SLCNT1	x'03D7A'	x'03DFA'	R/W	Sampling frequency control register 1
SLCNT2	x'03D7B'	x'03DFB'	R/W	Sampling frequency control register 2

FC : VBI Decoding Format Select Register **x'03E00'**
(FCW: VBI Decoding Format Select Register **x'03E20')**

Bit:	7	6	5	4	3	2	1	0
	SLPUL SEL	CRIC SEL	NCRIG SEL	CNT STAP4	CNT STAP3	CNT STAP2	CNT STAP1	CNT STAP0
Reset:	0	0	0	0	1	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Always tie the bits in FC to fixed settings.

SLPULSEL: Polarity select for the CRI cycle transition detection
- 0: Detect 0 to 1 transitions
- 1: Detect 1 to 0 transitions

CRICSEL: Detection interval select for the CRI frequency width
- 0: CRI capture interval only
- 1: CRI capture interval and transition detection interval

NCRIGSEL: Sampling pulse generation interval
- 0: Disable the CRI capture interval
- 1: Enable the CRI capture interval

CNTSTAP[4:0]: Caption data sampling start position
Valid range: x'00' to x'1F'

SLCNT : Slice Level Calculation Control Register **x'03E01'**
(SLCNTW: Slice Level Calculation Control Register **x'03E21')**

Bit:	7	6	5	4	3	2	1	0
	FCP SEL	SYNC SEL	HCNT SEL1	HCNT SEL0	SLICE SEL	SLICE LD2	SLICE LD1	SLICE LD0
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Always tie the bits in SLCNT to fixed settings.

This register contains the settings for selecting either a hardware or software slice level. When you do not use a bit or field in this register, tie it to the setting indicated below.
For designs using the closed caption decoder, always tie bits 6 to 0.

FCPSEL: Hard/soft sampling start position select
- 0: Select hardware calculation
- 1: Select software setting (set in SFTSTAP[10:0] field of STAP)

SYNCSEL: Sync signal select (HSYNC/VSYNC)
For designs using the closed caption decoder, always tie it to 0.

HCNTSEL[1:0]: HSYNC count value select
When this field is unused, tie it to b'00'.
- 00: When odd, add 1 to the HSYNC count value
- 01: When even, add 1 to the HSYNC count value
- 10: No change to the HSYNC count value
- 11: Reserved

SLICESEL: Hard/soft slice level select
- 0: Select hardware calculation
- 1: Select software setting

SLICELD[2:0]: Slice level load timing select
When this field is unused, tie it to b'000'.
000: 1H100: 1 field
001: 2H101: 2 fields
010: 4H110: 4 fields
011: 8H111: 8 fields

MAX : CRI Interval Maximum Register **x'03E03'**
(MAXW: CRI Interval Maximum Register **x'03E23')**
MIN : CRI Interval Minimum Register **x'03E02'**
(MINW: CRI Interval Minimum Register **x'03E22')**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	MAX7	MAX6	MAX5	MAX4	MAX3	MAX2	MAX1	MAX0	MIN7	MIN6	MIN5	MIN4	MIN3	MIN2	MIN1	MIN0
Reset:	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
R/W:	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

MAX[7:0]: Maximum value during the CRI interval
Valid range: x'00' to x'FF'

MIN[7:0]: Minimum value during the CRI interval
Valid range: x'00' to x'FF'

SLSF : VBI Data Slice Level Register (Software) **x'03E05'**
(SLSFW: VBI Data Slice Level Register (Software) **x'03E25')**
SLHD : VBI Data Slice Level Register (Hardware) **x'03E04'**
(SLHDW: VBI Data Slice Level Register (Hardware) **x'03E24')**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	SLSF 7	SLSF 6	SLSF 5	SLSF 4	SLSF 3	SLSF 2	SLSF 1	SLSF 0	SLHD 7	SLHD 6	SLHD 5	SLHD 4	SLHD 3	SLHD 2	SLHD 1	SLHD 0
Reset:																
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R	R	R	R	R	R	R

SLSF[7:0]: VBI data slice level—software setting
Valid range: x'00' to x'FF'

SLHD[7:0]: VBI data slice level—hardware calculation
Valid range: x'00' to x'FF'

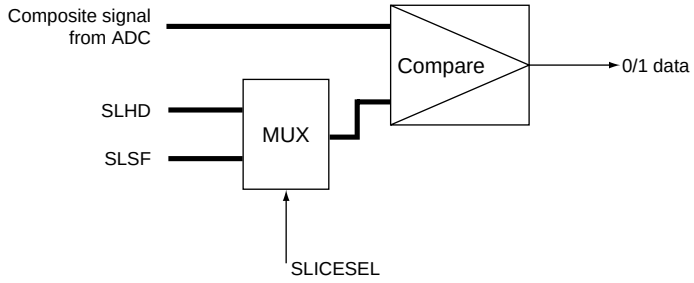


Figure 10-12 SLSF and SLHD Multiplexing

HNUM : HSYNC Count Register **x'03E06'**
(HNUMW: HSYNC Count Register **x'03E26')**

Bit:	7	6	5	4	3	2	1	0
	SB FLAG	—	—	HNUM4	HNUM3	HNUM2	HNUM1	HNUM0
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

SBFLAG: Start bit detection flag

0: No start bit detected

1: Start bit detected

HNUM[4:0]: HSYNC count during the VBI interval

This field indicates the H line number, from 0 to 25.

VBIIRQ : VBI Interrupt Timing Control Register **x'03E07'**
(VBIIRQW: VBI Interrupt Timing Control Register **x'03E27')**

Bit:	7	6	5	4	3	2	1	0
	—	—	—	VBIIRQ 4	VBIIRQ 3	VBIIRQ 2	VBIIRQ 1	VBIIRQ 0
Reset:	0	0	0	1	1	1	1	1
R/W:	R	R	R	R/W	R/W	R/W	R/W	R/W

This register allows you to time the interrupt occurring after the line 21 data capture to a line other than line 21.

VBIIRQ[4:0]: VBI interrupt timing control

In this field, set the H line number, from 0 to 25, for the VBI interrupt. You must set this field to x'13' or higher.

ACQ1H : ACQ Capture Timing Control Register 1 High **x'03E09'**
(ACQ1WH: ACQ Capture Timing Control Register 1 High **x'03E29')**
ACQ1 : ACQ Capture Timing Control Register 1 **x'03E08'**
(ACQ1W : ACQ Capture Timing Control Register 1 **x'03E28')**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	—	—	—	ACQ1 E4	ACQ1 E3	ACQ1 E2	ACQ1 E1	ACQ1 E0	—	—	—	ACQ1 S4	ACQ1 S3	ACQ1 S2	ACQ1 S1	ACQ1 S0
Reset:	0	0	0	1	1	1	1	1	0	0	0	1	1	1	1	1
R/W:	R	R	R	R/W	R/W	R/W	R/W	R/W	R	R	R	R/W	R/W	R/W	R/W	R/W

Always tie the bits in ACQ1 to fixed settings.

For designs using the closed caption decoder, always tie these registers to x'13' and x'12'.

ACQ1E[4:0]: Stop position for ACQ capture 1

Valid range: x'00' to x'25'

ACQ1S[4:0]: Start position for ACQ capture 1

Valid range: x'00' to x'25'

CAPDATAH : Caption Data Capture Register High **x'03E0B'**
(CAPDATAWH: Caption Data Capture Register High **x'03E2B')**
CAPDATA : Caption Data Capture Register **x'03E0A'**
(CAPDATAW : Caption Data Capture Register **x'03E2A')**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	CAPDA15	CAPDA14	CAPDA13	CAPDA12	CAPDA11	CAPDA10	CAPDA9	CAPDA8	CAPDA7	CAPDA6	CAPDA5	CAPDA4	CAPDA3	CAPDA2	CAPDA1	CAPDA0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

CAPDA[15:0]: Caption data

These registers stores the 16-bit captured caption data.

CRI2FQW : CRI Frequency Width Register A High **x'03E0D'**
(CRI2FQWW: CRI Frequency Width Register A High **x'03E2D')**
CRI1FQW : CRI Frequency Width Register A **x'03E0C'**
(CRI1FQWW: CRI Frequency Width Register A **x'03E2C')**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	CRI2FQW7	CRI2FQW6	CRI2FQW5	CRI2FQW4	CRI2FQW3	CRI2FQW2	CRI2FQW1	CRI2FQW0	CRI1FQW7	CRI1FQW6	CRI1FQW5	CRI1FQW4	CRI1FQW3	CRI1FQW2	CRI1FQW1	CRI1FQW0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

The CRI2FQW and CRI1FQW registers store the CRI cycles from rising edge to rising edge, for monitoring whether the CRIs were detected correctly during this period.

CRI2FQW[7:0]: CRI frequency width 2

This field indicates the width, in clock units, from the second from last to the third from last detected CRI rising edge.

CRI1FQW[7:0]: CRI frequency width 1

This field indicates the width, in clock units, from the last to the second from last detected CRI rising edge.

CRI4FQW : CRI Frequency Width Register B High **x'03E0F'**
(CRI4FQWW: CRI Frequency Width Register B High **x'03E2F')**
CRI3FQW : CRI Frequency Width Register B **x'03E0E'**
(CRI3FQWW: CRI Frequency Width Register B **x'03E2E')**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	CRI4FQW7	CRI4FQW6	CRI4FQW5	CRI4FQW4	CRI4FQW3	CRI4FQW2	CRI4FQW1	CRI4FQW0	CRI3FQW7	CRI3FQW6	CRI3FQW5	CRI3FQW4	CRI3FQW3	CRI3FQW2	CRI3FQW1	CRI3FQW0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

CRI4FQW[7:0]: CRI frequency width 4

This field indicates the width, in clock units, from the fourth from last to the fifth from last detected CRI rising edge.

CRI3FQW[7:0]: CRI frequency width 3

This field indicates the width, in clock units, from the third from last to the fourth from last detected CRI rising edge.

CRI1SH : CRI Capture Start Timing Control Register 1 High **x'03E11'**
(CRI1SWH: CRI Capture Start Timing Control Register 1 High **x'03E31')**
CRI1S : CRI Capture Start Timing Control Register 1 **x'03E10'**
(CRI1SW : CRI Capture Start Timing Control Register 1 **x'03E30')**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	—	—	—	—	—	CRI1S 10	CRI1S 9	CRI1S 8	CRI1S 7	CRI1S 6	CRI1S 5	CRI1S 4	CRI1S 3	CRI1S 2	CRI1S 1	CRI1S 0
Reset:	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1
R/W:	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

CRI1S[10:0]: Start position for CRI capture 1
Valid range: x'000' to x'7FF'

CRI1EH : CRI Capture Stop Timing Control Register 1 High **x'03E13'**
(CRI1EWH: CRI Capture Stop Timing Control Register 1 High **x'03E33')**
CRI1E : CRI Capture Stop Timing Control Register 1 **x'03E12'**
(CRI1EW : CRI Capture Stop Timing Control Register 1 **x'03E32')**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	—	—	—	—	—	CRI1E 10	CRI1E 9	CRI1E 8	CRI1E 7	CRI1E 6	CRI1E 5	CRI1E 4	CRI1E 3	CRI1E 2	CRI1E 1	CRI1E 0
Reset:	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1
R/W:	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

CRI1E[10:0]: Stop position for CRI capture 1
Valid range: x'000' to x'7FF'

CRI2SH : CRI Capture Start Timing Control Register 2 High **x'03E15'**
(CRI2SWH: CRI Capture Start Timing Control Register 2 High **x'03E35')**
CRI2S : CRI Capture Start Timing Control Register 2 **x'03E14'**
(CRI2SW : CRI Capture Start Timing Control Register 2 **x'03E34')**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	—	—	—	—	—	CRI2S 10	CRI2S 9	CRI2S 8	CRI2S 7	CRI2S 6	CRI2S 5	CRI2S 4	CRI2S 3	CRI2S 2	CRI2S 1	CRI2S 0
Reset:	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1
R/W:	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

CRI2S[10:0]: Start position for CRI capture 2
Valid range: x'000' to x'7FF'

CRI2EH : CRI Capture Stop Timing Control Register 2 High **x'03E17'**
(CRI2EWH: CRI Capture Stop Timing Control Register 2 High **x'03E37')**
CRI2E : CRI Capture Stop Timing Control Register 2 **x'03E16'**
(CRI2EW : CRI Capture Stop Timing Control Register 2 **x'03E36')**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	—	—	—	—	—	CRI2E 10	CRI2E 9	CRI2E 8	CRI2E 7	CRI2E 6	CRI2E 5	CRI2E 4	CRI2E 3	CRI2E 2	CRI2E 1	CRI2E 0
Reset:	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1
R/W:	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

CRI2E[10:0]: Stop position for CRI capture 2
Set this field so that the last CRI rising edge is included. The valid range is x'000' to x'7FF'.

DATASH : Data Capture Start Timing Control Register High **x'03E19'**
(DATASWH: Data Capture Start Timing Control Register High **x'03E39')**
DATAS : Data Capture Start Timing Control Register **x'03E18'**
(DATASW : Data Capture Start Timing Control Register **x'03E38')**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	—	—	—	—	—	DATAS 10	DATAS 9	DATAS 8	DATAS 7	DATAS 6	DATAS 5	DATAS 4	DATAS 3	DATAS 2	DATAS 1	DATAS 0
Reset:	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1
R/W:	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

DATAS[10:0]: Start position for data capture

Set this field to the same start position as that for CRI detection (set in CRI2S). The valid range is x'000' to x'7FF'.

DATAEH : Data Capture Stop Timing Control Register High **x'03E1B'**
(DATAEWH: Data Capture Stop Timing Control Register High **x'03E3B')**
DATAE : Data Capture Stop Timing Control Register **x'03E1A'**
(DATAEW : Data Capture Stop Timing Control Register **x'03E3A')**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	—	—	—	—	—	DATAE 10	DATAE 9	DATAE 8	DATAE 7	DATAE 6	DATAE 5	DATAE 4	DATAE 3	DATAE 2	DATAE 1	DATAE 0
Reset:	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1
R/W:	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

DATAE[10:0]: Stop position for data capture

Set this value high enough to allow the last data to be captured. The valid range is x'000' to x'7FF'.

STAPH : Sampling Start Position Register (Software Setting) High **x'03E1D'**
(STAPWH: Sampling Start Position Register (Software Setting) High **x'03E3D')**
STAP : Sampling Start Position Register (Software Setting) **x'03E1C'**
(STAPW : Sampling Start Position Register (Software Setting) **x'03E3C')**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	—	—	—	—	—	SFTSTAP 10	SFTSTAP 9	SFTSTAP 8	SFTSTAP 7	SFTSTAP 6	SFTSTAP 5	SFTSTAP 4	SFTSTAP 3	SFTSTAP 2	SFTSTAP 1	SFTSTAP 0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SFTSTAP[10:0]: Software setting for sampling start position (in clock units)

FCPNUMH : Sampling Start Position Register (Hardware Calculation) High **x'03E1F'**
(FCPNUMWH: Sampling Start Position Register (Hardware Calculation) High **x'03E3F')**
FCPNUM : Sampling Start Position Register (Hardware Calculation) **x'03E1E'**
(FCPNUMW: Sampling Start Position Register (Hardware Calculation) **x'03E3E')**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	—	—	—	—	—	FCPNUM 10	FCPNUM 9	FCPNUM 8	FCPNUM 7	FCPNUM 6	FCPNUM 5	FCPNUM 4	FCPNUM 3	FCPNUM 2	FCPNUM 1	FCPNUM 0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

FCPNUM[10:0]: Sampling start position calculated by the hardware

NFSELH : Noise Filter Select Register High **x'03E41'**
(NFSELWH: Noise Filter Select Register High **x'03E61')**
NFSEL : Noise Filter Select Register **x'03E40'**
(NFSELW : Noise Filter Select Register **x'03E60')**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	MING	—	—	—	—	—	—	NFSW 1	NFSW 0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R/W	R	R	R	R	R	R	R/W	R/W

These registers select the low-pass filter, which eliminates noise and high-frequency signals that are unnecessary to the sync separator and the clamping controller. The recommended settings for NFSELH and NFSEL are x'0' and x'00'.

MING: Output select for noise filter detecting minimum sync tip

0: Low-pass filter 1
1: Low-pass filter 2, 3, or 4 (set in NFSW[1:0])

NFSW[1:0]: Noise filter switch (for composite sync separator)

00: Low-pass filter 3
01: Low-pass filter 4
10: Low-pass filter 2
11: Low-pass filter 1

The cutoff frequencies for low-pass filters 1 to 4 are lower in ascending order, so that low-pass filter 4 eliminates the highest amount of noise.

FQSELH : Frequency Select Register High **x'03E43'**
(FQSELWH: Frequency Select Register High **x'03E63')**
FQSEL : Frequency Select Register **x'03E42'**
(FQSELW : Frequency Select Register **x'03E62'**)

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	—	—	VFQ DIV5	VFQ DIV4	VFQ DIV3	VFQ DIV2	VFQ DIV1	VFQ DIV0	—	—	—	—	FQ DIV3	FQ DIV2	FQ DIV1	FQ DIV0
Reset:	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	1
R/W:	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R	R	R	R	R/W	R/W	R/W	R/W

In these registers, set the sampling cycle for separating the HSYNC and VSYNC signals from the composite sync signal. The recommended settings are x'1F' and x'01'.

VFQDIV[5:0]: Sampling frequency setting for VSYNC separator

In this field, set the ratio by which to divide the sampling frequency for the HSYNC separator.

FQDIV[3:0]: Sampling frequency setting for HSYNC separator

In this field, set the ratio by which to divide the A/D sampling frequency.

SCMINGH : Minimum Sync Level Detection Interval Set Register High **x'03E45'**
(SCMINGWH: Minimum Sync Level Detection Interval Set Register High **x'03E65')**
SCMING : Minimum Sync Level Detection Interval Set Register **x'03E44'**
(SCMINGW : Minimum Sync Level Detection Interval Set Register **x'03E64')**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	SCMING9	SCMING8	SCMING7	SCMING6	SCMING5	SCMING4	SCMING3	SCMING2	SCMING1	SCMING0
Reset:	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SCMING[9:0]: Interval setting for the minimum sync level detection

Set the HSYNC cycle in this field in ADC clock units. This is the interval used for detecting the sync tip level for sync tip clamping. The valid range is x'000' to x'7FF'. Note that the HSYNC cycle set in this register is only used for detecting the minimum sync level. You must also set the correction HSYNC cycle in HSEP1H and HSEP1.

For the NTSC format, the settings for these registers are x'03' and x'8E', calculated as follows:

(A/D sampling frequency) × (HSYNC cycle) = 14.32 MHz × 63 µs = x'03' and x'8E'.

BPPSTH : Backporch Position Register High **x'03E47'**
(BPPSTWH: Backporch Position Register High **x'03E67')**
BPPST : Backporch Position Register **x'03E46'**
(BPPSTW : Backporch Position Register **x'03E66')**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	BPPST8	BPPST7	BPPST6	BPPST5	BPPST4	BPPST3	BPPST2	BPPST1	BPPST0
Reset:	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0
R/W:	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

BPPST[8:0]: Backporch start position for the leading edge of HSYNC

Use these registers to specify the position for capturing the pedestal level value used during pedestal clamping. Specify a number of ADC clocks after the leading edge of $\overline{\text{HSYNC}}$. The valid range is x'000' to x'1FF', and the recommended settings are x'00' and x'47'.

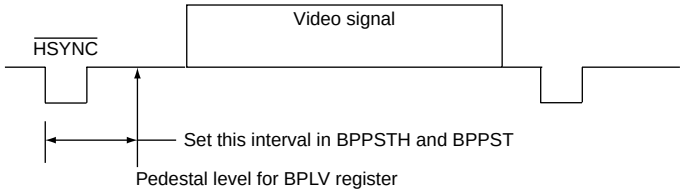


Figure 10-13 Backporch Position Setting

SYNCMIN : Minimum Sync Level Register x'03E48'
(SYNCMINW: Minimum Sync Level Register x'03E68')

Bit:	7	6	5	4	3	2	1	0
	—	SYNC MIN6	SYNC MIN5	SYNC MIN4	SYNC MIN3	SYNC MIN2	SYNC MIN1	SYNC MIN0
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

SYNCMIN[6:0]: Minimum sync level

This field stores the minimum level (the sync tip level) detected during the interval set in the SCMING register. For sync tip clamping, you should control clamping so as to make this value 16 (dec).

BPLV : Pedestal Level Register x'03E49'
(BPLVW: Pedestal Level Register x'03E69')

Bit:	7	6	5	4	3	2	1	0
	—	BPLV6	BPLV5	BPLV4	BPLV3	BPLV2	BPLV1	BPLV0
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

BPLV[6:0]: Pedestal level

This register stores the pedestal level captured from the position specified in BPPSTH and BPPST.

SPLVH : Sync Separator Level Set Register High x'03E4B'
(SPLVWH: Sync Separator Level Set Register High x'03E6B')
SPLV : Sync Separator Level Set Register x'03E4A'
(SPLVW : Sync Separator Level Set Register x'03E6A')

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	—	—	BSP5	BSP4	BSP3	BSP2	BSP1	BSP0	—	—	PSP5	PSP4	PSP3	PSP2	PSP1	PSP0
Reset:	0	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0
R/W:	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R	R	R/W	R/W	R/W	R/W	R/W	R/W

The sync separator uses the value set in these registers to separate the composite sync signal from the composite video signal.

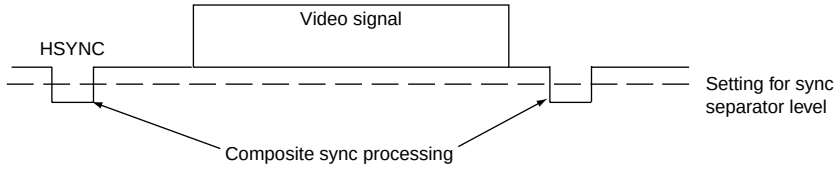


Figure 10-14 Sync Separator Level

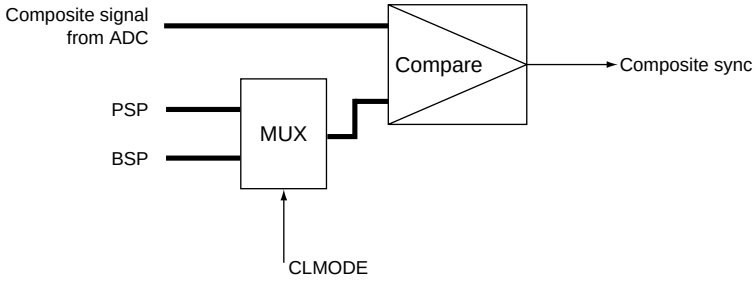


Figure 10-15 BSP and PSP Multiplexing

BSP[5:0]: Sync separator level for pedestal clamping

Sync separator level = (sync tip level/2) + BSP[5:0]. The valid range is x'00' to x'1F'.

PSP[5:0]: Sync separator level for sync tip clamping

Valid range: x'00' to x'1F'

CLAMPH : Clamping Control Register High x'03E4D'
(CLAMPWH: Clamping Control Register High x'03E6D')
CLAMP : Clamping Control Register x'03E4C'
(CLAMPW : Clamping Control Register x'03E6C')

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	—	PCLV6	PCLV5	PCLV4	PCLV3	PCLV2	PCLV1	PCLV0	SAFE	—	—	VBI ON	—	—	CL MODE 1	CL MODE 0
Reset:	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R	R/W	R	R	R/W	R/W

Use these registers to set the clamping mode (sync tip or pedestal clamping).

