



MOS INTEGRATED CIRCUIT μ**PD70433**

V55PI™ **16-BIT MICROPROCESSOR**

DESCRIPTION

The µPD70433 (V55PI) is a microprocessor in which a 16-bit CPU, RAM, serial interface, parallel interface, A/D converter, timers; DMA controller, interrupt controller, etc., are integrated in a single chip.

The V55PI is software-compatible with the μPD70320 and 70330 (V25™ and V35™) single-chip microcontrollers. The V55Pl provides a migration path from the V25. It offers higher-level functions and higher performance, and is particularly suitable for control of data processing systems associated with mechanical control, including printer and facsimile.

Detailed functions are described in the following user's manuals, which should be read when carrying out design work.

 V55PI User's Manual Hardware 	: IEU-1418
 V55SC, V55PI User's Manual Instruction 	: IEU-1416

FEATURES

- Internal 16-bit architecture, selectable external data bus width (16/8 bits)
- Software compatible with V20[™] and V30[™] (native mode) and V25 and V35 (includes additional instructions)
- Minimum instruction cycle: 160 ns/12.5 MHz (external 25 MHz)
 - 125 ns/16 MHz (external 32 MHz)
 - 1-Mbyte basic memory space
 - 16-Mbyte extended memory space
- Register file space (in on-chip RAM) : 512 bytes/16 register banks
- I/O space : 64K bytes

• Address space: 16M bytes:-

- Automatic wait control with memory space divided in variable sizes (max, 6 blocks)
- I/O line (input ports: 11 bits, input/output ports: 42 bits)
- DMA controller (DMAC): Max. 4-channel configuration possible
 - Four DMA transfer modes (single transfer, demand release, single step, burst)
 - Intelligent DMA modes 1 and 2
- Serial interface: 2 channels
 - Asynchronous mode (UART) or clocked mode (CSI) selectable
- Parallel interface: 8 bits
 - · Centronics data input/output and general-purpose data input/output
- A/D converter (8 bits): 4 channels
- Real-time output port: 4 bits × 2 channels or 8 bits × 1 channel
- PMW (Pulse Width Modulation) output function : 8 bits

The information in this document is subject to change without notice.

Document No. IC-3670 (O.D.No. IC-82578) Date Published June 1995 P © NEC Corporation 1995 Printed in Japan

6427525 0063107 744 📟

- interrupt controller
 - Programmable priority (4 levels)
 - Three interrupt servicing methods
 - Vectored interrupt function, register bank switching function, macro service function
- 16-bit timer: 4 channels
- Watchdog timer function
- Software interval timer (16 bits)
- Address field wait insertion function and RAS/CAS switchover timing generation function
- DRAM and pseudo-SRAM refresh functions
- Standby functions (STOP mode, HALT mode)
- On-chip clock generator

APPLICATIONS

• Control of data processing systems using serial or parallel communication (Data processing terminals, printer, G3 facsimile, etc.)

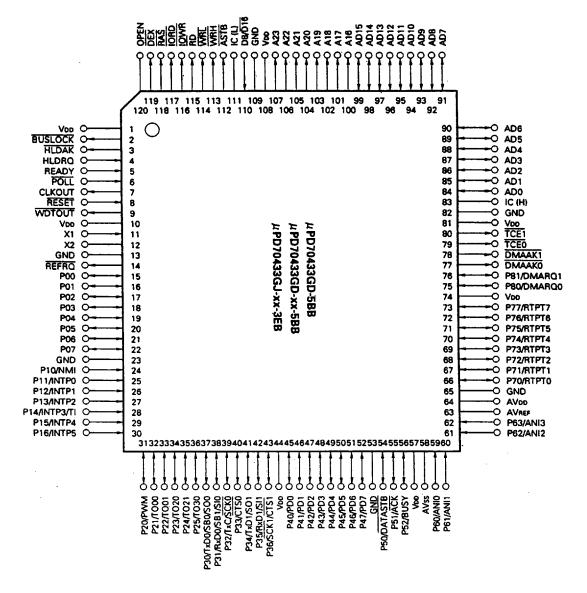
ORDERING INFORMATION

Part Number	Package	Maximum Operating Frequency (MHz)	
μPD70433GD-5BB	120-pin plastic QFP (28 x 28 mm)	12.5	
μPD70433GD-12-5BB	120-pin plastic QFP (28 x 28 mm)	12.5	
μPD70433GD-16-5BB	120-pin plastic QFP (28 x 28 mm)	16	
μPD70433R-12	132-pin ceramic PGA	12.5	
μPD70433R-16	132-pin ceramic PGA	16	
μPD70433GJ-12-3EB	120-pin plastic QFP (20 x 20 mm)	12.5	
μPD70433GJ-16-3EB	120-pin plastic QFP (20 x 20 mm)	16	

🖿 6427525 OO631O8 68O 📟

PIN CONFIGURATION (TOP VIEW)

(1) 120-Pin Plastic QFP (28 x 28 mm), 120-pin plastic QFP (fine pitch) (20 x 20 mm)



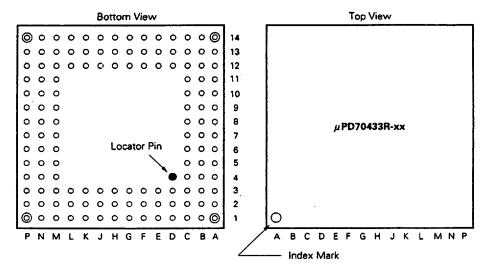
Remark IC: Internally Connected

- Notes 1. The IC (H) pin should be connected to Vod with an external resistor (1 to 10 k Ω).
 - 2. The IC (L) pin should be connected to GND with an external resistor (1 to 10 k Ω).

3. No connection should be made to the OPEN pin.

🔳 6427525 0063109 517 🔳

(2) 132-Pin Ceramic PGA



Remark	The locator pin is not included in the pin co	unt.
--------	---	------

No.	Signal Nane	Port	No.	Signal Name	Port	No.	Signal Name	Port
A1	ANI1	P61	B5	PD7	P47	C9	CTSO	P33
A2	AVss	-	B6	PD5	P45	C10	TO30	P25
A3	ACK	P51	B7	PD2	P42	C11	TO00	P21
A4	DATASTB	P50	B8	PD0	P40	C12	NC	-
A5	PD6	P46	B 9	RxD1/SI1	P 35	C13	INTP4	P15
A6	PD4	P44	B10	RxD0/SB1/SI0	P31	C14	INTP0	P11
A7	PD1	P41	B11	TO21	P24	D1	RTPT2	P72
A 8	NC	1	B12	TO01	P22	D2	GND	-
A9	SCK1/CTS1	P36	B13	NC	—	D3	ANI3	P63
A10	TxD1/SO1	P34	_B14	INTP3/TI	P14	D12	INTP5	P16
A11	TxC/SCK0	P32	C1	RTPT1	P71	D13	INTP2	P13
A12	TxD0/SB0/SO0	P30	C2	AVref	-	D14	NMI	P10
A13	TO20	P23	СЗ	NC	-	E1	RTPT5	P75
A14	PWM	P20	C4	NC	-	E2	RTPT3	P73
B1	AVDO	_	C5	Voo	—	E3	RTPT0	P70
B2	ANI2	P62	C6 [`]	GND	_	E12	INTP1	P12
83	ANIO	P60	C7	PD3	P43	E13	GND	-
B4	BUSY	P52	C8	Vod	-	E14	_	P06

- 4

🖬 6427525 0063110 239 🎟