PCLV[6:0]: Pedestal clamping level setting

Set the reference level for pedestal clamping in this field. The valid range is x'00' to x'1F'.

VBION: VBI setting

0: VBI off
1: VBI on

SAFE: Clamping current source select

This bit is the capacitance switch for (5) and (6) in figure 10-5 on page 176.

0: High current source ((5) and (6) capacity high)
1: High current source ((5) and (6) capacity low)

CLMODE[1:0]: Clamping mode setting

00: Automatic switching (depends on the cycle state)
01: Sync tip clamping only
10: Pedestal clamping only
11: Clamping off

HSEP1H : HSYNC Separator Control Register 1 High **x'03E4F'**
(HSEP1WH: HSYNC Separator Control Register 1 High **x'03E6F')**
HSEP1 : HSYNC Separator Control Register 1 **x'03E4E'**
(HSEP1W : HSYNC Separator Control Register 1 **x'03E6E'**)

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	—	—	—	—	—	HS FREQ 10	HS FREQ 9	HS FREQ 8	HS FREQ 7	HS FREQ 6	HS FREQ 5	HS FREQ 4	HS FREQ 3	HS FREQ 2	HS FREQ 1	HS FREQ 0
Reset:	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0
R/W:	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

HSFREQ[10:0]: Correction HSYNC frequency

Set the correction HSYNC cycle in this field in HSYNC separator sampling clock units. The valid range is x'000' to x'7FF'.

HSEP2H : HSYNC Separator Control Register 2 High **x'03E51'**
(HSEP2WH: HSYNC Separator Control Register 2 High **x'03E71')**
HSEP2 : HSYNC Separator Control Register 2 **x'03E50'**
(HSEP2W : HSYNC Separator Control Register 2 **x'03E70'**)

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	H CLOSE E9	H CLOSE E8	H CLOSE E7	H CLOSE E6	H CLOSE E5	H CLOSE E4	H CLOSE E3	H CLOSE E2	H CLOSE E1	H CLOSE E0
Reset:	0	0	0	0	0	0	0	0	1	1	1	0	0	1	0	0
R/W:	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

HCLOSEE[9:0]: Start position for HSYNC detection

Set the position in HSYNC separator sampling clock units. The valid range is x'000' to x'3FF'.

FIELDH : Field Detection Control Register High **x'03E53'**
(FIELDWH: Field Detection Control Register High **x'03E73')**
FIELD : Field Detection Control Register **x'03E52'**
(FIELDW : Field Detection Control Register **x'03E72')**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	ODD EVEN	—	—	—	—	—	V PHASE 9	V PHASE 8	V PHASE 7	V PHASE 6	V PHASE 5	V PHASE 4	V PHASE 3	V PHASE 2	V PHASE 1	V PHASE 0
Reset:	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

ODDEVEN: Field detection signal

0: Odd field
1: Even field

VPHASE[9:0]: Phase difference setting for VSYNC and HSYNC

Set the phase difference in HSYNC separator sampling clock units. The valid range is x'000' to x'3FF'.

HLOCKLVH : Sync Separator Detection Control Register 1 High **x'03E55'**
(HLOCKLVWH: Sync Separator Detection Control Register 1 High **x'03E75')**
HLOCKLV : Sync Separator Detection Control Register 1 **x'03E54'**
(HLOCKLVW : Sync Separator Detection Control Register 1 **x'03E74')**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	HLOCK LV8	HLOCK LV7	HLOCK LV6	HLOCK LV5	HLOCK LV4	HLOCK LV3	HLOCK LV2	HLOCK LV1	HLOCK LV0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
R/W:	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

HLOCKLV[8:0]: Sync separator detection threshold

This value is compared to the count of the corrected HSYNC. Its valid range is x'000' to x'1FF' and recommended settings are x'00' and x'08'.

HLOCKLV ≤ HSYNC count → asynchronous
HLOCKLV > HSYNC count → synchronous

HDISTWH : Sync Separator Detection Control Register 2 High **x'03E57'**
(HDISTWWH: Sync Separator Detection Control Register 2 High **x'03E77')**
HDISTW : Sync Separator Detection Control Register 2 **x'03E56'**
(HDISTWW : Sync Separator Detection Control Register 2 **x'03E76')**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	HDIST W8	HDIST W7	HDIST W6	HDIST W5	HDIST W4	HDIST W3	HDIST W2	HDIST W1	HDIST W0
Reset:	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

HDISTW[8:0]: HSYNC count setting the interval for sync separation detection

In these registers, set the interval during which sync separation occurs. The valid range is x'000' to x'1FF'. (commend value : x'100')

VCNTH : VSYNC Separator Control Register High **x'03E59'**
(VCNTWH: VSYNC Separator Control Register High **x'03E79')**
VCNT : VSYNC Separator Control Register **x'03E58'**
(VCNTW : VSYNC Separator Control Register **x'03E78')**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	VSEP SEL	—	—	—	—	—	VSEP LMT2	VSEP LMT1	VSEP LMT0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1
R/W:	R	R	R	R	R	R	R	R/W	R	R	R	R	R	R/W	R/W	R/W

VSEPSEL: VSYNC signal select

0: 0H to 127H VSYNC separation mask
1: No mask

VSEPLMT[2:0]: VSYNC separation detection threshold

HVCOND: Sync Separator Status Register **x'03E5A'**
(HVCONDW: Sync Separator Status Register **x'03E7A')**

Bit:	7	6	5	4	3	2	1	0
	—	—	—	STPN	COMP SY	VSEP	HSEP	HLOCK
Reset:	0	0	0	1	1	0	0	0
R/W:	R	R	R	R	R	R	R	R

Use this register to monitor the status of the sync separator.

STPN: Status of clamping control pulse signal during STOP

COMPSY: Composite sync signal status

VSEP: VSYNC signal status

HSEP: HSYNC signal status

HLOCK: Sync detection

0: Asynchronous

1: Synchronous

CLPCND1H : Clamping Control Signal Status Register 1 High **x'03E5D'**
(CLPCND1WH: Clamping Control Signal Status Register 1 High **x'03E7D')**
CLPCND1 : Clamping Control Signal Status Register 1 **x'03E5C'**
(CLPCND1W : Clamping Control Signal Status Register 1 **x'03E7C')**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	—	SAFP	SAFN	CLPP	CLPN	XPED UP	XPE DOWN	PED UP	PE DOWN
Reset:	0	0	0	0	0	0	0	0	1	0	1	0	1	0	1	0
R/W:	R	R	R	R	R	R	R	R	R	R	R	R	R	R/W	R/W	R/W

These registers are for monitoring the status of the clamping current source switch shown in figure 10-5 on page 176. An N-channel transistor is on when the associated bit (PEDOWN, XPEDOWN, CLPN, or SAFEN) is 1. A P-channel transistor is on when the associated bit (PEDUP, XPEDUP, CLPP, or SAFEP) is 0.

SAFP: Clamping control pulse for large high current source (P-channel)

SAFN: Clamping control pulse for large high current source (N-channel)

CLPP: Clamping control pulse for small high current source (P-channel)

CLPN: Clamping control pulse for small high current source (N-channel)

XPEDUP: Clamping control pulse for medium current source (P-channel)

XPEDOWN: Clamping control pulse for medium current source (N-channel)

PEDUP: Clamping control pulse for low current source (P-channel)

PEDOWN: Clamping control pulse for low current source (N-channel)

SLCNT2 : Sampling Frequency Control Register 2 **x'03D7B'**
(SLCNT2W: Sampling Frequency Control Register 2 **x'03DFB')**
SLCNT1 : Sampling Frequency Control Register 1 **x'03D7A'**
(SLCNT1W: Sampling Frequency Control Register 1 **x'03DFA')**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	—	—	SL CNT2 5	SL CNT2 4	SL CNT2 3	SL CNT2 2	SL CNT2 1	SL CNT2 0	—	—	SL CNT1 5	SL CNT1 4	SL CNT1 3	SL CNT1 2	SL CNT1 1	SL CNT1 0
Reset:	0	0	0	1	0	1	1	1	0	0	0	0	0	0	0	0
R/W:	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R	R	R/W	R/W	R/W	R/W	R/W	R/W

SLCNT2[5:0], SLCNT1[5:0]: Select the sampling frequency when the operating frequency changes.

These registers allow you to keep the sampling frequency for closed-caption data from changing when the operating frequency changes. The table below shows how to set SLCNT2 and SLCNT1 for particular operating

frequencies. Note that when you change the operating frequency, you must align other registers besides the VBI.

Table 10-10 Sampling Frequency Control Register Settings

Operating Frequency	12 MHz	14 MHz	14.32 MHz
SLCNT2	x’16’	x’1A’	x’1B’
SLCNT1	x’1B’	x’1B’	x’0F’

11 IR Remote Signal Receiver

11.1 Description

The MN101C46F contains a remote signal receiver that processes signals in two formats: Household Electrical Appliance Manufacturers Association (HEAMA) format and 5-/6-bit format. This chapter provides an overview of each block in the circuit and describes the operation of the receiver.

f_{SYSCLK} = 3.58 MHz in all of the examples and descriptions in this section.

The remote signal is input through the RMIN pin. Each time the edge detection circuit detects the active edge of the signal, the 6-bit counter resets and the sampling clock, T_S, starts counting. T_S is formed by dividing PWM2 by the value in the frequency division control register, RMTC. The clock status register, RMCS, which can be read at any time, holds the current value of the 6-bit counter.

The remote signal contains a leader, data, and a trailer, in that order. The micro-controller shifts received data into the LSB of the reception data shift register, RMSR. After it receives eight bits, it loads the contents of RMSR to the reception data transfer register, RMTR.

11.2 Block Diagram

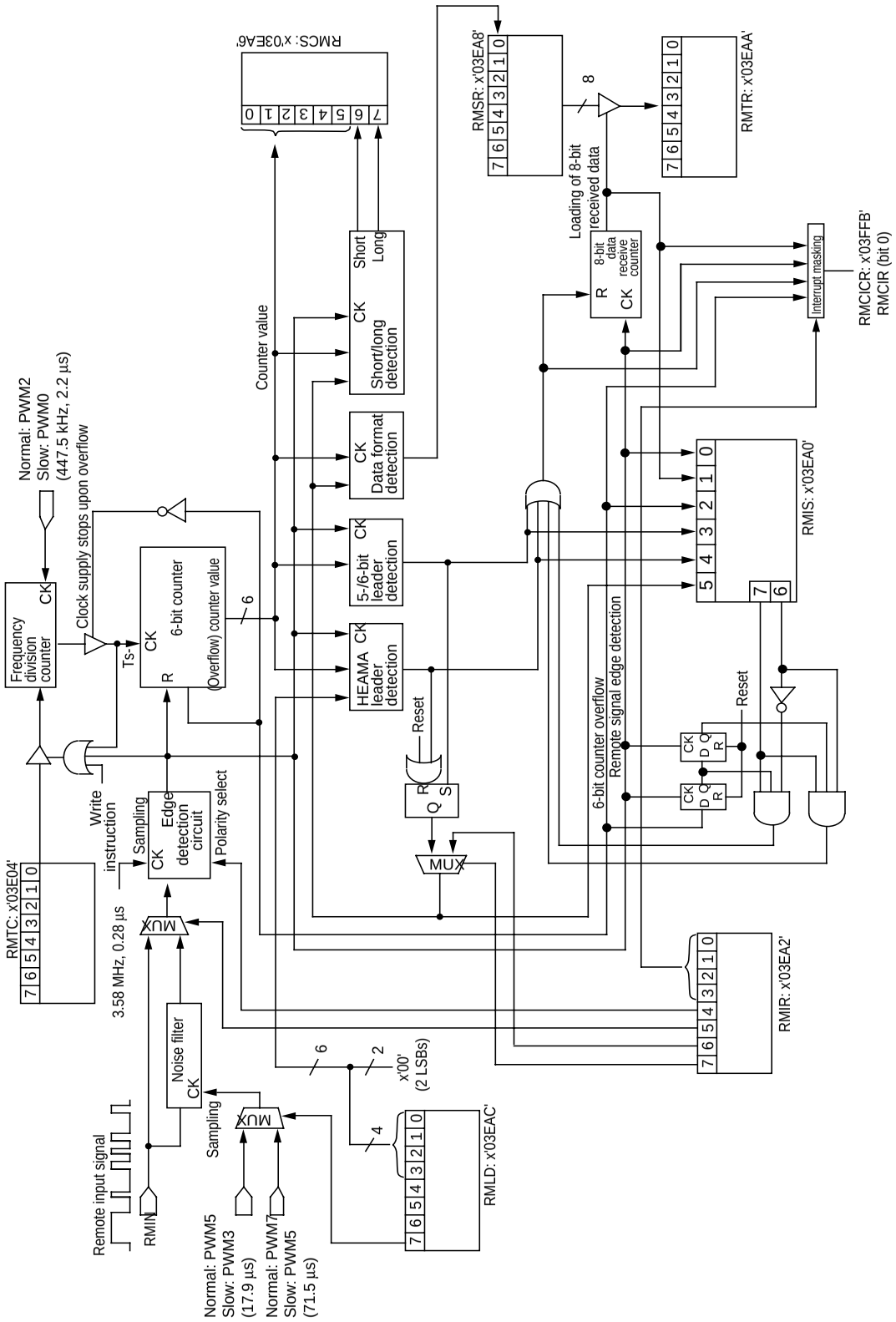


Figure 11-1 IR Remote Signal Receiver Block Diagram

11.3 IR Remote Signal Receiver Operation

11.3.1 Operating Modes

The IR remote signal receiver has three operating modes: HEAMA, 5-/6-bit, and HEAMA−5-/6-bit automatic detect. Set the mode in the MODAUTO and MODSEL bits of the interrupt control register, RMIR. The FMTMON bit of the interrupt status register, RMIS, monitors the operating mode.

In automatic detect mode, the microcontroller checks the interval between remote signal edges. If the interval is (n - $4T_S$) to (n + $3T_S$), where n is the leader value set in the LD[3:0] field of the RMLD register, it processes the data in HEAMA format. If the interval is 28 to 35 T_S cycles, it processes the data in 5-/6-bit format.

11.3.2 Noise Filter

The IR remote signal receiver contains a noise filter to eliminate noise from the remote signal. To enable the noise filter, set the FILTRE bit of the interrupt control register, RMIR, to 1. The noise filter samples the remote input signal every PWM5 cycle (17.9 µs) or PWM7 cycle (71.5 µs), then outputs the value that it sampled at least three times during the last four sampling cycles. This eliminates any noise occurring during one or two sampling cycles. Select the sampling clock (PWM5 or PWM7) with the SP bit of the RMLD register. (PWM5 is selected at reset.)

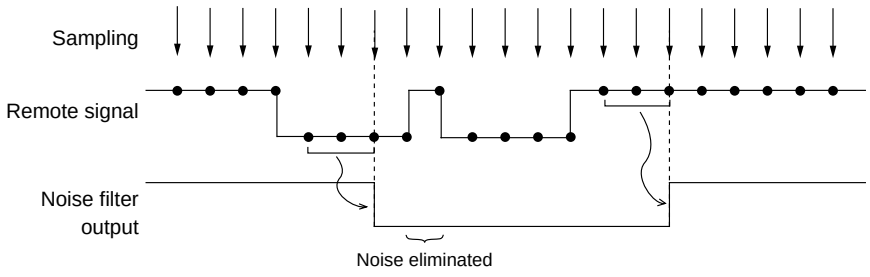


Figure 11-2 IR Remote Signal Noise Filtering

11.3.3 8-Bit Data Reception

Resetting the 8-bit data reception counter allows the microcontroller to receive 8-bit data, either with or without a leader. The software can reset the counter using the BCRSTE and BCEDGS bits of the interrupt status register, RMIS. You can also reset the counter with an external reset or a hardware reset at leader detection.

Set BCRSTE to enable resets to the 8-bit data reception counter. When the BCEDGS bit is 0, the counter resets at the first remote signal edge after each trailer detection. This mode is for data containing no leader. (See figure 11-3.)

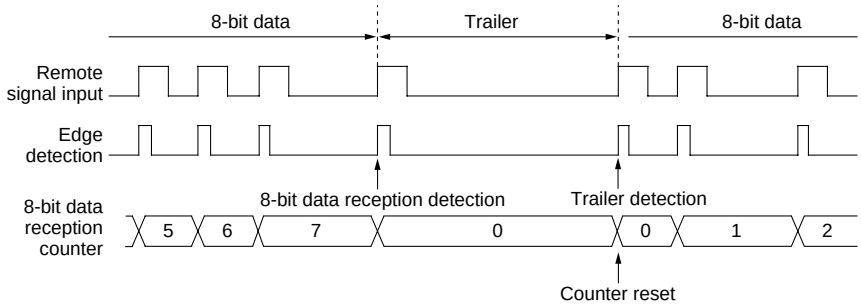


Figure 11-3 Reception of 8-Bit Data with No Leader

When BCEDGS is 1, the counter resets at the second remote signal edge after each trailer detection. By ignoring the leader, this mode allows the microcontroller to receive 8-bit data that contains a leader. (See figure 11-4.)

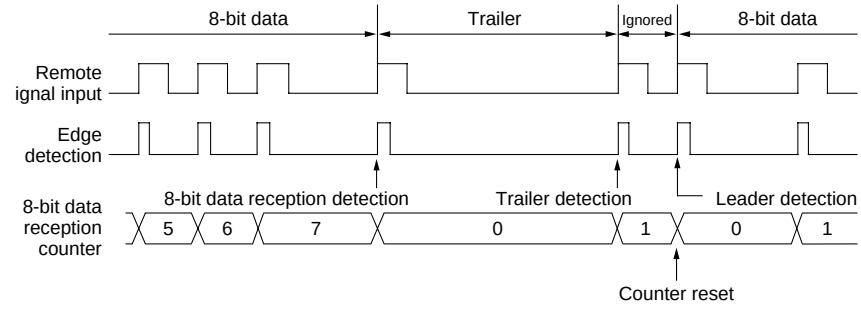


Figure 11-4 Reception of 8-Bit Data with Leader

11.3.4 Identifying the Data Format

The microcontroller determines the logic levels of the data by testing the interval between remote signal edges. Table 11-1 shows the intervals that the microcontroller interprets as 0 and 1 for both HEAMA and 5-/6-bit formats. Table 11-2 shows the conditions for identifying long and short data.

Table 11-1 Logic Level Conditions for Data Formats

Operating Mode	Logic Level Conditions	
	Data = 0	Data = 1
HEAMA format	< 6 T_S cycles	≥ 6 T_S cycles
5-/6-bit format	< 12 T_S cycles	≥ 12 T_S cycles

Table 11-2 Long and Short Data Identification

Operating Mode	Long Data	Short Data
HEAMA format	≥ 10 T_S cycles	< 2 T_S cycles
5-/6-bit format	≥ 20 T_S cycles	< 4 T_S cycles

The 6-bit counter regulates data format detection. When the microcontroller detects a data leader, it sets the LONGDF bit of the clock status register, RMCS, to indicate long data. Figure 11-5 is a graphic representation of all the conditions for identifying the data format.

When the microcontroller detects a data trailer, the hardware automatically shuts off the supply to sampling clock T_S, which the 6-bit counter counts. The counter resets and the clock supply restarts at the next edge detection.

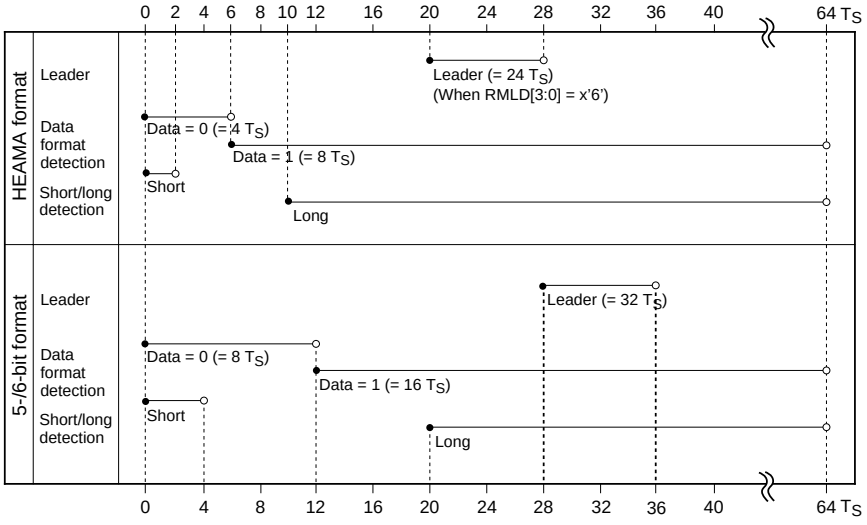


Figure 11-5 Conditions for Detecting Data Formats

11.3.5 Generating Interrupts

The IR remote signal receiver has four interrupt vectors: leader detection, trailer detection, 8-bit data reception detection, and pin edge detection. This section describes the operation for each of them.

11.3.5.1 Leader Detection

An interrupt occurs when the circuit detects a data leader. It detects leaders by testing the interval between remote signal edges. Table 11-3 shows the conditions.

Table 11-3 Leader Detection Conditions

Format	Edge Interval
HEAMA data leader	$(n - 4)T_S \leq$ interval $< (n + 4)T_S$ (1)
5-/6-bit data leader	$28T_S \leq$ interval $< 36T_S$

Note: 1. *n* = the leader value set in LD[3:0] of the RMLD register.

11.3.5.2 Trailer Detection

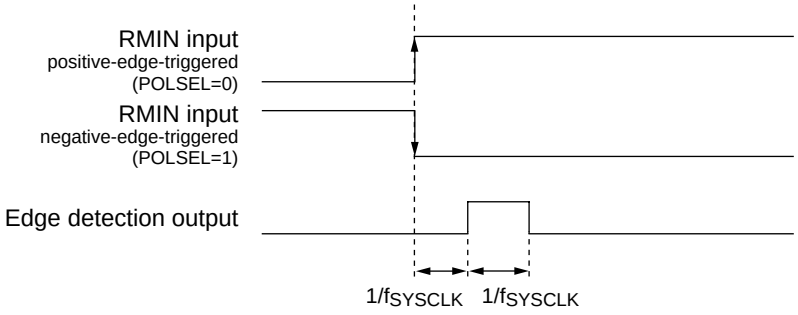
An interrupt occurs when the 6-bit counter overflows.

11.3.5.3 8-Bit Data Reception Detection

An interrupt occurs when the microcontroller loads 8-bit received data to the reception data transfer register, RMTR.

11.3.5.4 Pin Edge Detection

An interrupt occurs when the remote signal input pin, RMIN, is asserted. The POLSEL bit of RMIR sets the polarity of RMIN.



Note: $1/f_{SYSCLK} = 1/3.58$ MHz = 0.28 µs

Figure 11-6 Pin Edge Detection

The detection output for all four interrupt vectors is an active high pulse asserted at intervals of $1/f_{SYSCLK}$. Bits 3 to 0 of the RMIR register control the interrupt vectors individually. A 0 disables the interrupt vector and a 1 enables it.

A remote signal interrupt sets the RMCIR flag of the RMCICL interrupt register (x'00FC76').

11.4 IR Remote Signal Receiver Control Registers

Table 11-4 IR Remote Signal Receiver Registers

Register	Address	R/W	Function
RMTC	x'03EA4'	R/W	Remote signal frequency division control register
RMIR	x'03EA2'	R/W	Remote signal interrupt control register
RMIS	x'03EA0'	R/W	Remote signal interrupt status register
RMLD	x'03EAC'	R/W	Remote signal leader value set register
RMCS	x'03EA6'	R	Remote signal clock status register
RMSR	x'03EA8'	R	Remote signal reception data shift register
RMTR	x'03EAA'	R	Remote signal reception data transfer register

RMTC: Remote Signal Frequency Division Control Register **x'03EA4'**

Bit:	7	6	5	4	3	2	1	0
	RMTC7	RMTC6	RMTC5	RMTC4	RMTC3	RMTC2	RMTC1	RMTC0
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

To identify the remote signal, the IR signal receiver generates a sampling clock, T_S, by dividing the PWM2 pulse by the value set in RMTC[7:0]. f_{PWM2} is f_{SYSCLK} divided by 2^3 (= 447.5 kHz and 2.2 µs with a 4-MHz oscillator). The T_S cycle is the contents of RMTC + 1 in normal mode or the contents of RMTC + 2 in slow mode, so load a value from 1 to 255 to set a division ratio from 2 to 256.

The microcontroller reads the value in the frequency division counter as a ones' complement number (each digit is complemented).

Set the RMTC value so that $T_S = T/2$, where T is the pulse width of the remote input signal. Table 11-5 shows how to define T for the different formats.

The edge detection circuit samples the remote signal with f_{SYSCLK}. Set the frequency divide-by ratio to meet this condition. If you do not, the microcontroller may interpret the data 1s and 0s incorrectly.

After the program sets the divide-by ratio for the frequency in RMTC, the read values may be incorrect until the circuit detects the next active edge of the remote signal.

Table 11-5 HEAMA and 5-/6-Bit Data Pulse Widths

HEAMA format	H L Data = 0: T T Data = 1: T 3T
5-/6-bit format	Data = 0: 2T 2T Data = 1: 2T 6T

RMTC is an 8-bit access register.

RMIR: Remote Signal Interrupt Control Register **x'03EA2'**

Bit:	7	6	5	4	3	2	1	0
	MOD AUTO	MOD SEL	FILTR E	POL SEL	LEADER E	TRAILR E	DAT8 E	EDME E
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

RMIR controls the operating modes and interrupt operations for the receiver circuit. It is an 8-bit access register.

MODAUTO: Automatic operating mode detection on/off
- 0: Automatic detect
- 1: Fixed

MODSEL: Operating mode select
- 0: HEAMA format
- 1: 5-/6-bit format

FILTRE: Noise filter input multiplexer on/off
- 0: Pin level
- 1: Noise filter

POLSEL: Input polarity
- 0: Positive-edge triggered
- 1: Negative-edge triggered

LEADERE: Interrupt enable for leader detection
- 0: Disable
- 1: Enable

TRAILRE: Interrupt enable for trailer detection
- 0: Disable
- 1: Enable

DAT8E: Interrupt enable for 8-bit data reception detection
- 0: Disable
- 1: Enable

EDMEE: Interrupt enable for RMIN pin edge detection
- 0: Disable
- 1: Enable

RMIS: Remote Signal Interrupt Status Register **x'03EA0'**

Bit:	7	6	5	4	3	2	1	0
	BC RSTE	BC EDGS	FMT MON	DOMES D	M56BIT D	TRAILR D	DAT8 D	EDGE D
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W

RMIR indicates the detection and operation status of remote signal interrupts. It is an 8-bit access register.

BCRSTE: 8-bit data reception binary counter reset enable

0: Disable

1: Enable

BCEDGS: 8-bit data reception binary counter reset edge select

0: Reset at 1st edge

1: Reset at 2nd edge

FMTMON: Format monitor

0: HEAMA format

1: 5-/6-bit format

DOMESD: Interrupt request on HEAMA leader detection

0: No request

1: Request

M56BITD: Interrupt request on 5-/6-bit leader detection

0: No request

1: Request

TRAILRD: Interrupt request on trailer detection

0: No request

1: Request

DAT8D: Interrupt request on 8-bit reception detection

0: No request

1: Request

EDGED: Interrupt request on RMIN pin edge detection

0: No request

1: Request

RMLD: Remote Signal Leader Value Set Register **x'03EAC'**

Bit:	7	6	5	4	3	2	1	0
	SP	—	—	—	LD3	LD2	LD1	LD0
Reset:	0	0	0	0	0	1	1	0
R/W:	R/W	R	R	R	R/W	R/W	R/W	R/W

RMLD is an 8-bit access register.

SP: Noise filter sampling cycle

0: PWM5 (17.9 µs) ($f_{PWM5} = f_{SYSCLK}/2^6$)

1: PWM7 (71.5 µs) ($f_{PWM7} = f_{SYSCLK}/2^8$)

LD[3:0]: HEAMA data leader value

Set the four MSBs of the 6-bit leader value for HEAMA data in LD[3:0]. This 4-bit setting must be between 0 and 63 T_S cycles. The default value is x'6'. The two LSBs of the leader are always 0.

Do not set the leader value too small. Leader detection and data detection may occur simultaneously.

RMCS: Remote Signal Clock Status Register **x'03EA6'**

Bit:	7	6	5	4	3	2	1	0
	LONG DF	SHORT DF	TSCNT5	TSCNT4	TSCNT3	TSCNT2	TSCNT1	TSCNT0
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R	R	R	R	R	R

RMCS indicates the result of the short/long data detection. It is an 8-bit access register.

LONGDF: Long data format detection

Set to 1 when long data is detected.

SHORTDF: Short data format detection

Set to 1 when short data is detected.

TSCNT[5:0]: 6-bit counter value

RMSR: Remote Signal Reception Data Shift Register **x'03EA8'**

Bit:	7	6	5	4	3	2	1	0
	RMSR7	RMSR6	RMSR5	RMSR4	RMSR3	RMSR2	RMSR1	RMSR0
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

RMTR: Remote Signal Reception Data Transfer Register **x'03EAA'**

Bit:	7	6	5	4	3	2	1	0
	RMTR7	RMTR6	RMTR5	RMTR4	RMTR3	RMTR2	RMTR1	RMTR0
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

The microcontroller shifts received data into RMSR, converting it to parallel data. After it shifts in 8 bits, it loads the data byte to RMTR. The CPU reads the data from RMTR. The data shifts from LSB to MSB.

RMSR and RMTR are 8-bit access registers.

12 ROM Correction

12.1 Description

The ROM correction function can correct the program data in any address within the 96-kilobyte ROM. (It cannot correct OSD ROM data.) A maximum of sixteen addresses can be corrected. Addresses are set as address match interrupts. This function shortens time-to-market for large-scale designs, since changes can be implemented in the software after the mask ROM is complete.

The ROM correction function has numerous other applications. For instance, you can insert keywords into the functional routines, then use the function to send internal status information to an external location. This enables system-level examination of the internal status even with the mask ROM version.

To use the ROM correction function, embed a routine such as that shown in figure 12-2 in the ROM.

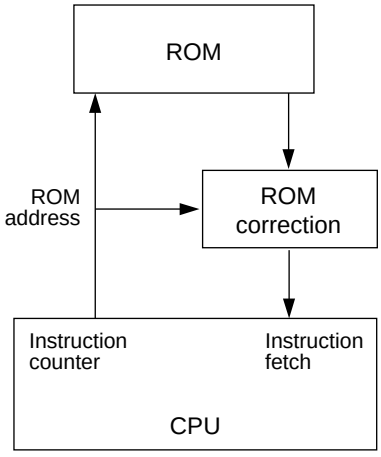


Figure 12-1 ROM Area Schematic Diagram

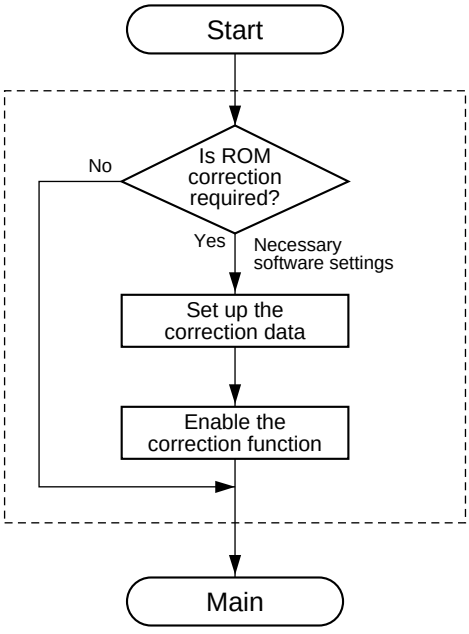


Figure 12-2 ROM Correction Flow

As figure 12-1 shows, the function lies between the microcontroller and ROM blocks. First set the correction data for any sixteen non-OSD addresses in the ROM correction address match and data registers. (Follow the flow shown in figure 12-2.) Once this is done, the circuit will correct the ROM output for the designated addresses.

12.2 Block Diagram

Figure 12-3 is a block diagram of the ROM correction circuit. A match detection circuit constantly monitors the ROM address specified by the CPU instruction pointer (IP). When the value matches a correction address, the circuit replaces the data output from the ROM with the data in the appropriate correction data register. It then sends the corrected data to the CPU.

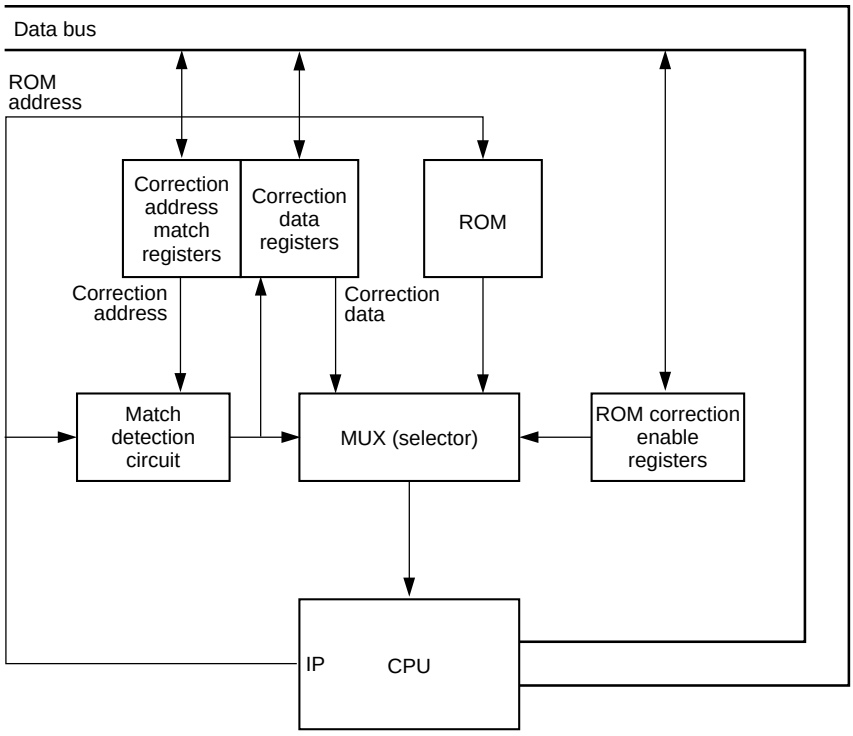


Figure 12-3 ROM Correction Block Diagram

12.3 Programming Considerations

At reset, the ROM correction address match and data registers contain all 0s. Since a reset also disables ROM correction (in ROMCEN), the ROM will still operate normally.

Only read from or write to the address match registers while ROM correction is disabled in ROMCEN. Otherwise, an error may occur in the match detection circuit.

Note that the address match and data registers only allow full-register access (eight bits). You cannot write to individual bits.

The ROM correction function cannot be emulated in ICE mode.

12.4 ROM Correction Control Registers

Table 12-1 shows the organization of the address match and data registers for ROM correction. Write a ROM address to be corrected to an AMCHIHn, AMCHIMn, and AMCHILn register trio and write the corrected data to the associated CHDATn register. Enable ROM correction for the associated address in the ROMCEN register.

Table 12-1 ROM Correction Address Match and Data Registers

ROM Address	Address Match Register			Data Register
	High	Middle	Low	
Address 0	AMCHIH0 x'03F90'	AMCHIM0 x'03F71'	AMCHIL0 x'03F70'	CHDAT0 x'03FD0'
Address 1	AMCHIH1 x'03F91'	AMCHIM1 x'03F73'	AMCHIL1 x'03F72'	CHDAT1 x'03FD1'
Address 2	AMCHIH2 x'03F92'	AMCHIM2 x'03F75'	AMCHIL2 x'03F74'	CHDAT2 x'03FD2'
Address 3	AMCHIH3 x'03F93'	AMCHIM3 x'03F77'	AMCHIL3 x'03F76'	CHDAT3 x'03FD3'
Address 4	AMCHIH4 x'03F94'	AMCHIM4 x'03F79'	AMCHIL4 x'03F78'	CHDAT4 x'03FD4'
Address 5	AMCHIH5 x'03F95'	AMCHIM5 x'03F7B'	AMCHIL5 x'03F7A'	CHDAT5 x'03FD5'
Address 6	AMCHIH6 x'03F96'	AMCHIM6 x'03F7D'	AMCHIL6 x'03F7C'	CHDAT6 x'03FD6'
Address 7	AMCHIH7 x'03F97'	AMCHIM7 x'03F7F'	AMCHIL7 x'03F7E'	CHDAT7 x'03FD7'
Address 8	AMCHIH8 x'03F98'	AMCHIM8 x'03F81'	AMCHIL8 x'03F80'	CHDAT8 x'03FD8'
Address 9	AMCHIH9 x'03F99'	AMCHIM9 x'03F83'	AMCHIL9 x'03F82'	CHDAT9 x'03FD9'
Address 10	AMCHIHA x'03F9A'	AMCHIMA x'03F85'	AMCHILA x'03F84'	CHDATA x'03FDA'
Address 11	AMCHIHB x'03F9B'	AMCHIMB x'03F87'	AMCHILB x'03F86'	CHDATB x'03FDB'
Address 12	AMCHIHC x'03F9C'	AMCHIMC x'03F89'	AMCHILC x'03F88'	CHDATC x'03FDC'
Address 13	AMCHIHD x'03F9D'	AMCHIMD x'03F8B'	AMCHILD x'03F8A'	CHDATD x'03FDD'
Address 14	AMCHIHE x'03F9E'	AMCHIME x'03F8D'	AMCHILE x'03F8C'	CHDATE x'03FDE'
Address 15	AMCHIHF x'03F9F'	AMCHIMF x'03F8F'	AMCHILF x'03F8E'	CHDATF x'03FDF'

Note: All registers reset to 0.

ROMCENH: ROM Correction Enable Register High **x'03F3F'**
ROMCEN: ROM Correction Enable Register **x'03F3E'**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	RCENH7	RCENH6	RCENH5	RCENH4	RCENH3	RCENH2	RCENH1	RCENH0	RCEN7	RCEN6	RCEN5	RCEN4	RCEN3	RCEN2	RCEN1	RCEN0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

RCENHn: Address n ROM correction enable (n = 15–8)

0: Disable

1: Enable

RCENn: Address n ROM correction enable (n = 7–0)

0: Disable

1: Enable

AMCHIHn: ROM Correction Address Match Register n (High) **x'03F90' to x'03F9F'**

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	CHAn 17	CHAn 16
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R/W	R/W

AMCHIHn are 8-bit access registers. (n = 0–F)

CHAn[17:16]: Correction address bits A17 to A16 (A17 = MSB)

AMCHIMn: ROM Correction Address Match Register n (Middle) **x'03F71' to x'03F8F'**

Bit:	7	6	5	4	3	2	1	0
	CHAn 15	CHAn 14	CHAn 13	CHAn 12	CHAn 11	CHAn 10	CHAn 9	CHAn 8
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

AMCHIMn are 8-bit access registers. (n = 0–F)

CHAn[15:8]: Correction address bits A15 to A8

AMCHILn: ROM Correction Address Match Register n (Low) **x'03F70' to x'03F8E'**

Bit:	7	6	5	4	3	2	1	0
	CHAn 7	CHAn 6	CHAn 5	CHAn 4	CHAn 3	CHAn 2	CHAn 1	CHAn 0
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

AMCHILn are 8-bit access registers. (n = 0–F)

CHAn[7:0]: Correction address bits A7 to A0

CHDAT0n: ROM Correction Data Register n **x'03FD0' to x'03FDF'**

Bit:	7	6	5	4	3	2	1	0
	CHn7	CHn6	CHn5	CHn4	CHn3	CHn2	CHn1	CHn0
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

CHDATn are 8-bit access registers. (n = 0–F)

CHn7: Correction data D7

CHn6: Correction data D6

CHn5: Correction data D5

CHn4: Correction data D4

CHn3: Correction data D3

CHn2: Correction data D2

CHn1: Correction data D1

CHn0: Correction data D0

13 I²C Bus Controller

13.1 Description

The MN101C46F contains one I²C bus controller, fully compliant with the I²C specification, that can control one of two I²C bus connections.

An I²C bus is a simple, two-wire bus for transferring data between ICs. Since it requires only a serial data line (SDA) and a serial clock line (SCL), it minimizes interconnections so ICs have fewer pins and there are fewer PCB tracks. The result is a smaller and less expensive PCB. Figure 13-1 shows a typical I²C bus application.

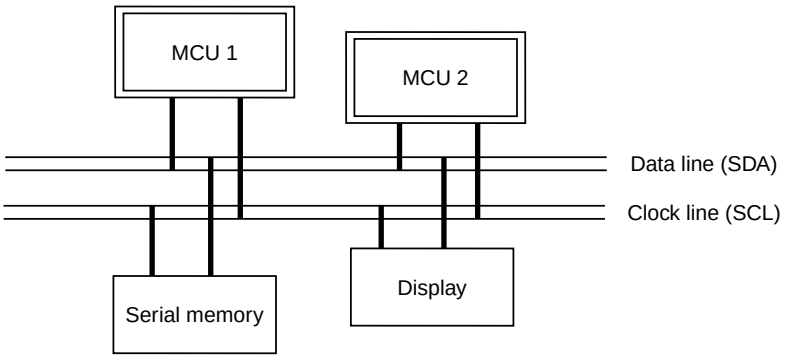


Figure 13-1 Example of I²C Bus Application

In an I²C bus system, devices are considered masters or slaves when performing data transfers. A master is a device that initiates a data transfer on the bus and generates the clock signals to permit that transfer. At that time, all devices addressed are considered slaves. Table 13-1 defines some I²C bus terminology.

Table 13-1 I²C Bus Terminology

Term	Description
Transmitter	A device that sends data to a bus
Receiver	A device that receives data from a bus
Master	A device that initiates a transfer, generates clock signals, and terminates a transfer
Slave	A device addressed by a master
Multimaster	More than one device capable of controlling the bus can be connected to the bus and more than one master can attempt to control the bus at the same time without corrupting the message. The system is not dependent on any single master.
Arbitration	A procedure to ensure that, if more than one master simultaneously tries to control the bus, only one is allowed to do so and the message is not corrupted. The device that loses arbitration becomes the slave of the device that wins.
Synchronization	A procedure to synchronize the clock signals of two or more devices

Figure 13-2 shows an example of an I²C bus configuration using two microcontrollers. Both I²C bus lines, SDA and SCL, are bidirectional and connected to a positive supply voltage via a pullup resistor. The open-drain output pins of the microcontrollers perform the wired-AND function on the bus. Software controls whether a microcontroller operates as a transmitter or receiver, and whether it is in master or slave mode.

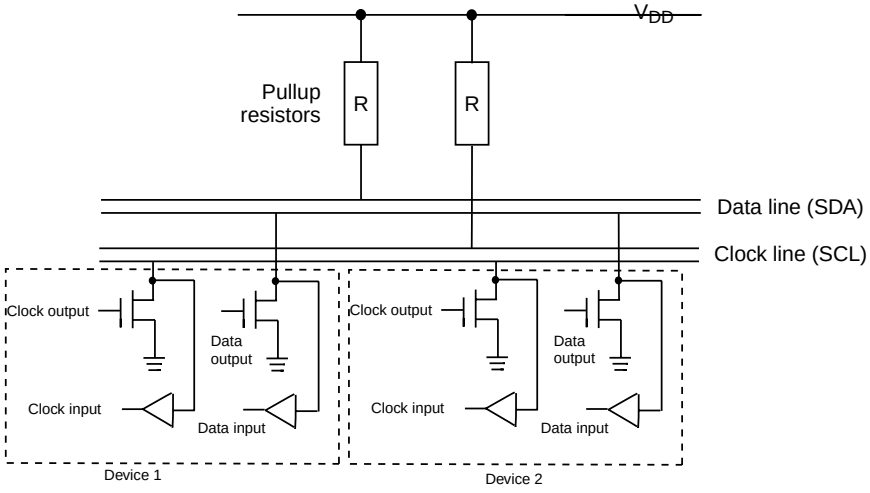


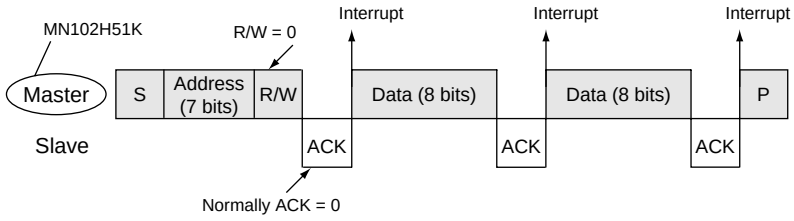
Figure 13-2 Connection of Two Microcontrollers to the I²C Bus

Table 13-2 describes the four operating modes for devices on the I²C bus.

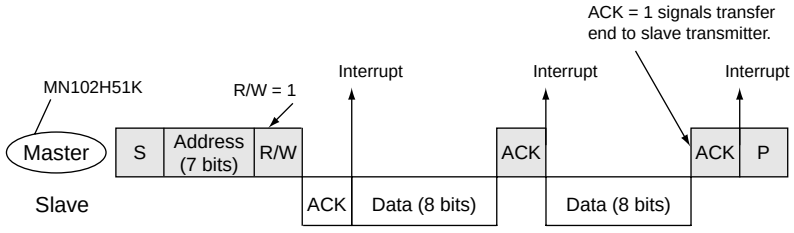
Table 13-2 Operating Modes for Devices on an I²C Bus

Operating Mode	Description
Master transmitter	The device that generates the serial transfer clock (SCL) signal and transmits serial data to a slave device in sync with SCL
Master receiver	The device that generates the SCL signal and receives serial data from a slave device in sync with SCL
Slave transmitter	A device that transmits data in sync with the SCL signal from the master
Slave receiver	A device that receives data in sync with the SCL signal from the master

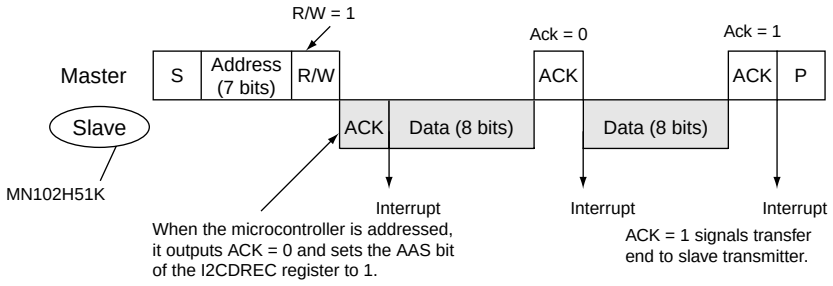
Figure 13-3 shows the MN101C46F operation sequence in each of these modes. In all modes, the I2C bus controller generates an interrupt after each data byte transfer. The software then loads the next data byte.



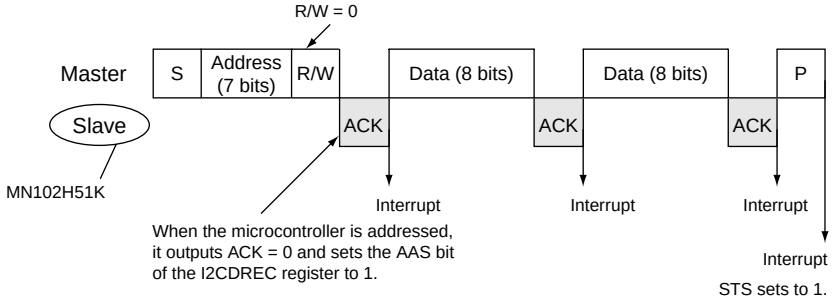
A. Master Transmitter



B. Master Receiver



C. Slave Transmitter



D. Slave Receiver

Figure 13-3 I2C Bus Interface Operation

13.2 Block Diagram

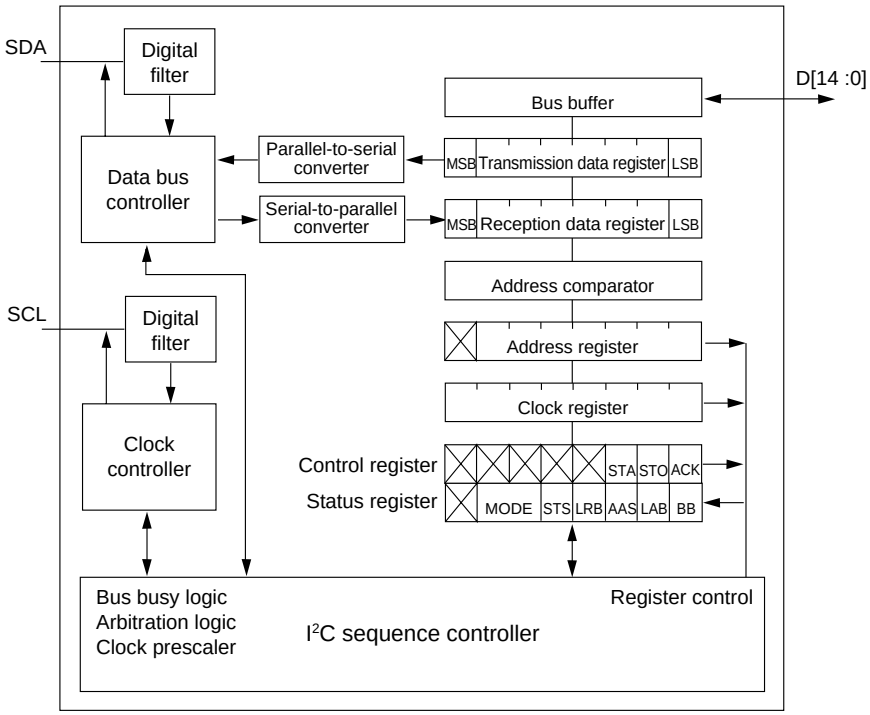


Figure 13-4 I²C Bus Controller Block Diagram

13.3 Functional Description

The I²C bus controller contains the registers shown in table 13-3. See the page number indicated for register and bit descriptions.

Table 13-3 Control Registers for Clamping Circuit

Register	Page	Address	Description
I2CDTRM	224	x'03E80'	I²C transmission data register
I2CDTRMH	224	x'03E81'	I²C transfer control register
I2CDREC	224	x'03E82'	I²C reception data register
I2CDRECH	224	x'03E83'	I²C mode register
I2CMYAD	225	x'03E84'	I²C self address register
I2CCLK	225	x'03E86'	I²C clock control register
I2CBRST	226	x'03E88'	I²C bus reset register
I2CBSTS	226	x'03E8A'	I²C bus status register

■ Arbitration and bus busy control

The I²C bus controller allows software control, but implements communication timing and bus arbitration completely in the hardware.

- **Arbitration:** Controlled by the software, but implemented completely in the hardware.

- **Bus busy:** Checked by the hardware. This eliminates the need for the software to check whether the bus is busy. The program can request a transfer to the I²C bus at any time.

Conversion of register settings to I²C protocol

The I²C bus controller converts the data in the I2CDTRM register to the I²C protocol.

Changes of transfer mode

A write to the I2CDTRMH register indicates the transfer mode (master transmitter/receiver or slave transmitter/receiver) for a new transfer. To minimize software control, the hardware generates an interrupt each time a transfer ends. During interrupt servicing, the SCL line stays low, then clears to high on a write to I2CDTRMH. (When the microcontroller is a slave transmitter and the transfer ends, SCL goes high on a read to the I2CDRECH register after an ACK = 1 (negative acknowledge) interrupt.)

Multimaster support

The hardware performs bus arbitration for a multimaster system. When it loses an arbitration, the hardware immediately stops the data transfer and generates an interrupt.

Address decoding

The I²C bus controller decodes the microcontroller's address, set in the I2CMYAD register, when the microcontroller is a slave device. It also decodes the general code address (0).

Forced bus reset

Through software control (by a write to the I2CBRST register), the I²C bus controller can force the SCL line to reset to low when a bus error occurs. This resets the entire I²C bus controller circuit, leaving the microcontroller in slave receiver mode. It does not change the contents of the I2CMYAD and I2CCLK registers.

Clock frequency adjustment

The I2CCLK register sets the serial clock frequency, allowing synchronization with low-speed devices. With a 3.58 MHz oscillator, the maximum setting is 100 kHz and the minimum setting is 10 kHz.

Bus state monitoring

With the I2CBSTS register, the I²C bus controller determines the logic levels of the SCL and SDA lines.

13.4 Setting up the I2C Bus Connection

Set the I2C connection in the I2CSEL0 and I2CSEL1 bits of the PCNT0 register (x'03F4A'). Since the SCL0, SDA0, SCL1, and SDA1 pins also serve as general-purpose port pins, and reset to the general-purpose function, you must set these bits every time the program uses the I2C function. You must also select the I2C function in the port mode registers. For I2C bus connection 0, set bits 0 and 1 of the P4MD register (x'03F2C'). For I2C bus connection 1, set bits 1 and 2 of the P0MD register (x'03F28').

Table 13-4 shows the register settings required to use either SDA0/SCL0 or SDA1/SCL1 alone; figure 13-5 shows the control circuit for this pin setup.

Table 13-4 Registers Settings for SDA0/SCL0 or SDA1/SCL1 Ports

Register	Bit	SDA0, SCL0 Only	SDA1, SCL1 Only
P0MD (x'03F28)	1	0 (selects P01)	1 (selects SDA1)
	2	0 (selects P02)	1 (selects SCL1)
P4MD (x'03F2C')	0	1 (selects SDA0)	0 (selects P60)
	1	1 (selects SCL0)	0 (selects P61)
PCNT0 (x'03F4A')	8	1 (enables SDA0, SCL0)	0 (disables SDA0, SCL0)
	9	0 (disables SDA1, SCL1)	1 (enables SDA1, SCL1)

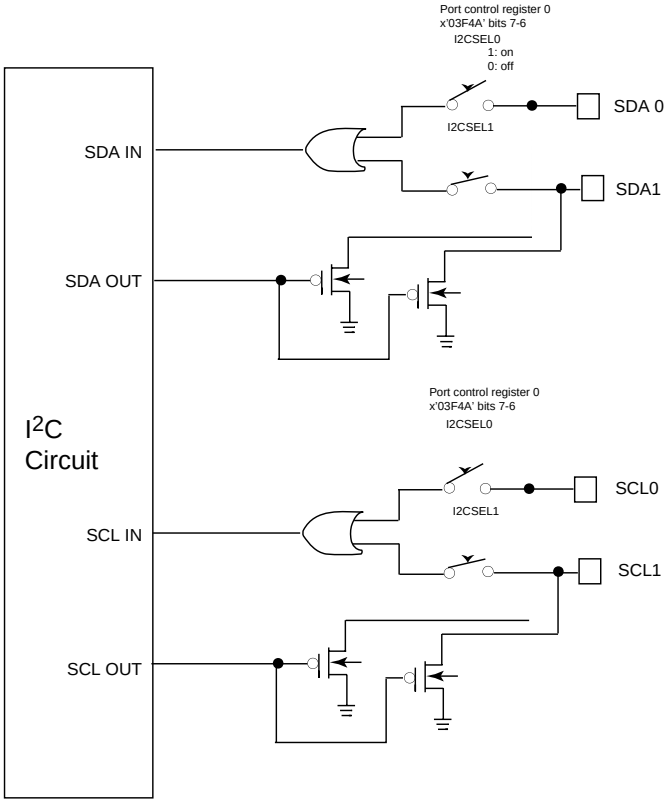


Figure 13-5 Pin Control Circuit for the I2C Bus Controller

13.5 SDA and SCL Waveform Characteristics

Figure 13-6 and table 13-5 provide the timing definitions and specifications for the MN101C46F I2C bus interface.

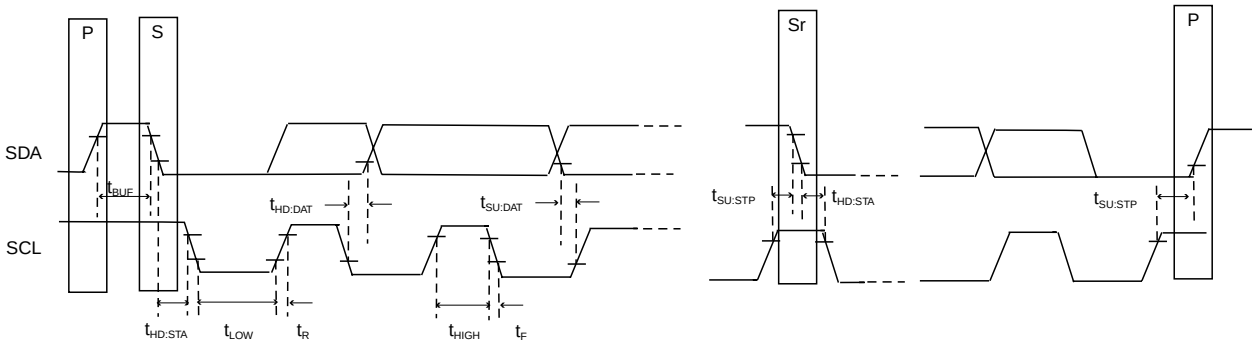


Figure 13-6 SDA and SCL Waveforms

Table 13-5 SDA and SCL Waveform Characteristics

Parameter	Symbol	Min	Max	Unit
SCL clock frequency	f_{SCL}	0	100	kHz
Bus free time between a stop and start condition	t_{BUF}	20	—	µs
Hold time (repeated) start condition	$t_{HD:STA}$	4.0	—	
Low period of the SCL clock	t_{LOW}	4.7	—	
High period of the SCL clock	t_{HIGH}	4.0	—	
Setup time for a repeated start condition	$t_{SU;STA:}$	300	—	
Data hold time	$t_{HD:DAT}$	250	—	ns
Data setup time	$t_{SU:DAT}$		—	
SDA and SCL rise time	t_R	—	1000	
SDA and SCL fall time	t_F	—	300	
Stop condition setup time	$t_{SU:STP}$	4.0		µs

13.6 I²C Interface Setup Examples

13.6.1 Setting up a Transition from Master Transmitter to Master Receiver

This example demonstrates how to set up a data transfer when changing from master transmitter to master receiver. Figure 13-7 shows an example waveform.

13.6.1.1 Pre-configuring

■ **To set up the I/O port:**
Set port control register 0 (PCNT0; x'03F4A') to x'40' (enabling the SDA0 and SCL0 pins) and set the port 4 output mode register (P4MD; x'03F2C') to x'06' (selecting the SDA0 and SCL0 functions).

■ **To enable I²C interrupts:**
Set the I²C interrupt control register I2CICR (x'03FF6') to x'02'.

■ **To set up the I²C registers:**

1. Set the I2CCLK register (x'03E86') to x'AD', selecting a clock frequency of 10 kHz.

2. Set the I2CDTRM register (x'03E80') to x'FD' and I2CDTRMH (x'03E81) to x'05'. This sets STA to 1, STP to 0, and ACK to 1. Bits 7 to 1 of the transmission data setting (x'FD') indicate the address (b'1111110') of the slave device from which the microcontroller will request the data, and bit 0 indicates the read/write setting (bit 0 = 1 = read).

13.6.1.2 Setting up the First Interrupt

When an ACK = 0 signal returns from the slave device, the I²C bus controller generates an interrupt. At this point, implement the following settings:

■ **To set up the interrupt:**
Set the I2CICR register (x'03FF6') to x'02'. This enables I²C interrupts and clears the previous interrupt request.

■ **To set up the I²C registers:**

1. Read the I2CDREC register (x'03E82') and the I2CDRECH register (x'03E83') to determine the I²C bus controller status.

2. Since the microcontroller will become a receiver on the next operation, set the I2CDTRMH register (x'03E81') to x'00'. This sets STA, STP, ACK, and the transmission data to 0s. With this setting, the microcontroller returns an ACK = 0 signal on the ninth clock.

13.6.1.3Setting up the Second Interrupt

When the microcontroller receives the data x'85' from the slave device, it returns an ACK = 0 signal and the I2C bus controller generates an interrupt. At this point, implement the following settings:

■ **To set up the interrupt:**

Set the I2CICR register (x'03FF6') to x'02'. This enables I2C interrupts and clears the previous interrupt request.

■ **To set up the I2C registers:**

1. Read the I2CDREC register (x'03E82') and the I2CDRECH register (x'03E83') to determine the I2C bus controller status.

2. Since the communication will end when the microcontroller receives the next data byte, set the I2CDTRMH register (x'03E81') to x'01'. This sets STA to 0, STP to 0, ACK to 1, and the transmission data to x'00'. With this setting, the microcontroller returns an ACK = 1 signal on the ninth clock.

13.6.1.4Setting up the Third Interrupt

When the microcontroller receives the data x'033' from the slave device, it returns an ACK = 1 signal and the I2C bus controller generates an interrupt. At this point, implement the following settings:

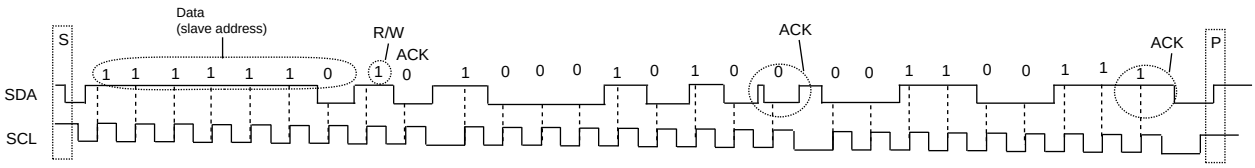
■ **To set up the interrupt:**

Set the I2CICR register (x'03FF6') to x'02'. This enables I2C interrupts and clears the previous interrupt request.

■ **To set up the I2C registers:**

1. Read the I2CDREC register (x'03E82') and the I2CDRECH register (x'03E83') to determine the I2C bus controller status.

2. Since the transfer has ended, set the I2CDTRMH register (x'03E81') to x'03'. This sets STA to 0, STP to 1, ACK to 1, and the transmission data to x'00'. With this setting, the microcontroller issues a stop condition and frees the bus.



Note: The circled areas are signals output from the MN101C46F.

Figure 13-7 Waveform for Master Transmitter Transitioning to Master Receiver

13.6.2 Setting up a Transition from Slave Receiver to Slave Transmitter

This example demonstrates how to set up a data transfer when changing from slave receiver to slave transmitter. Figure 13-8 shows an example waveform.

13.6.2.1 Pre-configuring

■ **To set up the I/O port:**
Set port control register 0 (PCNT0; x'03F4A') to x'80' (enabling the SDA1 and SCL1 pins) and set the port 0 output mode register (P0MD; x'FFFC') to x'06' (selecting the SDA1 and SCL1 functions).

■ **To enable I²C interrupts:**
Set the I2CICR register (x'03FF6') to x'02'.

■ **To set up the I²C registers:**

1. Set the I2CMYAD register (x'03E84') to x'24'. This sets the slave address of the microcontroller.

2. Set the I2CDTRMH register (x'03E81') to x'00'. This sets STA, STP, ACK, and the transmission data to 0s. With this setting, the microcontroller returns an ACK = 0 signal when an address match occurs. The master sends data (the slave address) to the slave microcontroller in sync with the master clock. When the R/W bit = 1, the microcontroller changes from a slave receiver to a slave transmitter.

13.6.2.2 Setting up the First Interrupt

Once the microcontroller becomes a slave transmitter, set up the transmission data.

■ **To set up the interrupt:**
Set the I2CICR register (x'03FF6') to x'02'. This enables I²C interrupts and clears the previous interrupt request.

■ **To set up the I²C registers:**

1. Read the I2CDREC register (x'03E82') and the I2CDRECH register (x'03E83') to determine the I²C bus controller status. AAS should be 1.

2. Set the I2CDTRM register (x'03E80') to x'55' and the I2CDTRMH register (x'03E81') to x'01'. This sets STA to 0, STP to 0, ACK to 1, and the transmission data to x'55'. The microcontroller does not need to issue an ACK signal in this transfer, so the ACK bit should be 1.

3. Begin transmitting data in sync with the clock from the master.

13.6.2.3 Setting up the Second Interrupt

The master sends an ACK = 0 signal, so the microcontroller must send the next data byte. Set up the transmission data as follows:

■ **To set up the interrupt:**

Set the I2CICR register (x'03FF6') to x'02'. This enables I²C interrupts and clears the previous interrupt request.

■ **To set up the I²C registers:**

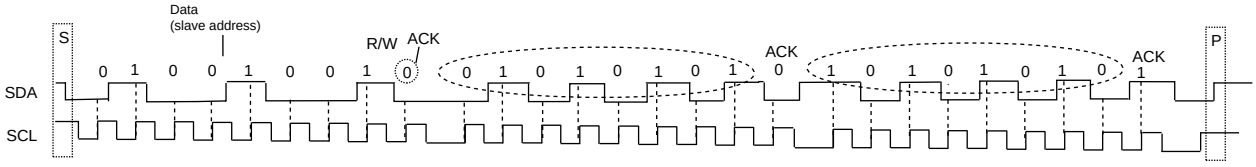
1. Read the I2CDREC register (x'03E82') and the I2CDRECH register (x'03E83') to determine the I²C bus controller status. The previous read from I2CDREC cleared the AAS, so AAS should be 0.

2. Set the I2CDTRM register (x'03E80') to x'AA' and the I2CDTRMH register (x'03E81') to x'01'. This sets STA to 0, STP to 0, ACK to 1, and the transmission data to x'AA'. The microcontroller does not need to issue an ACK signal in this transfer, so the ACK bit should be 1.

3. Begin transmitting data in sync with the clock from the master.

13.6.2.4 Setting up the Third Interrupt

The master send an ACK = 1 signal, then issues a stop condition, ending the communication.



Note: The circled areas are signals output from the MN101C46F.

Figure 13-8 Waveform for Slave Receiver Transitioning to Slave Transmitter

13.7 I²C Bus Interface Registers

I2CDTRMH: I²C Transmission Data Register High **x'03E81'**
I2CDTRM: I²C Transmission Data Register **x'03E80'**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	—	—	—	—	—	STA	STP	ACK	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SCL is held low during interrupt servicing, and is cleared high by a write to I2CDTRM.

STA: I²C start control

STP: I²C stop control

Writing to the STA and STP bits allows you to change the state of the transmission or reception operation. Table 13-6 shows the settings for different start and stop conditions.

Table 13-6 STA and STP Settings

STA	STP	Mode	Function	Description
0	0	All	NOP	No state change
1	1	All	NOP	No state change
1	0	Slave receiver	Start	Change to mode indicated by R/W bit. R/W = 0: Change to master transmitter R/W = 1: Change to master receiver
		Master transmitter	Repeat start	
0	1	Slave receiver	Stop read	Change to slave receiver after stop condition.
		Master transmitter	Stop write	

ACK: Acknowledge signal output control

The acknowledge signal is output after every byte transfer, on the ninth clock pulse. ACK is normally 1 and transitions to 0 to output an acknowledge (for instance if the master or slave receiver has received a data byte)

DT[7:0]: Data to be transmitted

The parallel data in this field is converted to serial data for transmission to the I²C bus. It is shifted out MSB first to the interface.

I2CDRECH: I²C Reception Data Register High **x'03E83'**
I2CDREC: I²C Reception Data Register **x'03E82'**

Bit:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	—	MODE 1	MODE 0	STS	LRB	AAS	LAB	BB	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0
Reset:	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

The I2CDRECH register contains the status bits for monitoring the device. The I2CDREC register contains the reception data. I2CDRECH and I2CDREC are read-only registers.

MODE[1:0]: I²C device mode

This field indicates which I²C mode the microcontroller is in. MODE1 indicates slave or master, and MODE0 indicates receiver or transmitter. If the microcontroller loses an arbitration or if a stop condition occurs, the hardware clears MODE[1:0] to b'00'.

00: Slave receiver 10: Master receiver
01: Slave transmitter 11: Master transmitter

STS: Stop condition at slave receiver

Set to 1 when a stop condition is detected while the microcontroller is in slave receiver mode.

LRB: Last received bit.

Stores the last serial data bit received. LRB normally indicates the ACK cycle data.

AAS: Addressed as slave

Set to 1 when the slave address on the bus matches the contents of the address register or matches the general address (x'00'). AAS resets after a read from the I2CDRECH register.

LAB: Lost arbitration bit

Set to 1 when the microcontroller loses a bus arbitration. LAB resets when I2CDTRMH indicates a start condition (STA = 1).

BB: Bus busy bit

A start condition on the bus sets this flag to 0, and a stop condition resets it to 1. The microcontroller considers the bus to be busy as long as BB = 0.

D[7:0]: Received data

The serial data received from the I²C bus is shifted into this field MSB first.

I2CMYAD: I²C Self Address Register **x'03E84'**

Bit:	7	6	5	4	3	2	1	0
	—	A6	A5	A4	A3	A2	A1	A0
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

A[6:0]: Microcontroller address

This register is formed from a 7-bit field address latch. It holds the microcontroller's own address, used for a compare when the microcontroller is addressed as a slave. When a match occurs, AAS is set to 1.

I2CCLK: I²C Clock Control Register **x'03E86'**

Bit:	7	6	5	4	3	2	1	0
	C7	C6	C5	C4	C3	C2	C1	C0
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

C[7:0]: Output clock frequency select

This 8-bit field determines the SCL output. With a 3.58-MHz system clock, calculate the frequency as follows:

$$f_{SCL} = \frac{3.58\ MHz}{2 \times (Register\ setting + 6)}$$

To conform to the specification, the clock signal must be between 0 and 100 kHz. To satisfy this requirement, always set I2CCLK to x'0C' or higher.

In this case, the following C[7:0] settings apply:

x'0C':	99.4 kHz	x'1E':	49.7 kHz
x'0E':	89.5 kHz	x'27':	39.8 kHz
x'10':	81.4 kHz	x'036':	29.8 kHz
x'14':	68.8 kHz	x'54':	19.9 kHz
x'18':	59.7 kHz	x'AD':	10.0 kHz

I2CBRST: I2C Bus Reset Register **x'03E88'**

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	BRST
Reset:	0	0	0	0	0	0	0	1
R/W:	R	R	R	R	R	R	R	R/W

BRST: Bus reset

When a serious bus error occurs, this bit can be set to 0, forcing the clock line low and resetting the I2C bus. This function works in all I2C modes. After a forced reset, the microcontroller is in slave receiver mode. This reset does not change the contents of the I2CMYAD and I2CCLK registers.

0: Force bus to reset

1: Steady state

I2CBSTS: I2C Bus Status Register **x'03E8A'**

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	SDAS	SCLS
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

I2CBSTS is a two-bit, read-only register that monitors the status of the I2C bus.

SDAS: SDA data line status

This bit monitors the state of the I2C data line, SDA.

SCLS: SCL clock line status

This bit monitors the state of the I2C clock line, SCL.

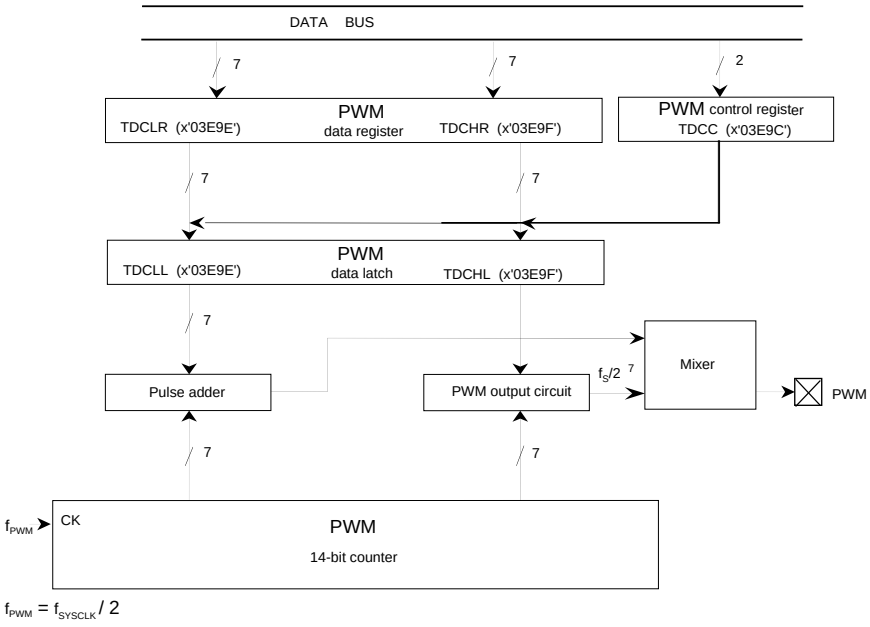
14 Pulse Width Modulator

14.1 14-Bit Pulse Width Modulator

The MN101C46F has a 14-bit pulse width modulator (PWM). The PWM has a resolution of 14 bits, a minimum pulse width of $2/f_{SYSCLK}$, and a cycle of $2^7/f_{PWM}$.

14.1.1 14-Bit PWM Description

The following description assumes an internal oscillation frequency f_{SYSCLK} of 3.58 MHz. Figure 14-1 shows a block diagram of the 14-bit PWM. Transfer the 14-bit waveform data written in 14-bit PWM data registers TDCHR and TDCLR to 14-bit PWM data latches TDCHL and TDCLL respectively by setting bit 11 in 14-bit PWM control register TDCC. Since the clock divided by PWM is used as the clock for remote control, PWM must be engaged when using the remote signal receiver.



Notes: 1. Minimum pulse width: $2/f_{SYSCLK}$ = 0.56 µs

2. Repeat cycle: $2^7/f_{PWM}$ = 71.5 µs

Figure 14-1 14-Bit PWM Block Diagram

14.1.2 14-Bit PWM Output Waveforms

Figure 14-2 shows the 14-bit output waveform. (This example assumes f_{SYSCLK} = 3.58 MHz.) The PWM output modulated by the 14-bit PWM's high-order 7 bits (TDCHR data) whose cycle is t_{SUB} (71.5 µs) is overlapped with an added pulse whose minimum pulse is at the 128 types of positions set by the 14-bit PWM low-order bits (TDCLR data) and then output.

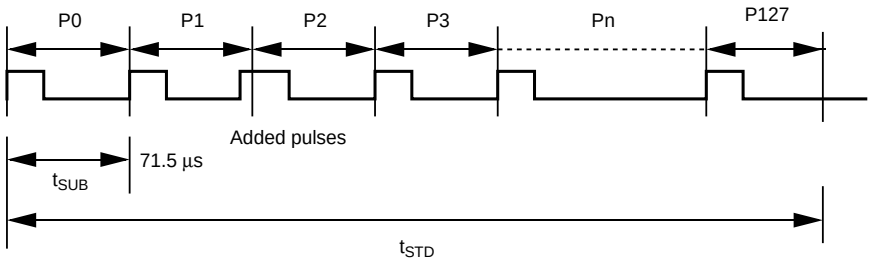


Figure 14-2 14-Bit PWM Output Waveform

Figure 14-4 shows the relationship of the t_{SUB} (71.5 µs) cycle to the PWM output of the high-order 7-bit TDCHR data. Table 14-1 shows the relationship of the position of the added pulse overlapped by the TDCLR data (low-order 7 bits of the 14-bit PWM data). Figure 14-4 shows the output waveform diagram..

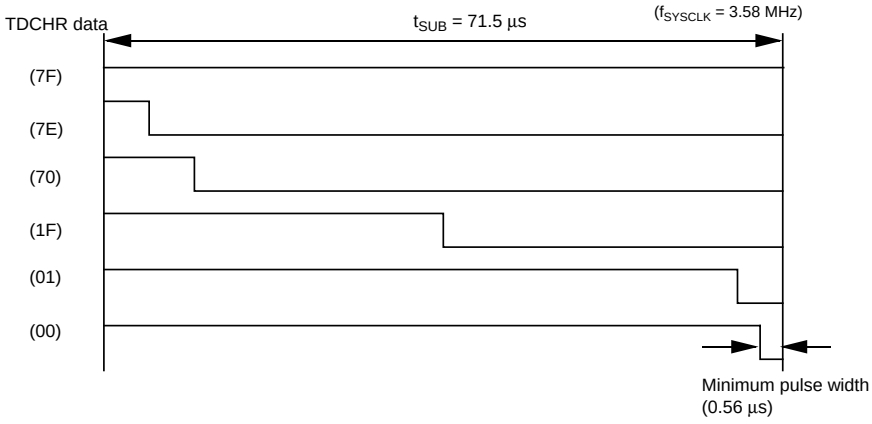


Figure 14-3 t_{SUB} PWM Output Waveform

Table 14-1 Added Pulse Overlapping Position

TDCLR Data	Pn on which Added Pulse Is overlapped (n = 0 to 127)
7F	127
7E	63, 127
7D	31, 95, 127
7C	31, 63, 95, 127
7B	15, 47, 79, 111, 127
7A	15, 47, 63, 79, 111, 127
—	—
01	0 to 127 (except 63)
00	0 to 127

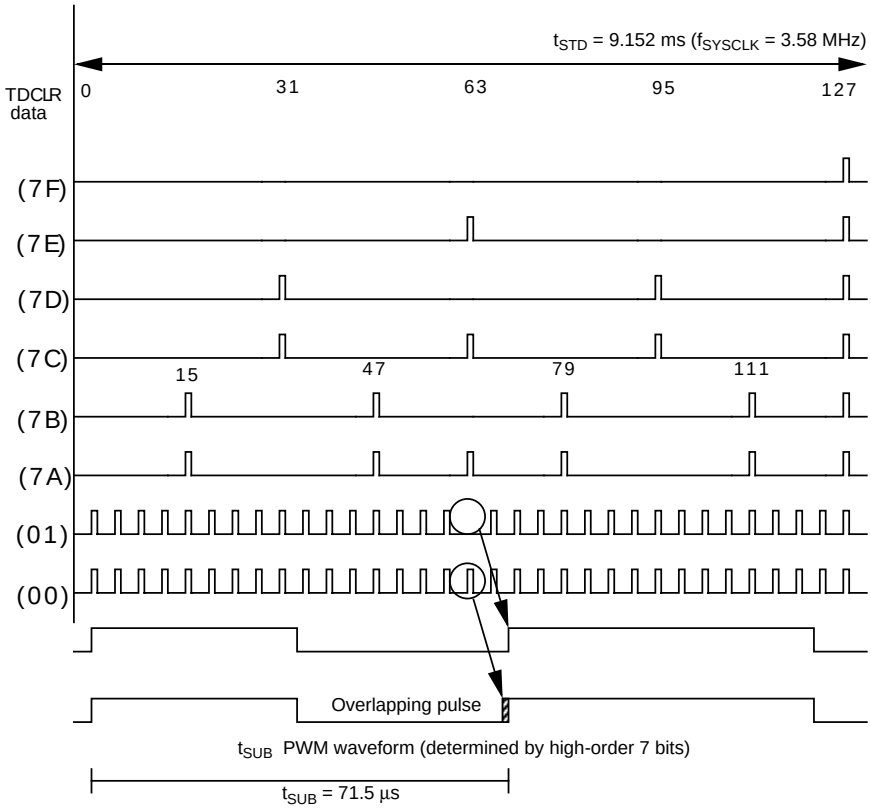


Figure 14-4 Added Pulse Waveform

14.1.3 Data Transfers from Registers to Latches

When TDCC is set to 11, 14-bit data is transferred from a 14-bit PWM data register to a 14-bit PWM data latch during the repeat cycle of the high-order 7 bits of the next PWM. This is to prevent the PWM waveform from being disrupted by changes in the data that may occur partway through the PWM waveform. If TDCC is set to 00, the 14-bit PWM output waveform does not change even when the 14-bit PWM data register value is rewritten.

14.2 8-Bit Pulse Width Modulators

The MN101C46F has six 8-bit PWMs. These PWMs have resolutions of eight bits, minimum pulse widths of $2/f_{SYSCLK}$, and cycles of $2^8/f_{PWM}$.

14.2.1 8-Bit PWM Description

The MN101C46F has six 8-bit PWM blocks. The 8-bit registers (PWMn) are used to write pulse width modulation data concerning the PWM blocks. The 8-bit PWM output waveform or repeat cycle is 143 µs and the minimum pulse width is 0.56 µs.

Figure 14-5 shows a block diagram of the 8-bit PWMs

..

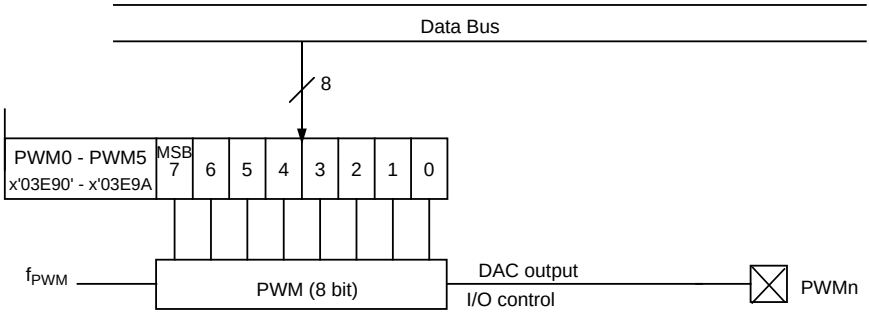


Figure 14-5 Block Diagram of 8-Bit PWM

Notes:
1. $f_{PWM} = f_{SYSCLK}/2$
2. Output pulse cycle=$2^8/f_{PWM}$ = 143 µs (when f_{SYSCLK} = 3.58 MHz)
3. Minimum pulse width = $1/f_{PWM}$ = 0.56 µs
4. t_{LOW} = (PWMn+1) × 0.56 µs

14.2.2 8-Bit PWM Output Waveform

Figure 14-3 shows the 8-bit PWM waveform. It is assumed that f_{SYSCLK} = 3.58 MHz. The 8-bit PWM output pulse width changes relative to settings of data registers in units of $2^8/f_{PWM}$ cycles. When it changes between the data settings 00 and 01, however, it changes at double $1/f_{PWM}$.

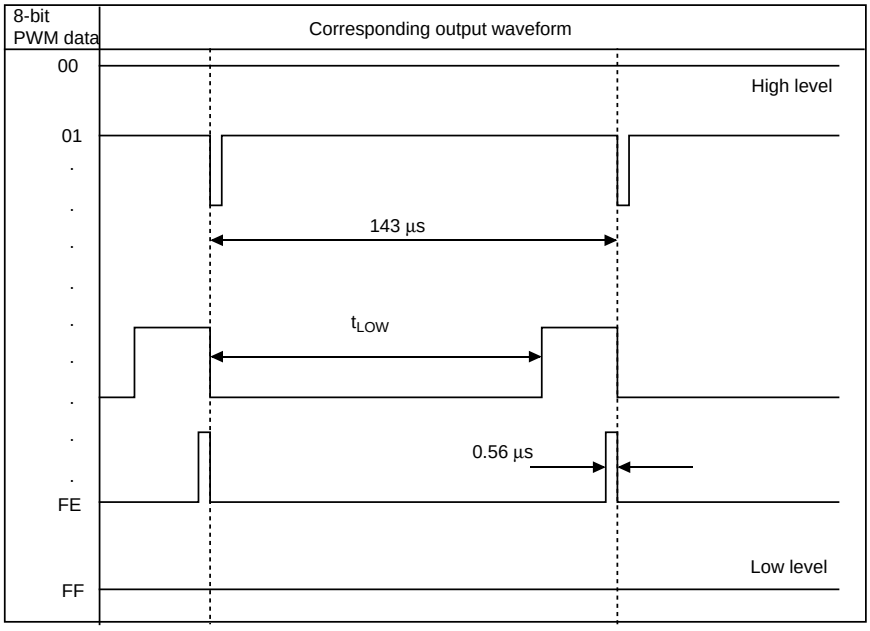


Figure 14-6 8-Bit PWM Output Waveform

14.3 PWM Registers

Register	Address	R/W	Description
TDCC	x'03E9C'	R/W	14-bit PWM control register
TDCHR	x'03E9F'	W	14-bit PWM data register high
TDCHL	x'03E9E'	W	14-bit PWM data register low
TDCHL	x'03E9F'	R	14-bit PWM data latch high
TDCHR	x'03E9E'	R	14-bit PWM data latch low
PWM0	x'03E90'	R/W	8-bit PWM data register 0
PWM1	x'03E92'	R/W	8-bit PWM data register 1
PWM2	x'03E94'	R/W	8-bit PWM data register 2
PWM3	x'03E96'	R/W	8-bit PWM data register 3
PWM4	x'03E98'	R/W	8-bit PWM data register 4
PWM5	x'03E9A'	R/W	8-bit PWM data register 5

14.3.1 14-Bit PWM Control Registers

Data must be set in the following three types of registers for 14-bit PWM control.

TDCC: 14-bit PWM Control Register **x'03E9C'**

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	Reserved	TDCC1	TDCC0
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R/W	R/W	R/W

TDCC is a 2-bit register that controls 14-bit PWM operation.

Reserved: Always set to 0.

TDCC[1:0]: Data latch transfer bits

00: Do not transfer
01: TDCLR → TDCLL
10: TDCHR → TDCHL
11: TDCHR → TDCHL and TDCLR → TDCLL

TDCHR: 14-bit PWM Data Register High **x'03E9F'**

Bit:	7	6	5	4	3	2	1	0
	Reserved	TDCHR 6	TDCHR 5	TDCHR 4	TDCHR 3	TDCHR 2	TDCHR 1	TDCHR 0
Reset:	0	0	0	0	0	0	0	0
R/W:	W	W	W	W	W	W	W	W

TDCHRn: 14-bit PWM high-order data. Indicates 14-bit PWM output waveform modulation data. (n=6–0)

TDCLR: 14-bit PWM Data Register Low **x'03E9E'**

Bit:	7	6	5	4	3	2	1	0
	Reserved	TDCLR6	TDCLR5	TDCLR4	TDCLR3	TDCLR2	TDCLR1	TDCLR0
Reset:	0	0	0	0	0	0	0	0
R/W:	W	W	W	W	W	W	W	W

TDCLRn: 14-bit PWM low-order data. Indicates overlapping position of pulse added to 14-bit PWM output waveform. (n=6–0)

TDCHL: 14-bit PWM Data Latch High **x'03E9F'**

Bit:	7	6	5	4	3	2	1	0
	Reserved	TDCHL 6	TDCHL 5	TDCHL 4	TDCHL 3	TDCHL 2	TDCHL 1	TDCHL 0
Reset:	X	X	X	X	X	X	X	X
R/W:	R	R	R	R	R	R	R	R

TDCHLn: 14-bit PWM high-order data latch. Latches TDCHR register data. (n=6–0)

TDCLL: PWM Data Latch Low **x'03E9E'**

Bit:	7	6	5	4	3	2	1	0
	Reserved	TDCLL6	TDCLL5	TDCLL4	TDCLL3	TDCLL2	TDCLL1	TDCLL0
Reset:	X	X	X	X	X	X	X	X
R/W:	R	R	R	R	R	R	R	R

TDCLLn: 14-bit PWM low-order data latch. Latches TDCLR register data. (n=6–0)

14.3.2 8-Bit PWM Control Registers

PWM0-PWM5 are 8-bit registers. They are used for writing 8-bit PWM output waveform modulation data within one repeat cycle. They are set to 0 upon reset and 8-bit PWM output is set to H.

PWM0: 8-bit PWM Data Register 0 **x'03E90'**

Bit:	7	6	5	4	3	2	1	0
	PWM0 DT7	PWM0 DT6	PWM0 DT5	PWM0 DT4	PWM0 DT3	PWM0 DT2	PWM0 DT1	PWM0 DT0
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

PWM1: 8-bit PWM Data Register 1 **x'03E92'**

Bit:	7	6	5	4	3	2	1	0
	PWM1 DT7	PWM1 DT6	PWM1 DT5	PWM1 DT4	PWM1 DT3	PWM1 DT2	PWM1 DT1	PWM1 DT0
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

PWM2: 8-bit PWM Data Register 2 **x'03E94'**

Bit:	7	6	5	4	3	2	1	0
	PWM2 DT7	PWM2 DT6	PWM2 DT5	PWM2 DT4	PWM2 DT3	PWM2 DT2	PWM2 DT1	PWM2 DT0
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

PWM3: 8-bit PWM Data Register 3 **x'03E96'**

Bit:	7	6	5	4	3	2	1	0
	PWM3 DT7	PWM3 DT6	PWM3 DT5	PWM3 DT4	PWM3 DT3	PWM3 DT2	PWM3 DT1	PWM3 DT0
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

PWM4: 8-bit PWM Data Register 4 **x'03E98'**

Bit:	7	6	5	4	3	2	1	0
	PWM4 DT7	PWM4 DT6	PWM4 DT5	PWM4 DT4	PWM4 DT3	PWM4 DT2	PWM4 DT1	PWM4 DT0
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

PWM5: 8-bit PWM Data Register 5 **x'03E9A'**

Bit:	7	6	5	4	3	2	1	0
	PWM5 DT7	PWM5 DT6	PWM5 DT5	PWM5 DT4	PWM5 DT3	PWM5 DT2	PWM5 DT1	PWM5 DT0
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Appendix A Register Map

Table A-1 Register Map: x'03D00'-x'03DFF'

12 MSBs	4 LSBs																Description
	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	
x'03D00																	
x'03D10																	
x'03D20																	
x'03D30																	
x'03D40																	
x'03D50																	
x'03D60																	
x'03D70					SLCNT2	SLCNT1											VBI (1)
x'03D80																	
x'03D90																	
x'03DA0																	
x'03DB0																	
x'03DC0																	
x'03DD0																	
x'03DE0																	
x'03DF0					SLCNT2W	SLCNT1W											VBI (2)

Table A-2 Register Map: x'03E00'-x'03EFF'

12 MSBs	4 LSBs																Description
	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	
x'03E00	CRI4FQW	CRI3FQW	CRI2FQW	CRI1FQW	CAPDATAH	CAPDATA	ACQ1H	ACQ1	VBIIRQ	HNUM	SLSF	SLHD	MAX	MIN	SLCNT	FC	VBI 1 registers
x'03E10	FCPNUMH	FCPNUM	STAPH	STAP	DATAEH	DATAE	DATASH	DATAS	CRI2EH	CRI2E	CRI2SH	CRI2S	CRI1EH	CRI1E	CRI1SH	CRI1S	
x'03E20	CRI4FQWW	CRI3FQWW	CRI2FQWW	CRI1FQWW	CAPDATAWH	CAPDATAW	ACQ1WH	ACQ1W	VBIIRQW	HNUMW	SLSFW	SLHDW	MAXW	MINW	SLCNTW	FCW	VBI 2 registers
x'03E30	FCPNUMWH	FCPNUMW	STAPWH	STAPW	DATAEWH	DATAEW	DATASWH	DATASW	CRI2EWH	CRI2EW	CRI2SWH	CRI2SW	CRI1EWH	CRI1EW	CRI1SWH	CRI1SW	
x'03E40	HSEP1H	HSEP1	CLAMPH	CLAMP	SPLVH	SPLV	BPLV	SYNCMIN	BPPSTH	BPPST	SCMINGH	SCMING	FQSELH	FQSEL	NFSELH	NFSEL	Sync separator 1 registers
x'03E50			CLPCND1H	CLPCND1		HVCOND	VCNTH	VCNT	HDISTWH	HDISTW	HLOCKLVH	HLOCKLV	FIELDH	FIELD	HSEP2H	HSEP2	
x'03E60	HSEP1WH	HSEP1W	CLAMPWH	CLAMPW	SPLVWH	SPLVW	BPLVW	SYNCMINW	BPPSTWH	BPPSTW	SCMINGWH	SCMINGW	FQSELWH	FQSELW	NFSELWH	NFSELW	Sync separator 2 registers
x'03E70			CLPCND1WH	CLPCND1W		HVCONDW	VCNTWH	VCNTW	HDISTWWH	HDISTWW	HLOCKLVWH	HLOCKLVW	FIELDWH	FIELDW	HSEP2WH	HSEP2W	
x'03E80						I2CBSTS		I2CBRST		I2CCLK		I2CMYAD	I2CDRECH	I2CDREC	I2CDTRMH	I2CDTRM	I^2C interface registers
x'03E90	TDCHR	TDCLR		TDCC		PWM5		PWM4		PWM3		PWM2		PWM1		PWM0	PWM registers

Table A-2 Register Map: x'03E00'-x'03EFF' (Continued)

x’03EA0				RMLD		RMTR		RMSR		RMCS		RMTC		RMIR		RMIS	Remote signal receiver registers
x’03EB0	EVODH	EVOD	HCOUNTH	HCOUNT		OSD3		OSD2		OSD1		RAMEND		GROMEND		CROMEND	OSD control registers
x’03EC0	IAPH	IAP	IVPH	IVP	IHPH	IHP		SHTC	HSHT1H	HSHT1	HSHT0H	HSHT0	VSHT1H	VSHT1	VSHT0H	VSHT0	
x’03ED0										WBSHD		BBSHD		FRAME		COLB	
x’03EE0	PLT17	PLT16	PLT15	PLT14	PLT13	PLT12	PLT11	PLT10	PLT07	PLT06	PLT05	PLT04	PLT03	PLT02	PLT01	PLT00	
x’03EF0	PLT37	PLT36	PLT35	PLT34	PLT33	PLT32	PLT31	PLT30	PLT27	PLT26	PLT25	PLT24	PLT23	PLT22	PLT21	PLT20	
x’03F00			OSCMD							ACTMD			DLYCTR	WDCTR	MEMCTR	CPUM	CPU mode and memory control
x’03F10												P4OUT	P3OUT	P2OUT	P1OUT	P0OUT	I/O port output
x’03F20				P4MD	P3MD	P2MD	P1MD	P0MD				P4IN	P3IN	P2IN	P1IN	P0IN	I/O port input
x’03F30	ROMCENH	ROMCEN										P4DIR	P3DIR	P2DIR	P1DIR	P0DIR	I/O port I/O mode control
x’03F40		PCNT2				PCNT0						P4PLU	P3PLU	P2PLU	P1PLU	P0PLU	I/O port pullup resistor control
x’03F50	CK3MD	CK2MD	TM3MD	TM2MD	TM3OC	TM2OC	TM3BC	TM2BC									Timer control
x’03F60	PSCMD									CK4MD		TM4MD		TM4OC		TM4BC	
x’03F70	AMCHIM7	AMCHIL7	AMCHIM6	AMCHIL6	AMCHIM5	AMCHIL5	AMCHIM4	AMCHIL4	AMCHIM3	AMCHIL3	AMCHIM2	AMCHIL2	AMCHIM1	AMCHIL1	AMCHIM0	AMCHIL0	ROM correction control 1 (correction address)
x’03F80	AMCHIMF	AMCHILF	AMCHIME	AMCHILE	AMCHIMD	AMCHILD	AMCHIMC	AMCHILC	AMCHIMB	AMCHILB	AMCHIMA	AMCHILA	AMCHIM9	AMCHIL9	AMCHIM8	AMCHIL8	
x’03F90	AMCHIHF	AMCHIHE	AMCHIHD	AMCHIHC	AMCHIHB	AMCHIHA	AMCHIH9	AMCHIH8	AMCHIH7	AMCHIH6	AMCHIH5	AMCHIH4	AMCHIH3	AMCHIH2	AMCHIH1	AMCHIH0	
x’03FA0																	Serial interface control
x’03FB0												AN BUF1		AN CTR2	AN CTR1	AN CTR0	Analog interface control
x’03FC0																	
x’03FD0	CH DATF	CH DATE	CH DATD	CH DATC	CH DATB	CH DATA	CH DAT9	CH DAT8	CH DAT7	CH DAT6	CH DAT5	CH DAT4	CH DAT3	CH DAT2	CH DAT1	CH DAT0	ROM correction control 2 (correction data)
x’03FE0			TM4ICR	TM3ICR	TM2ICR				IRQ5ICR	IRQ4ICR	IRQ3ICR	IRQ2ICR	IRQ1ICR	IRQ0ICR	NMICR		Interrupt control
x’03FF0					RMCICR	ADICR		VBIV1ICR	VBIV0ICR	I2CICR	OSDICR	VBI1ICR	VBI0ICR				

Appendix B MN101CF46F Flash EEPROM Version

B.1 Description

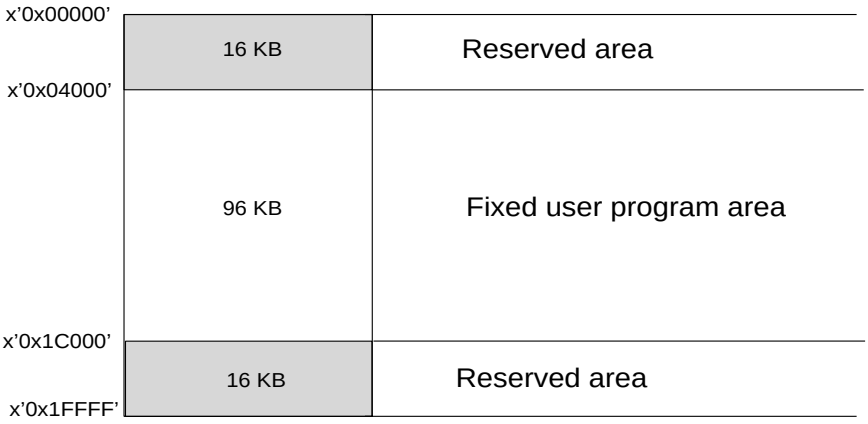
The MN101CF46F is an electrically programmable, 96-kilobyte flash ROM versions of the MN101CF46F. It is programmed in PROM writer mode, which uses a dedicated writer.

The 96-kilobyte flash memory is divided into two main areas:

- Fixed user program area (96 kilobytes: x'0x04000 to x'0x1BFFF)
  This area stores the user program. It is overwritten in PROM writer mode.

- User program area (32 kilobytes: x'0x00000 to x'0x03FFF and x'0x1C000 to x;0x1FFFF)
  This area cannot be used. (When using the writer, fill this area with FF.)

Normal operation is guaranteed with up to ten programmings.



Note: The shaded region is write-protected in this mode.

Figure B-1 Memory Map for Internal Flash EEPROM

B.2 Benefits

Because you can maintain and upgrade the program in the MN101CF46F up to and immediately following product release, this version of the device shortens time-to-market by as much as one month. This device is ideal for applications in quickly changing markets, since it allows you to revise the microcontroller program in an existing product.

B.3 Using the PROM Writer Mode

In this mode, the MN101CF46F allows a PROM writer to program the internal flash memory as if it was a standalone memory chip. The microcontroller is inserted into a dedicated adaptor socket, which connects to DATA-I/O's LabSite PROM writer. When the microcontroller connects to the adaptor socket, it automatically enters PROM writer mode. The adaptor socket ties the microcontroller pin states to PROM writer mode, and programming occurs without any reference to the microcontroller pin states.

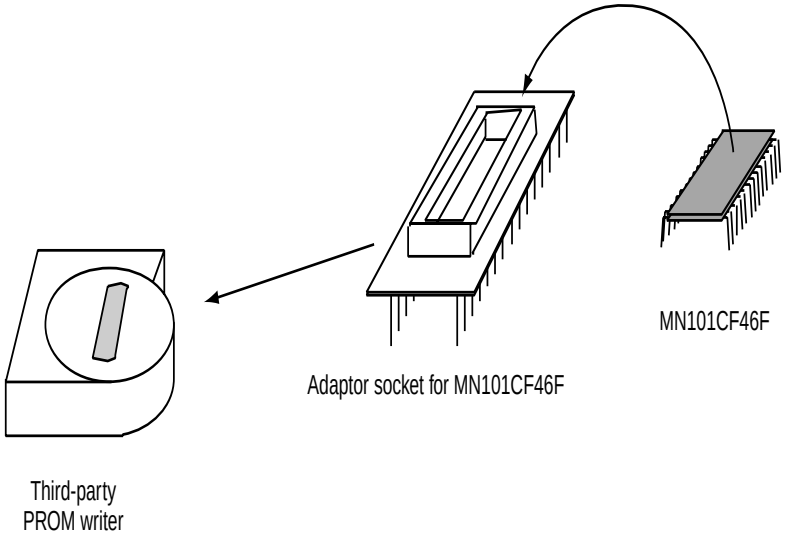


Figure B-2 PROM Writer Hardware Setup

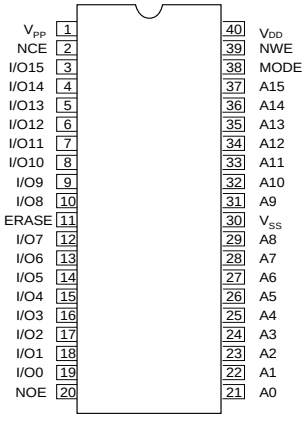


Figure B-3 Pin Configuration for Socket Adaptor

B.4 Reprogramming Flow

Figure B-4 shows the flow for reprogramming (erasing and programming) the flash memory

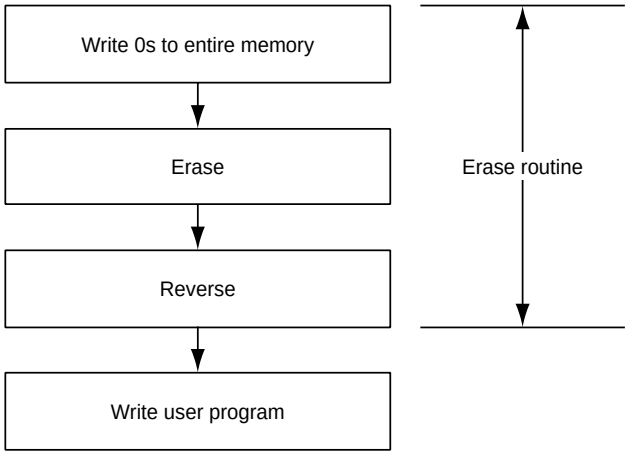


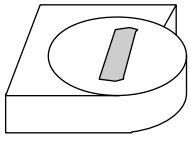
Figure B-4 EEPROM Programming Flow

As the figure shows, the write occurs after the memory is completely erased. The erase routine consists of three steps, first writing all zeros to the entire memory space, next erasing the memory, and finally reversing.

B.5 Programming Times

Table B-1 shows the time required for PROM reprogramming (erasing and programming).

Table B-1 Programming Times for PROM Writers

Writer	Programming Time (User Program Only)	Reprogramming Time
DATA-I/O LabSite DIP48-1  (provisional)	TBA	TBA

Appendix C MN101C Series Instruction Set

MN101C SERIES INSTRUCTION SET

Group	Mnemonic	Operation	Flag VF	NF	CF	ZF	Code Size	Cycle	Re-peat	Machine Code Ext.	1	2	3	4	5	6	7	8	9	10	11	Notes	Page
Data Move Instructions																							
MOV	MOV Dn,Dm	Dn→Dm	--	--	--	--	2	1			1010	DnDm											25
	MOV imm8,Dm	imm8→Dm	--	--	--	--	4	2			1010	DmDm	<#8.	...>									25
	MOV Dn,PSW	Dn→PSW	●	●	●	●	3	3		0010	1001	01Dn											26
	MOV PSW,Dm	PSW→Dm	--	--	--	--	3	2		0010	0001	01Dm											26
	MOV (An),Dm	mem8(An)→Dm	--	--	--	--	2	2			0100	1ADm											27
	MOV (d8,An),Dm	mem8(d8+An)→Dm	--	--	--	--	4	2			0110	1ADm	<d8.	...>								*1	27
	MOV (d16,An),Dm	mem8(d16+An)→Dm	--	--	--	--	7	4		0010	0110	1ADm	<d16	....	....	...>							28
	MOV (d4,SP),Dm	mem8(d4+SP)→Dm	--	--	--	--	3	2			0110	01Dm	<d4>									*2	28
	MOV (d8,SP),Dm	mem8(d8+SP)→Dm	--	--	--	--	5	3		0010	0110	01Dm	<d8.	...>								*3	29
	MOV (d16,SP),Dm	mem8(d16+SP)→Dm	--	--	--	--	7	4		0010	0110	00Dm	<d16	....	....	...>							29
	MOV (io8),Dm	mem8(IOTOP+io8)→Dm	--	--	--	--	4	2			0110	00Dm	<io8	...>									30
	MOV (abs8),Dm	mem8(abs8)→Dm	--	--	--	--	4	2			0100	01Dm	<abs	8..>									30
	MOV (abs12),Dm	mem8(abs12)→Dm	--	--	--	--	5	2			0100	00Dm	<abs	12..	...>								31
	MOV (abs16),Dm	mem8(abs16)→Dm	--	--	--	--	7	4		0010	1100	00Dm	<abs	16..	....	...>							31
	MOV Dn,(Am)	Dn→mem8(Am)	--	--	--	--	2	2			0101	1aDn											32
	MOV Dn,(d8,Am)	Dn→mem8(d8+Am)	--	--	--	--	4	2			0111	1aDn	<d8.	...>								*1	32
	MOV Dn,(d16,Am)	Dn→mem8(d16+Am)	--	--	--	--	7	4		0010	0111	1aDn	<d16	....	....	...>							33
	MOV Dn,(d4,SP)	Dn→mem8(d4+SP)	--	--	--	--	3	2			0111	01Dn	<d4>									*2	33
	MOV Dn,(d8,SP)	Dn→mem8(d8+SP)	--	--	--	--	5	3		0010	0111	01Dn	<d8.	...>								*3	34
	MOV Dn,(d16,SP)	Dn→mem8(d16+SP)	--	--	--	--	7	4		0010	0111	00Dn	<d16	....	....	...>							34
	MOV Dn,(io8)	Dn→mem8(IOTOP+io8)	--	--	--	--	4	2			0111	00Dn	<io8	...>									35
	MOV Dn,(abs8)	Dn→mem8(abs8)	--	--	--	--	4	2			0101	01Dn	<abs	8..>									35
	MOV Dn,(abs12)	Dn→mem8(abs12)	--	--	--	--	5	2			0101	00Dn	<abs	12..	...>								36
	MOV Dn,(abs16)	Dn→mem8(abs16)	--	--	--	--	7	4		0010	1101	00Dn	<abs	16..	....	...>							36
	MOV imm8,(io8)	imm8→mem8(IOTOP+io8)	--	--	--	--	6	3			0000	0010	<io8	...>	<#8.	...>							37
	MOV imm8,(abs8)	imm8→mem8(abs8)	--	--	--	--	6	3			0001	0100	<abs	8..>	<#8.	...>							37
	MOV imm8,(abs12)	imm8→mem8(abs12)	--	--	--	--	7	3			0001	0101	<abs	12..	...>	<#8.	...>						38
	MOV imm8,(abs16)	imm8→mem8(abs16)	--	--	--	--	9	5		0011	1101	1001	<abs	16..	....	...>	<#8.	...>					38
	MOV Dn,(HA)	Dn→mem8(HA)	--	--	--	--	2	2			1101	00Dn											39
MOVW	MOVW (An),DWm	mem16(An)→DWm	--	--	--	--	2	3			1110	00Ad											40
	MOVW (An),Am	mem16(An)→Am	--	--	--	--	3	4		0010	1110	10Aa										*4	40
	MOVW (d4,SP),DWm	mem16(d4+SP)→DWm	--	--	--	--	3	3			1110	011d	<d4>									*2	41
	MOVW (d4,SP),Am	mem16(d4+SP)→Am	--	--	--	--	3	3			1110	010a	<d4>									*2	41
	MOVW (d8,SP),DWm	mem16(d8+SP)→DWm	--	--	--	--	5	4		0010	1110	011d	<d8.	...>								*3	42
	MOVW (d8,SP),Am	mem16(d8+SP)→Am	--	--	--	--	5	4		0010	1110	010a	<d8.	...>								*3	42
	MOVW (d16,SP),DWm	mem16(d16+SP)→DWm	--	--	--	--	7	5		0010	1110	001d	<d16	....	....	...>							43
	MOVW (d16,SP),Am	mem16(d16+SP)→Am	--	--	--	--	7	5		0010	1110	000a	<d16	....	....	...>							43
	MOVW (abs8),DWm	mem16(abs8)→DWm	--	--	--	--	4	3			1100	011d	<abs	8..>									44
	MOVW (abs8),Am	mem16(abs8)→Am	--	--	--	--	4	3			1100	010a	<abs	8..>									44
	MOVW (abs16),DWm	mem16(abs16)→DWm	--	--	--	--	7	5		0010	1100	011d	<abs	16..	....	...>							45
	MOVW (abs16),Am	mem16(abs16)→Am	--	--	--	--	7	5		0010	1100	010a	<abs	16..	....	...>							45
	MOVW DWn,(Am)	DWn→mem16(Am)	--	--	--	--	2	3			1111	00aD											46
	MOVW An,(Am)	An→mem16(Am)	--	--	--	--	3	4		0010	1111	10aA										*4	46
	MOVW DWn,(d4,SP)	DWn→mem16(d4+SP)	--	--	--	--	3	3			1111	011D	<d4>									*2	47
	MOVW An,(d4,SP)	An→mem16(d4+SP)	--	--	--	--	3	3			1111	010A	<d4>									*2	47
	MOVW DWn,(d8,SP)	DWn→mem16(d8+SP)	--	--	--	--	5	4		0010	1111	011D	<d8.	...>								*3	48
	MOVW An,(d8,SP)	An→mem16(d8+SP)	--	--	--	--	5	4		0010	1111	010A	<d8.	...>								*3	48
	MOVW DWn,(d16,SP)	DWn→mem16(d16+SP)	--	--	--	--	7	5		0010	1111	001D	<d16	....	....	...>							49
	MOVW An,(d16,SP)	An→mem16(d16+SP)	--	--	--	--	7	5		0010	1111	000A	<d16	....	....	...>							49
	MOVW DWn,(abs8)	DWn→mem16(abs8)	--	--	--	--	4	3			1101	011D	<abs	8..>									50
	MOVW An,(abs8)	An→mem16(abs8)	--	--	--	--	4	3			1101	010A	<abs	8..>									50
	MOVW DWn,(abs16)	DWn→mem16(abs16)	--	--	--	--	7	5		0010	1101	011D	<abs	16..	....	...>							51
	MOVW An,(abs16)	An→mem16(abs16)	--	--	--	--	7	5		0010	1101	010A	<abs	16..	....	...>							51
	MOVW DWn,(HA)	DWn→mem16(HA)	--	--	--	--	2	3			1001	010D											52
	MOVW An,(HA)	An→mem16(HA)	--	--	--	--	2	3			1001	011A											52
	MOVW imm8,DWm	sign(imm8)→DWm	--	--	--	--	4	2			0000	110d	<#8.	...>								*5	53
	MOVW imm8,Am	zero(imm8)→Am	--	--	--	--	4	2			0000	111a	<#8.	...>								*6	53
	MOVW imm16,DWm	imm16→DWm	--	--	--	--	6	3			1100	111d	<#16	....	....	...>							54

NOTE: Pages for the MN101C Series Instruction Manual

*1 d8 sign-extension *4 A=An, a=Am
*2 d4 zero-extension *5 #8 sign-extension
*3 d8 zero-extension *6 #8 zero-extension

MN101C SERIES INSTRUCTION SET

Group	Mnemonic	Operation	Flag VF	Flag NF	Flag CF	Flag ZF	Code Size	Cycle	Re-peat	extension	Machine Code 1	2	3	4	5	6	7	8	9	10	11	Notes	Page
	MOVW imm16,Am	imm16→Am	--	--	--	--	6	3			1101	111a	<#16	....	....	...>							54
	MOVW SP,Am	SP→Am	--	--	--	--	3	3			0010	0000	100a										55
	MOVW An,SP	An→SP	--	--	--	--	3	3			0010	0000	101A										55
	MOVW DWn,DWm	DWn→DWm	--	--	--	--	3	3			0010	1000	00Dd									*1	56
	MOVW DWn,Am	DWn→Am	--	--	--	--	3	3			0010	0100	11Da										56
	MOVW An,DWm	An→DWm	--	--	--	--	3	3			0010	1100	11Ad										57
	MOVW An,Am	An→Am	--	--	--	--	3	3			0010	0000	00Aa									*2	57
PUSH	PUSH Dn	SP-1→SP,Dn→mem8(SP)	--	--	--	--	2	3			1111	10Dn											58
	PUSH An	SP-2→SP,An→mem16(SP)	--	--	--	--	2	5			0001	011A											58
POP	POP Dn	mem8(SP)→Dn,SP+1→SP	--	--	--	--	2	3			1110	10Dn											59
	POP An	mem16(SP)→An,SP+2→SP	--	--	--	--	2	4			0000	011A											59
EXT	EXT Dn,DWm	sign(Dn)→DWm	--	--	--	--	3	3			0010	1001	000d									*3	60
Arithmetic manupulation instructions																							
ADD	ADD Dn,Dm	Dm+Dn→Dm	●	●	●	●	3	2			0011	0011	DnDm										61
	ADD imm4,Dm	Dm+sign(imm4)→Dm	●	●	●	●	3	2			1000	00Dm	<#4>									*6	61
	ADD imm8,Dm	Dm+imm8→Dm	●	●	●	●	4	2			0000	10Dm	<#8.	...>									62
ADDC	ADDC Dn,Dm	Dm+Dn+CF→Dm	●	●	●	●	3	2	○		0011	1011	DnDm										63
ADDW	ADDW DWn,DWm	DWm+DWn→DWm	●	●	●	●	3	3	○		0010	0101	00Dd									*1	64
	ADDW DWn,Am	Am+DWn→Am	●	●	●	●	3	3	○		0010	0101	10Da										64
	ADDW imm4,Am	Am+sign(imm4)→Am	●	●	●	●	3	2			1110	110a	<#4>									*6	65
	ADDW imm8,Am	Am+sign(imm8)→Am	●	●	●	●	5	3			0010	1110	110a	<#8.	...>							*7	65
	ADDW imm16,Am	Am+imm16→Am	●	●	●	●	7	4			0010	0101	011a	<#16	....	....	...>						66
	ADDW imm4,SP	SP+sign(imm4)→SP	--	--	--	--	3	2			1111	1101	<#4>									*6	66
	ADDW imm8,SP	SP+sign(imm8)→SP	--	--	--	--	4	2			1111	1100	<#8.	...>								*7	67
	ADDW imm16,SP	SP+imm16→SP	--	--	--	--	7	4			0010	1111	1100	<#16	....	....	...>						67
	ADDW imm16,DWm	DWm+imm16→DWm	●	●	●	●	7	4			0010	0101	010d	<#16	....	....	...>						68
ADDUW	ADDUW Dn,Am	Am+zero(Dn)→Am	●	●	●	●	3	3	○		0010	1000	1aDn									*8	69
ADDSW	ADDSW Dn,Am	Am+sign(Dn)→Am	●	●	●	●	3	3	○		0010	1001	1aDn										70
SUB	SUB Dn,Dm(when Dn≠Dm)	Dm-Dn→Dm	●	●	●	●	3	2	○		0010	1010	DnDm										71
	SUB Dn,Dn	Dn-Dn→Dn	0	0	0	1	2	1			1000	01Dn											71
	SUB imm8,Dm	Dm-imm8→Dm	●	●	●	●	5	3			0010	1010	DmDm	<#8.	...>								72
SUBC	SUBC Dn,Dm	Dm-Dn-CF→Dm	●	●	●	●	3	2	○		0010	1011	DnDm										73
SUBW	SUBW DWn,DWm	DWm-DWn→DWm	●	●	●	●	3	3			0010	0100	00Dd									*1	74
	SUBW DWn,Am	Am-DWn→Am	●	●	●	●	3	3			0010	0100	10Da										74
	SUBW imm16,DWm	DWm-imm16→DWm	●	●	●	●	7	4			0010	0100	010d	<#16	....	....	...>						75
	SUBW imm16,Am	Am-imm16→Am	●	●	●	●	7	4			0010	0100	011a	<#16	....	....	...>						75
MULU	MULU Dn,Dm	Dm*Dn→DWk	0	●	●	●	3	8			0010	1111	111D									*4	76
DIVU	DIVU Dn,DWm	DWm/Dn→DWm-l...DWm-h	●	●	●	●	3	9			0010	1110	111d									*5	77
CMP	CMP Dn,Dm	Dm-Dn...PSW	●	●	●	●	3	2			0011	0010	DnDm										78
	CMP imm8,Dm	Dm-imm8...PSW	●	●	●	●	4	2			1100	00Dm	<#8.	...>									78
	CMP imm8,(abs8)	mem8(abs8)-imm8...PSW	●	●	●	●	6	3			0000	0100	<abs	8..>	<#8.	...>							79
	CMP imm8,(abs12)	mem8(abs12)-imm8...PSW	●	●	●	●	7	3			0000	0101	<abs	12..	...>	<#8.	...>						79
	CMP imm8,(abs16)	mem8(abs16)-imm8...PSW	●	●	●	●	9	5			0011	1101	1000	<abs	16..	....	...>	<#8.	...>				80
CMPW	CMPW DWn,DWm	DWm-DWn...PSW	●	●	●	●	3	3			0010	1000	01Dd									*1	81
	CMPW DWn,Am	Am-DWn...PSW	●	●	●	●	3	3			0010	0101	11Da										81
	CMPW An,Am	Am-An...PSW	●	●	●	●	3	3			0010	0000	01Aa									*2	82
	CMPW imm16,DWm	DWm-imm16...PSW	●	●	●	●	6	3			1100	110d	<#16	....	....	...>							82
	CMPW imm16,Am	Am-imm16...PSW	●	●	●	●	6	3			1101	110a	<#16	....	....	...>							83
Logical manipulation instructions																							
AND	AND Dn,Dm	Dm&Dn→Dm	0	●	0	●	3	2			0011	0111	DnDm										84
	AND imm8,Dm	Dm&imm8→Dm	0	●	0	●	4	2			0001	11Dm	<#8.	...>									84
	AND imm8,PSW	PSW&imm8→PSW	●	●	●	●	5	3			0010	1001	0010	<#8.	...>								85
OR	OR Dn,Dm	DmIDn→Dm	0	●	0	●	3	2			0011	0110	DnDm										86
	OR imm8,Dm	Dmlimm8→Dm	0	●	0	●	4	2			0001	10Dm	<#8.	...>									86
	OR imm8,PSW	PSWlimm8→PSW	●	●	●	●	5	3			0010	1001	0011	<#8.	...>								87
XOR	XOR Dn,Dm	Dm^Dn→Dm	0	●	0	●	3	2			0011	1010	DnDm									*9	88
	XOR imm8,Dm	Dm^imm8→Dm	0	●	0	●	5	3			0011	1010	DmDm	<#8.	...>								88

NOTE: Pages for MN101C Series Instruction Manual

*1 D=DWn, d=DWm
*2 A=An, a=Am
*3 d=DWm
*4 D=DWk
*5 D=DWm
*6 #4 sign-extension
*7 #8 sign-extension
*8 Dn zero extension
*9 m=n/

MN101C SERIES INSTRUCTION SET

Group	Mnemonic	Operation	Flag VF	Flag NF	Flag CF	Flag ZF	Code Size	Cycle	Re-peat	Extension	Machine Code 1	2	3	4	5	6	7	8	9	10	11	Notes	Page
NOT	NOT Dn	¯Dn→Dn	0	●	0	●	3	2			0010	0010	10Dn										89
ASR	ASR Dn	Dn.msb→temp,Dn.lsb→CF Dn>>1→Dn,temp→Dn.msb	0	--	●	●	3	2	○		0010	0011	10Dn										90
LSR	LSR Dn	Dn.lsb→CF,Dn>>1→Dn 0→Dn.msb	0	0	●	●	3	2	○		0010	0011	11Dn										91
ROR	ROR Dn	Dn.lsb→temp,Dn>>1→Dn CF→Dn.msb,temp→CF	0	●	●	●	3	2	○		0010	0010	11Dn										92
Bit manipulation instructions																							
BSET	BSET (io8)bp	mem8(IOTOP+io8)&bpdata...PSW 1→mem8(IOTOP+io8)bp	0	●	0	●	5	5			0011	1000	0bp.	<io8	...>								93
	BSET (abs8)bp	mem8(abs8)&bpdata...PSW 1→mem8(abs8)bp	0	●	0	●	4	4				1011	0bp.	<abs	8..>								93
	BSET (abs16)bp	mem8(abs16)&bpdata...PSW 1→mem8(abs16)bp	0	●	0	●	7	6			0011	1100	0bp.	<abs	16..	....	...>						94
BCLR	BCLR (io8)bp	mem8(IOTOP+io8)&bpdata...PSW 0→mem8(IOTOP+io8)bp	0	●	0	●	5	5			0011	1000	1bp.	<io8	...>								95
	BCLR (abs8)bp	mem8(abs8)&bpdata...PSW 0→mem8(abs8)bp	0	●	0	●	4	4				1011	1bp.	<abs	8..>								95
	BCLR (abs16)bp	mem8(abs16)&bpdata...PSW 0→mem8(abs16)bp	0	●	0	●	7	6			0011	1100	1bp.	<abs	16..	....	...>						96
BTST	BTST imm8,Dm	Dm&imm8...PSW	0	●	0	●	5	3			0010	0000	11Dm	<#8.	...>								97
	BTST (abs16)bp	mem8(abs16)&bpdata...PSW	0	●	0	●	7	5			0011	1101	0bp.	<abs	16..	....	...>						97
Branch instructions																							
Bcc	BEQ label	if(ZF=1), PC+3+d4(label)+H→PC if(ZF=0), PC+3→PC	--	--	--	--	3	2/3				1001	000H	<d4>								*1	98
	BEQ label	if(ZF=1), PC+4+d7(label)+H→PC if(ZF=0), PC+4→PC	--	--	--	--	4	2/3				1000	1010	<d7.	...H							*2	98
	BEQ label	if(ZF=1), PC+5+d11(label)+H→PC if(ZF=0), PC+5→PC	--	--	--	--	5	2/3				1001	1010	<d11	....	...H						*3	99
	BNE label	if(ZF=0), PC+3+d4(label)+H→PC if(ZF=1), PC+3→PC	--	--	--	--	3	2/3				1001	001H	<d4>								1	100
	BNE label	if(ZF=0), PC+4+d7(label)+H→PC if(ZF=1), PC+4→PC	--	--	--	--	4	2/3				1000	1011	<d7.	...H							*2	100
	BNE label	if(ZF=0), PC+5+d11(label)+H→PC if(ZF=1), PC+5→PC	--	--	--	--	5	2/3				1001	1011	<d11	....	...H						*3	101
	BGE label	if((VF^NF)=0),PC+4+d7(label)+H→PC if((VF^NF)=1),PC+4→PC	--	--	--	--	4	2/3				1000	1000	<d7.	...H							*2	102
	BGE label	if((VF^NF)=0),PC+5+d11(label)+H→PC if((VF^NF)=1),PC+5→PC	--	--	--	--	5	2/3				1001	1000	<d11	....	...H						*3	102
	BCC label	if(CF=0),PC+4+d7(label)+H→PC if(CF=1), PC+4→PC	--	--	--	--	4	2/3				1000	1100	<d7.	...H							*2	103
	BCC label	if(CF=0), PC+5+d11(label)+H→PC if(CF=1), PC+5→PC	--	--	--	--	5	2/3				1001	1100	<d11	....	...H						*3	103
	BCS label	if(CF=1),PC+4+d7(label)+H→PC if(CF=0), PC+4→PC	--	--	--	--	4	2/3				1000	1101	<d7.	...H							*2	104
	BCS label	if(CF=1), PC+5+d11(label)+H→PC if(CF=0), PC+5→PC	--	--	--	--	5	2/3				1001	1101	<d11	....	...H						*3	104
	BLT label	if((VF^NF)=1),PC+4+d7(label)+H→PC if((VF^NF)=0),PC+4→PC	--	--	--	--	4	2/3				1000	1110	<d7.	...H							*2	105
	BLT label	if((VF^NF)=1),PC+5+d11(label)+H→PC if((VF^NF)=0),PC+5→PC	--	--	--	--	5	2/3				1001	1110	<d11	....	...H						*3	105
	BLE label	if((VF^NF)|ZF=1),PC+4+d7(label)+H→PC if((VF^NF)|ZF=0),PC+4→PC	--	--	--	--	4	2/3				1000	1111	<d7.	...H							*2	106
	BLE label	if((VF^NF)|ZF=1),PC+5+d11(label)+H→PC if((VF^NF)|ZF=0),PC+5→PC	--	--	--	--	5	2/3				1001	1111	<d11	....	...H						*3	106
	BGT label	if((VF^NF)|ZF=0),PC+5+d7(label)+H→PC if((VF^NF)|ZF=1),PC+5→PC	--	--	--	--	5	3/4			0010	0010	0001	<d7.	...H							*2	107

NOTE: Pages for MN101C Series Instruction Manual

*1 d4 sign-extension
*2 d7 sign-extension
*3 d11 sign-extensio

MN101C SERIES INSTRUCTION SET

Group	Mnemonic	Operation	Flag VF	Flag NF	Flag CF	Flag ZF	Code Size	Cycle	Re-peat	Extension	Machine Code 1	2	3	4	5	6	7	8	9	10	11	Notes	Page
Bcc	BGT label	if((VF^NF)|ZF=0),PC+6+d11(label)+H→PC if((VF^NF)|ZF=1),PC+6→PC	--	--	--	--	6	3/4			0010	0011	0001	<d11	....	...H						*3	107
	BHI label	if(CFIZF=0),PC+5+d7(label)+H→PC if(CFIZF=1), PC+5→PC	--	--	--	--	5	3/4			0010	0010	0010	<d7.	...H							*2	108
	BHI label	if(CFIZF=0),PC+6+d11(label)+H→PC if(CFIZF=1), PC+6→PC	--	--	--	--	6	3/4			0010	0011	0010	<d11	....	...H						*3	108
	BLS label	if(CFIZF=1),PC+5+d7(label)+H→PC if(CFIZF=0), PC+5→PC	--	--	--	--	5	3/4			0010	0010	0011	<d7.	...H							*2	109
	BLS label	if(CFIZF=1),PC+6+d11(label)+H→PC if(CFIZF=0), PC+6→PC	--	--	--	--	6	3/4			0010	0011	0011	<d11	....	...H						*3	109
	BNC label	if(NF=0),PC+5+d7(label)+H→PC if(NF=1),PC+5→PC	--	--	--	--	5	3/4			0010	0010	0100	<d7.	...H							*2	110
	BNC label	if(NF=0),PC+6+d11(label)+H→PC if(NF=1),PC+6→PC	--	--	--	--	6	3/4			0010	0011	0100	<d11	....	...H						*3	110
	BNS label	if(NF=1),PC+5+d7(label)+H→PC if(NF=0),PC+5→PC	--	--	--	--	5	3/4			0010	0010	0101	<d7.	...H							*2	111
	BNS label	if(NF=1),PC+6+d11(label)+H→PC if(NF=0),PC+6→PC	--	--	--	--	6	3/4			0010	0011	0101	<d11	....	...H						*3	111
	BVC label	if(VF=0),PC+5+d7(label)+H→PC if(VF=1),PC+5→PC	--	--	--	--	5	3/4			0010	0010	0110	<d7.	...H							*2	112
	BVC label	if(VF=0),PC+6+d11(label)+H→PC if(VF=1),PC+6→PC	--	--	--	--	6	3/4			0010	0011	0110	<d11	....	...H						*3	112
	BVS label	if(VF=1),PC+5+d7(label)+H→PC if(VF=0),PC+5→PC	--	--	--	--	5	3/4			0010	0010	0111	<d7.	...H							*2	113
	BVS label	if(VF=1),PC+6+d11(label)+H→PC if(VF=0),PC+6→PC	--	--	--	--	6	3/4			0010	0011	0111	<d11	....	...H						*3	113
	BRA label	PC+3+d4(label)+H→PC	--	--	--	--	3	3			1110	111H	<d4>									*1	114
	BRA label	PC+4+d7(label)+H→PC	--	--	--	--	4	3			1000	1001	<d7.	...H								*2	114
	BRA label	PC+5+d11(label)+H→PC	--	--	--	--	5	3			1001	1001	<d11	....	...H							*3	115
CBEQ	CBEQ imm8,Dm,label	if(Dm=imm8),PC+6+d7(label)+H→PC if(Dm≠imm8),PC+6→PC	●	●	●	●	6	3/4			1100	10Dm	<#8.	...>	<d7.	...H						*2	116
	CBEQ imm8,Dm,label	if(Dm=imm8),PC+8+d11(label)+H→PC if(Dm≠imm8),PC+8→PC	●	●	●	●	8	4/5			0010	1100	10Dm	<#8.	...>	<d11	....	...H				*3	116
	CBEQ imm8,(abs8),label	if(mem8(abs8)=imm8),PC+9+d7(label)+H→PC if(mem8(abs8)≠imm8),PC+9→PC	●	●	●	●	9	6/7			0010	1101	1100	<abs	8..>	<#8.	...>	<d7.	...H			*2	117
	CBEQ imm8,(abs8),label	if(mem8(abs8)=imm8),PC+10+d11(label)+H→PC if(mem8(abs8)≠imm8),PC+10→PC	●	●	●	●	10	6/7			0010	1101	1101	<abs	8..>	<#8.	...>	<d11	....	...H		*3	117
	CBEQ imm8,(abs16),label	if(mem8(abs16)=imm8),PC+11+d7(label)+H→PC if(mem8(abs16)≠imm8),PC+11→PC	●	●	●	●	11	7/8			0011	1101	1100	<abs	16..	....	...>	<#8.	...>	<d7.	...H	*2	118
	CBEQ imm8,(abs16),label	if(mem8(abs16)=imm8),PC+12+d11(label)+H→PC if(mem8(abs16)≠imm8),PC+12→PC	●	●	●	●	12	7/8			0011	1101	1101	<abs	16..	....	...>	<#8.	...>	<d11	....H	*3	118
CBNE	CBNE imm8,Dm,label	if(Dm≠imm8),PC+6+d7(label)+H→PC if(Dm=imm8),PC+6→PC	●	●	●	●	6	3/4			1101	10Dm	<#8.	...>	<d7.	..H>						*2	119
	CBNE imm8,Dm,label	if(Dm≠imm8),PC+8+d11(label)+H→PC if(Dm=imm8),PC+8→PC	●	●	●	●	8	4/5			0010	1101	10Dm	<#8.	...>	<d11	....	...H				*3	119
	CBNE imm8,(abs8),label	if(mem8(abs8)≠imm8),PC+9+d7(label)+H→PC if(mem8(abs8)=imm8),PC+9→PC	●	●	●	●	9	6/7			0010	1101	1110	<abs	8..>	<#8.	...>	<d7.	...H			*2	120
	CBNE imm8,(abs8),label	if(mem8(abs8)≠imm8),PC+10+d11(label)+H→PC if(mem8(abs8)=imm8),PC+10→PC	●	●	●	●	10	6/7			0010	1101	1111	<abs	8..>	<#8.	...>	<d11	....	...H		*3	120
	CBNE imm8,(abs16),label	if(mem8(abs16)≠imm8),PC+11+d7(label)+H→PC if(mem8(abs16)=imm8),PC+11→PC	●	●	●	●	11	7/8			0011	1101	1110	<abs	16..	....	...>	<#8.	...>	<d7.	...H	*2	121
	CBNE imm8,(abs16),label	if(mem8(abs16)≠imm8),PC+12+d11(label)+H→PC if(mem8(abs16)=imm8),PC+12→PC	●	●	●	●	12	7/8			0011	1101	1111	<abs	16..	....	...>	<#8.	...>	<d11	....H	*3	121
TBZ	TBZ (abs8)bp,label	if(mem8(abs8)bp=0),PC+7+d7(label)+H→PC if(mem8(abs8)bp=1),PC+7→PC	0	●	0	●	7	6/7			0011	0000	0bp.	<abs	8..>	<d7.	...H					*2	122
	TBZ (abs8)bp,label	if(mem8(abs8)bp=0),PC+8+d11(label)+H→PC if(mem8(abs8)bp=1),PC+8→PC	0	●	0	●	8	6/7			0011	0000	1bp.	<abs	8..>	<d11	....	...H				*3	122

*1 d4 sign-extension
*2 d7 sign-extension
*3 d11 sign-extensio

MN101C SERIES INSTRUCTION SET

Group	Mnemonic	Operation	Flag VF	Flag NF	Flag CF	Flag ZF	Code Size	Cycle	Re-peat	Extension	Machine Code 1	2	3	4	5	6	7	8	9	10	11	Notes	Page
TBZ	TBZ (io8)bp,label	if(mem8(IOTOP+io8)bp=0),PC+7+d7(label)+H→PC if(mem8(IOTOP+io8)bp=1),PC+7→PC	0	●	0	●	7	6/7			0011	0100	0bp.	<io8	...>	<d7.	...H					*1	123
	TBZ (io8)bp,label	if(mem8(IOTOP+io8)bp=0),PC+8+d11(label)+H→PC if(mem8(IOTOP+io8)bp=1),PC+8→PC	0	●	0	●	8	6/7			0011	0100	1bp.	<io8	...>	<d11	....	...H				*2	123
	TBZ (abs16)bp,label	if(mem8(abs16)bp=0),PC+9+d7(label)+H→PC if(mem8(abs16)bp=1),PC+9→PC	0	●	0	●	9	7/8			0011	1110	0bp.	<abs	16..	....	...>	<d7.	...H			*1	124
	TBZ (abs16)bp,label	if(mem8(abs16)bp=0),PC+10+d11(label)+H→PC if(mem8(abs16)bp=1),PC+10→PC	0	●	0	●	10	7/8			0011	1110	1bp.	<abs	16..	....	...>	<d11	....	...H		*2	124
TBNZ	TBNZ (abs8)bp,label	if(mem8(abs8)bp=1),PC+7+d7(label)+H→PC if(mem8(abs8)bp=0),PC+7→PC	0	●	0	●	7	6/7			0011	0001	0bp.	<abs	8..>	<d7.	...H					*1	125
	TBNZ (abs8)bp,label	if(mem8(abs8)bp=1),PC+8+d11(label)+H→PC if(mem8(abs8)bp=0),PC+8→PC	0	●	0	●	8	6/7			0011	0001	1bp.	<abs	8..>	<d11	....	...H				*2	125
	TBNZ (io8)bp,label	if(mem8(io)bp=1),PC+7+d7(label)+H→PC if(mem8(io)bp=0),PC+7→PC	0	●	0	●	7	6/7			0011	0101	0bp.	<io8	...>	<d7.	...H					*1	126
	TBNZ (io8)bp,label	if(mem8(io)bp=1),PC+8+d11(label)+H→PC if(mem8(io)bp=0),PC+8→PC	0	●	0	●	8	6/7			0011	0101	1bp.	<io8	...>	<d11	....	...H				*2	126
	TBNZ (abs16)bp,label	if(mem8(abs16)bp=1),PC+9+d7(label)+H→PC if(mem8(abs16)bp=0),PC+9→PC	0	●	0	●	9	7/8			0011	1111	0bp.	<abs	16..	....	...>	<d7.	...H			*1	127
	TBNZ (abs16)bp,label	if(mem8(abs16)bp=1),PC+10+d11(label)+H→PC if(mem8(abs16)bp=0),PC+10→PC	0	●	0	●	10	7/8			0011	1111	1bp.	<abs	16..	....	...>	<d11	....	...H		*2	127
JMP	JMP (An)	0→PC.17-16,An→PC.15-0,0→PC.H	---	---	---	---	3	4			0010	0001	00A0										128
	JMP label	abs18(label)+H→PC	---	---	---	---	7	5			0011	1001	0aaH	<abs	18.b	p15~	0..>					*5	128
JSR	JSR (An)	SP-3→SP,(PC+3).bp7-0→mem8(SP) (PC+3).bp15-8→mem8(SP+1) (PC+3).H→mem8(SP+2).bp7, 0→mem8(SP+2).bp6-2, (PC+3).bp17-16→mem8(SP+2).bp1-0 0→PC.bp17-16 An→PC.bp15-0,0→PC.H	---	---	---	---	3	7			0010	0001	00A1										129
	JSR label	SP-3→SP,(PC+5).bp7-0→mem8(SP) (PC+5).bp15-8→mem8(SP+1) (PC+5).H→mem8(SP+2).bp7, 0→mem8(SP+2).bp6-2, (PC+5).bp17-16→mem8(SP+2).bp1-0 PC+5+d12(label)+H→PC	---	---	---	---	5	6				0001	000H	<d12	....	...>						*3	129
	JSR label	SP-3→SP,(PC+6).bp7-0→mem8(SP) (PC+6).bp15-8→mem8(SP+1) (PC+6).H→mem8(SP+2).bp7, 0→mem8(SP+2).bp6-2, (PC+6).bp17-16→mem8(SP+2).bp1-0 PC+6+d16(label)+H→PC	---	---	---	---	6	7				0001	001H	<d16	....	....	...>					*4	130
	JSR label	SP-3→SP,(PC+7).bp7-0→mem8(SP) (PC+7).bp15-8→mem8(SP+1) (PC+7).H→mem8(SP+2).bp7, 0→mem8(SP+2).bp6-2, (PC+7).bp17-16→mem8(SP+2).bp1-0 abs18(label)+H→PC	---	---	---	---	7	8			0011	1001	1aaH	<abs	18.b	p15~	0..>					*5	130
	JSRV (tbl4)	SP-3→SP,(PC+3).bp7-0→mem8(SP) (PC+3).bp15-8→mem8(SP+1) (PC+3).H→mem8(SP+2).bp7 0→mem8(SP+2).bp6-2, (PC+3).bp17-16→mem8(SP+2).bp1-0 mem8(x'004080+tbl4<<2)→PC.bp7-0 mem8(x'004080+tbl4<<2+1)→PC.bp15-8 mem8(x'004080+tbl4<<2+2).bp7→PC.H mem8(x'004080+tbl4<<2+2).bp1-0→ PC.bp17-16	---	---	---	---	3	9				1111	1110	<t4>									131
NOP	NOP	PC+2→PC	---	---	---	---	2	1	○		0000	0000											132

NOTE: Pages for MN101C00 Series Instruction Manual.

*1 d7 sign-extension
*2 d11 sign-extension
*3 d12 sign-extension
*4 d16 sign-extension
*5 aa=abs18.17 - 16

MN101C SERIES INSTRUCTION SET

Group	Mnemonic	Operation	Flag VF	Flag NF	Flag CF	Flag ZF	Code Size	Cycle	Re-peat	Extension	Machine Code 1	2	3	4	5	6	7	8	9	10	11	Notes	Page
RTS	RTS	mem8(SP)→(PC).bp7-0 mem8(SP+1)→(PC).bp15-8 mem8(SP+2).bp7→(PC).H mem8(SP+2).bp1-0→(PC).bp17-16 SP+3→SP	---	---	---	---	2	7			0000 0001												133
RTI	RTI	mem8(SP)→PSW mem8(SP+1)→(PC).bp7-0 mem8(SP+2)→(PC).bp15-8 mem8(SP+3).bp7→(PC).H mem8(SP+3).bp1-0→(PC).bp17-16 mem8(SP+4)→HA-l mem8(SP+5)→HA-h SP+6→SP	●	●	●	●	2	11			0000 0011												134
Contorl instructions																							
REP	REP imm3	imm3-1→RPC	---	---	---	---	3	2			0010 0001 1rep											*1	135

NOTE: Pages for MN101C Series Instruction Manual. *1 no repeat whn imm3=0, (rep: imm3-1)

Other than the instruction of MN101C Series,the assembler of this Series has the following instructions as macro instructions.
The assembler will interpret the macro instructions below as the assembler instructions.

macro instructions	replaced instructions	remarks
INC Dn	ADD 1,Dn	
DEC Dn	ADD -1,Dn	
INC An	ADDW 1,An	
DEC An	ADDW -1,An	
INC2 An	ADDW 2,An	
DEC2 An	ADDW -2,An	
CLR Dn	SUB Dn,Dm	n=m
ASL D	ADD Dn,Dm	n=m
ROL Dn	ADDC Dn,Dm	n=m
NEG Dn	NOT Dn ADD 1,Dn	
NOPL	MOVW DWn,DWm	n=m
MOV (SP),Dn	MOV (0,SP),Dn	
MOV Dn,(SP)	MOV Dn,(0,SP)	
MOVW (SP),DWn	MOVW (0,SP),DWn	
MOVW DWn,(SP)	MOVW DWn,(0,SP)	
MOVW (SP),An	MOVW (0,SP),An	
MOVW An,(SP)	MOVW An,(0,SP)	

Ver3.1(2001.03.26)

Appendix D MN101C Series Instruction Map

MN101C SERIES INSTRUCTION MAP

1st nibble\2nd nibble

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP	RTS	MOV #8,(io8)	RTI	CMP #8,(abs8)/(abs12)		POP An		ADD #8,Dm				MOVW #8,DWm		MOVW #8,Am	
1	JSR d12(label)		JSR d16(label)		MOV #8,(abs8)/(abs12)		PUSH An		OR #8,Dm				AND #8,Dm			
2	When the exension code is b'oo10'															
3	When the extension code is b'0011'															
4	MOV (abs12),Dm				MOV (abs8),Dm				MOV (An),Dm							
5	MOV Dn,(abs12)				MOV Dn,(abs8)				MOV Dn,(Am)							
6	MOV (io8),Dm				MOV (d4,SP),Dm				MOV (d8,An),Dm							
7	MOV Dn,(io8)				MOV Dn,(d4,SP)				MOV Dn,(d8,Am)							
8	ADD #4,Dm				SUB Dn,Dn				BGE d7	BRA d7	BEQ d7	BNE d7	BCC d7	BCS d7	BLT d7	BLE d7
9	BEQ d4		BNE d4		MOVW DWn,(HA)		MOVW An,(HA)		BGE d11	BRA d11	BEQ d11	BNE d11	BCC d11	BCS d11	BLT d11	BLE d11
A	MOV Dn,Dm / MOV #8,Dm															
B	BSET (abs8)bp								BCLR (abs8)bp							
C	CMP #8,Dm				MOVW (abs8),Am		MOVW (abs8),DWm		CBEQ #8,Dm,d7				CMPW #16,DWm		MOVW #16,DWm	
D	MOV Dn,(HA)				MOVW An,(abs8)		MOVW DWn,(abs8)		CBNE #8,Dm,d7				CMPW #16,Am		MOVW #16,Am	
E	MOVW (An),DWm				MOVW (d4,SP),Am		MOVW (d4,SP),DWm		POP Dn				ADDW #4,Am		BRA d4	
F	MOVW DWn,(Am)				MOVW An,(d4,SP)		MOVW DWn,(d4,SP)		PUSH Dn				ADDW #8,SP	ADDW #4,SP	JSRV (tbl4)	

Extension code: b'0010'

2nd nibble\3rd nibble

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	MOVW An,Am				CMPW An,Am				MOVW SP,Am		MOVW An,SP		BTST #8,Dm			
1	JMP (A0)	JSR (A0)	JMP (A1)	JSR (A1)	MOV PSW,Dm				REP #3							
2		BGT d7	BHI d7	BLS d7	BNC d7	BNS d7	BVC d7	BVS d7	NOT Dn				ROR Dn			
3		BGT d11	BHI d11	BLS d11	BNC d11	BNS d11	BVC d11	BVS d11	ASR Dn				LSR Dn			
4	SUBW DWn,DWm				SUBW #16,DWm		SUBW #16,Am		SUBW DWn,Am				MOVW DWn,Am			
5	ADDW DWn,DWm				ADDW #16,DWm		ADDW #16,Am		ADDW DWn,Am				CMPW DWn,Am			
6	MOV (d16,SP),Dm				MOV (d8,SP),Dm				MOV (d16,An),Dm							
7	MOV Dn,(d16,SP)				MOV Dn,(d8,SP)				MOV Dn,(d16,Am)							
8	MOVW DWn,DWm (NOPL @n=m)				CMPW DWn,DWm				ADDUW Dn,Am							
9	EXT Dn,DWm		AND #8,PSW	OR #8,PSW	MOV Dn,PSW				ADDSW Dn,Am							
A	SUB Dn,Dm / SUB #8,Dm															
B	SUBC Dn,Dm															
C	MOV (abs16),Dm				MOVW (abs16),Am		MOVW (abs16),DWm		CBEQ #8,Dm,d12				MOVW An,DWm			
D	MOV Dn,(abs16)				MOVW An,(abs16)		MOVW DWn,(abs16)		CBNE #8,Dm,d12				CBEQ #8,(abs8),d7/d11		CBNE #8,(abs8),d7/d11	
E	MOVW (d16,SP),Am		MOVW (d16,SP),DWm		MOVW (d8,SP),Am		MOVW (d8,SP),DWm		MOVW (An),Am				ADDW #8,Am		DIVU	
F	MOVW An,(d16,SP)		MOVW DWn,(d16,SP)		MOVW An,(d8,SP)		MOVW DWn,(d8,SP)		MOVW An,(Am)				ADDW #16,SP		MULU	

Extension code: b'0011'

2nd nibble\ 3rd nibble	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	TBZ (abs8)bp,d7								TBZ (abs8)bp,d11							
1	TBNZ (abs8)bp,d7								TBNZ (abs8)bp,d11							
2	CMP Dn,Dm															
3	ADD Dn,Dm															
4	TBZ (io8)bp,d7								TBZ (io8)bp,d11							
5	TBNZ (io8)bp,d7								TBNZ (io8)bp,d11							
6	OR Dn,Dm															
7	AND Dn,Dm															
8	BSET (io8)bp								BCLR (io8)bp							
9	JMP abs18(label)								JSR abs18(label)							
A	XOR Dn,Dm / XOR #8,Dm															
B	ADDC Dn,Dm															
C	BSET (abs16)bp								BCLR (abs16)bp							
D	BTST (abs16)bp								cmp #8,(abs16)	mov #8,(abs16)			CBEQ #8,(abs16),d7/11		CBNE #8,(abs16),d7/11	
E	TBZ (abs16)bp,d7								TBZ (abs16)bp,d11							
F	TBNZ (abs16)bp,d7								TBNZ (abs16)bp,d11							

Ver2.1(2001.03.26)

MN101C46F Revision History

Revision 2.00 to Revision 2.10
July 29, 2001

Page	Description of Revision
Chapter 1General Description	
P1	Changed 42-pin SDIL to 42-pin SDIP.
P2	Corrected number of internal interrupts from 10 to 12.
P3	Changed Package from 42-pin SDIL to 42-pin SDIP in table 1-2.
P3	Added 64-pin LQFP to Package in table 1-2.
P3	Added LQFP064-P-1414 to Package model number in table 1-2..
P3	Added notes to list of description of hardware(table 1-2).
P4	Corrected OSC2 in figure 1-1 from input-output to output.
P5	Added figure 1-2(Flat package version).
P6	Corrected OSC2 in pin description table from iuput-output to output.
P9	Added Conversion relativity accuracy in table 1-6.
P10	Revised figure 1-4.
P11	Deleted description of HSYNC cycle and HSYNC hpase difference.
P12	Added notes of software setup example to table 1-7.
Chapter 2 Basic CPU Functions	
P31	Corrected special register area from 256bytes to 768bytes.
P32	Deleted HVCONDH and HVCONDWH sync separater to registor map.
P41	Added notes of invoking the Standby mode.
Chapter 4 I/O Ports	
P89	Added notes of closed-caption decoder contral bit setup for P2MD.
Chapter 8 Analog/Digital Converter	
P167	Corrected addres of ADICR register from x'03FFB' to x'03FFA'.
Chapter 10 Closed-Caption Decoder	
P175	Revised figure 10-2, 10-3 and 10-4.
P175	Added description of register setup rule to text.
P176	Revised figure 10-5.
P176	Added description of CCD1 to table 10-3.
P177, P179, P181, P182 and P183	Added CCD1 Address to table 10-5, 10-6, 10-7, 10-8 and 10-9.
P185 to P197	Added names and address of CCD1.
P185	Added description of SLCNT register setup to text.
P191	Corrected description of FQSEL register.
P196	Added commend value of HDISTW register.
Appendix A Register Map	
P235	Deleted HVCONDH and HVCONDWH sync separater.

Page	Description of Revision
Appendix C Instruction Set	
P240 to P245	Revised Instruction Set to Latest version.
Appendix D Instruction Map	
P246 to P247	Revised Instruction Map to Latest version.

Revision 1.30 to Revision 2.00
July 19, 2000

Page in Japanese	Description of Revision
Chapter 1General Description	
1-2	Changed original frequency to 14.32 MHz and minimum instruction execution time to 273.3 ns.
1-3	Corrected machine cycle and system clock frequency values. Added description of CCD VSYNC interrupt.
1-4	Revised CCD frequencies.
1-6	Changed input ports on pin configuration to I/O ports.
1-7	Changed input ports in pin description table to I/O ports.
1-8 to 1-18	Corrected electrical specs, adjusting them for new 14.32-MHz maximum frequency.
1-19	Added HSYNC and VSYNC input conditions.
Chapter 2 Basic CPU Functions	
2-1	Changed internal operating frequency to 14.32 MHz and minimum instruction execution time to 273.3 ns.
2-16	Added VBIV0ICR and VBIV1ICR interrupts to register map.
2-18	Adjusted values for new 14.32-MHz maximum frequency.
2-23	Adjusted values for new 14.32-MHz maximum frequency.
2-27	Adjusted values for new 14.32-MHz maximum frequency.
Chapter 3 Interrupts	
3-3	Added table with list of interrupt functions.
3-6, 3-15	Added VBIV0 and VBIV1 interrupt vectors (vector numbers 23 and 24).
3-26	Changed VBI0 interrupt to CCD0 interrupt.
3-27	Changed VBI1 interrupt to CCD1 interrupt.
3-30	Added CCD0 VSYNC (VBIV0) interrupt.
3-31	Added CCD1 VSYNC (VBIV1) interrupt.
Chapter 7 On-Screen Display	
7-4	In note 1, added example of maximum horizontal display characters during NTSC interlacing.
7-5	Changed bit positions in OSDREGE register.
7-6	Adjusted values for new 14.32-MHz maximum frequency.
7-10 to 7-13	Added display examples.

Page in Japanese	Description of Revision
7-14	Changed VRAM bit allocation--(CB code and bit width for blanks/repetitions).
7-17	Changed CRAMEND to RAMEND.
7-36 to 7-37	Added to and modified DMA and interrupt timing description.
7-42	Changed OSDREGE bit positions in shutter movement diagram.
7-54	Changed register name.
Chapter 8 Analog-to-Digital Converter	
8-2 to 8-11	Adjusted values for new 14.32-MHz maximum frequency.
8-15	Corrected unit symbol for capacitor values.
Chapter 10 Closed-Caption Decoder	
10-1	Changed settings for sampling frequency control registers.
10-2	Changed settings for sampling frequency control registers.
10-18, 10-33	Adjusted values for new 14.32-MHz maximum frequency.
Appendices	
15-10	Added VBIV0ICR and VBIV1ICR interrupts to register map.

Revision 1.20 to Revision 1.30
July 26, 1999

Page in Japanese	Description of Revision
Chapter 3 Interrupts	
3-2	Changed number of internal interrupts from 10 to 9.
Chapter 4 I/O Ports	
4-5	Changed notes for P3OUT.
4-12	Changed notes for P2MD.
4-14	Changed description of PCNT2, bits 1 to 0.
4-21	Revised figure 4-3-7.
4-31	Revised figure 4-3-17.
Chapter 6 8-Bit Timers	
6-10	Revised setup example 2.
6-13	Revised setup example 3.
Chapter 7 On-Screen Display	
7-26	Changed hi-z control description in figure 7-5-1.
Chapter 8 Analog-to-Digital Converter	
8-2	Changed names of ADC input pins in text and table 8-1-1.
8-3	Changed names of ADC input pins in figure 8-2-2.
8-6	In figure 8-2-2, changed names of analog input channels for A/D control register 1.
8-8	Changed names of ADC input pins in setup procedures.

Page in Japanese	Description of Revision
8-10	Changed names of ADC input pins in table 8-3-1.
8-12	Changed names of ADC input pins in text. Changed pin setup registers in (1) of setup example. Changed names of ADC input pins in (2) of setup example.
8-13	In figure 8-3-3, changed names of ADC input pins and deleted VREF+ and VREF- pin names. Deleted note on ANBUF0 register from (9) of setup example.
Chapter 12 ROM Correction	
12-5	Added AMCHIM to text. Changed bit numbers in table 12-3-1.
12-10	Clarified that the ROM correction function cannot be emulated in ICE mode.

Revision 1.10 to Revision 1.20 July 19, 1999

Page in Japanese	Description of Revision
Chapter 0 About This Manual	
0-4	Changed example page from image taken from MN101C12G to one from MN101C46F
Chapter 10 Closed-Caption Decoder	
10-5	Changed P24 to P25, P25 to P26, VREFHS to VREFH0, VREFLS to VREFH1.
10-19	Added description of slice level calculation control register, SLCNT.
10-33, 10-34	Added description of slice level calculation control register, SLCNT.
10-39	Added slice level calculation control register, SLCNT.
Chapter 11 IR Remote Signal Receiver	
11-3	In figure 11-1-1, added description of PWM input in SLOW mode.
11-5	Added description of RMTC register setup in SLOW mode.
11-16	Added description of RMTC register setup in SLOW mode.
Chapter 14 Pulse Width Modulator	
14-1	Deleted “8-bit” from chapter title.
14-2 to 14-7	Described 14-bit PWMs.
14-8 to 14-10	Described 8-bit PWMs.
14-2	Added bus width in figure 14-1-1.
14-4	Revised reset values for TDCHL and TDCCLL registers.
14-6	Revised table 14-1-1 and figure 14-1-9.
14-7	Changed “11” to b’11’ and “00” to b’00’.

Revision 1.00 to Revision 1.10
July 12, 1999

Page in Japanese	Description of Revision
Chapter 1General Description	
1-6	Changed fixed polarity of MMOD pin from low to high. Added description of P21 as n-channel, open-drain, 5-volt pin.
1-7	Added list of pin functions (figure 1-3-2).
1-8 to 1-19	Added electrical specification.
Appendices	
15-2 to 15-7	Added description of flash EEPROM version.

MN101C46F/F46F LSI User's Manual Description Record of Changes (Ver.2.1 to 2.11)

page	Line	definition	Description of Changes	
			Former version	New version
Cover	Pub number	C	21446-021E	21446-0211E
Colophon		C	August, 2001 2nd Edition 1st Printing	October, 2001 Ver2.11
Sales office		C		Latest version

<Definition>
A: add
D: delete
C: modify, change

MN101C46F/F46F
LSI User's Manual

October, 2001 Ver2.11

Issued by Matsushita Electric Industrial Co., Ltd.

© Matsushita Electric Industrial Co., Ltd.

Semiconductor Company, Matsushita Electric Industrial Co., Ltd.

Nagaokakyo, Kyoto, 617-8520 Japan
Tel: (075) 951-8151
http://www.panasonic.co.jp/semicon/

SALES OFFICES

■ NORTH AMERICA

●U.S.A. Sales Office:

Panasonic Industrial Company [PIC]

• New Jersey Office:
Two Panasonic Way Secaucus, New Jersey 07094 U.S.A.
Tel: 1-201-348-5257 Fax:1-201-392-4652

• Chicago Office:
1707 N. Randall Road Elgin, Illinois 60123-7847 U.S.A.
Tel: 1-847-468-5720 Fax:1-847-468-5725

• Milpitas Office:
1600 McCandless Drive Milpitas, California 95035 U.S.A.
Tel: 1-408-942-2912 Fax:1-408-946-9063

• Atlanta Office:
1225 Northbrook Parkway Suite 1-151 Suwanee, GA 30024 U.S.A.
Tel: 1-770-338-6953 Fax:1-770-338-6849

• San Diego Office:
9444 Balboa Avenue, Suite 185, San Diego, California 92123 U.S.A.
Tel: 1-619-503-2903 Fax:1-858-715-5545

●Canada Sales Office:

Panasonic Canada Inc. [PCI]
5770 Ambler Drive 27 Mississauga, Ontario, L4W 2T3 CANADA
Tel: 1-905-238-2101 Fax:1-905-238-2414

■ LATIN AMERICA

●Mexico Sales Office:

Panasonic de Mexico, S.A. de C.V. [PANAMEX]
Amores 1120 Col. Del Valle Delegacion Benito Juarez C.P. 03100 Mexico, D.F. MEXICO
Tel: 52-5-488-1000 Fax:52-5-488-1073

• Guadalajara Office:
SUCURSAL GUADALAJARA
Av. Lazaro Cardenas 2305 Local G-102 Plaza Comercial Abastos; Col. Las Torres Guadalajara, Jal. 44920 MEXICO
Tel: 52-3-671-1205 Fax:52-3-671-1256

●Brazil Sales Office:

Panasonic do Brasil Ltda. [PANABRAS]
Caixa Postal 1641, Sao Jose dos Campos, Estado de Sao Paulo
Tel: 55-12-335-9000 Fax:55-12-331-3789

■ EUROPE

●U.K. Sales Office:

Panasonic Industrial Europe Ltd. [PIEL]
Willoughby Road, Bracknell, Berks., RG12 8FP, THE UNITED KINGDOM
Tel: 44-1344-85-3671 Fax:44-1344-85-3853

●Germany Sales Office:

Panasonic Industrial Europe GmbH [PIEG]
Hans-Pinsel-Strasse 2 85540 Haar, GERMANY
Tel: 49-89-46159-119 Fax:49-89-46159-195

■ ASIA

●Singapore Sales Office:

Panasonic Semiconductor of South Asia [PSSA]
300 Beach Road, #16-01, The Concourse, Singapore 199555 THE REPUBLIC OF SINGAPORE
Tel: 65-390-3688 Fax:65-390-3689

●Malaysia Sales Office:

Panasonic Industrial Company (M) Sdn. Bhd. [PICM]

• Head Office:
Tingkat 16B, Menara PKNS Petaling Jaya, No.17, Jalan Yong Shook Lin 46050 Petaling Jaya, Selangor Darul Ehsan, MALAYSIA
Tel: 60-3-7951-6601 Fax:60-3-7954-5968

• Penang Office:
Suite 20-07,20th Floor, MWE Plaza, No.8, Lebuh Farquhar,10200 Penang, Malaysia
Tel: 60-4-201-5113 Fax:60-4-261-9989

• Johore Sales Office:
Menara Pelangi, Suite8.3A, Level8, No.2, Jalan Kuning Taman Pelangi, 80400 Johor Bahru, Johor, MALAYSIA
Tel: 60-7-331-3822 Fax:60-7-355-3996

●Thailand Sales Office:

Panasonic Industrial (THAILAND) Ltd. [PICT]
252-133 Muang Thai-Phatra Complex Building, 31st Fl. Rachadaphisek Rd., Huaykwang, Bangkok 10320, THAILAND
Tel: 66-2-693-3428 Fax:66-2-693-3422

●Philippines Sales Office: [PISP]

Panasonic Indsutrial Sales Philippines Division of Matsushita Electric Philippines Corporation
102 Laguna Boulevard,Bo.Don Jose Laguna Technopark, Santa. Rosa, Laguna 4026 PHILIPPINES
Tel: 63-2-520-8615 Fax:63-2-520-8629

●India Sales Office:

National Panasonic India Ltd. [NPI]
E Block, 510, International Trade Tower Nehru Place, New Delhi_110019 INDIA
Tel: 91-11-629-2870 Fax:91-11-629-2877

●Indonesia Sales Office:

P.T.MET & Gobel [M&G]
JL. Dewi Sartika (Cawang 2) Jakarta 13630, INDONESIA
Tel: 62-21-801-5666 Fax:62-21-801-5675

●Taiwan Sales Office:

Panasonic Industrial Sales (Taiwan) Co.,Ltd. [PIST]

• Head Office:
6F, 550, Sec. 4, Chung Hsiao E. RD. Taipei, 110, TAIWAN
Tel: 886-2-2757-1900 Fax:886-2-2757-1906

• Kaohsiung Office:
6th Floor, Hsin Kong Bldg. No.251, Chi Hsien 1st Road Kaohsiung 800, TAIWAN
Tel: 886-7-346-3815 Fax:886-7-236-8362

●China Sales Office:

Panasonic Industrial (Shanghai) Co., Ltd. [PI(SH)]
Floor 6, Zhong Bao Mansion, 166 East Road Lujian Zui, PU Dong New District, Shanghai, 200120 CHINA
Tel: 86-21-5866-6114 Fax:86-21-5866-8000

Panasonic Industrial (Tianjin) Co., Ltd. [PI(TJ)]
Room No.1001, Tianjin International Building 75, Nanjin Road, Tianjin 300050, CHINA
Tel: 86-22-2313-9771 Fax:86-22-2313-9770

Panasonic SH Industrial Sales (Shenzhen) Co., Ltd. [PSI(SZ)]
7A-107, International Bussiness & Exhibition Centre, Futian Free Trade Zone, Shenzhen 518048, CHINA
Tel: 86-755-359-8500 Fax:86-755-359-8516

Panasonic Shun Hing Industrial Sales (Hong Kong) Co., Ltd. [PSI(HK)]
11th Floor, Great Eagle Center 23 Harbour Road, Wanchai, HONG KONG
Tel: 852-2529-7322 Fax:852-2865-3697

●Korea Sales Office:

Panasonic Industrial Korea Co., Ltd. [PIKL]
Kukje Center Bldg. 11th Fl., 191 Hangangro 2ga, Youngsan-ku, Seoul 140-702, KOREA
Tel: 82-2-795-9600 Fax:82-2-795-1542

© Matsushita Electric Industrial Co., Ltd. 2001

280901
Printed in JAPAN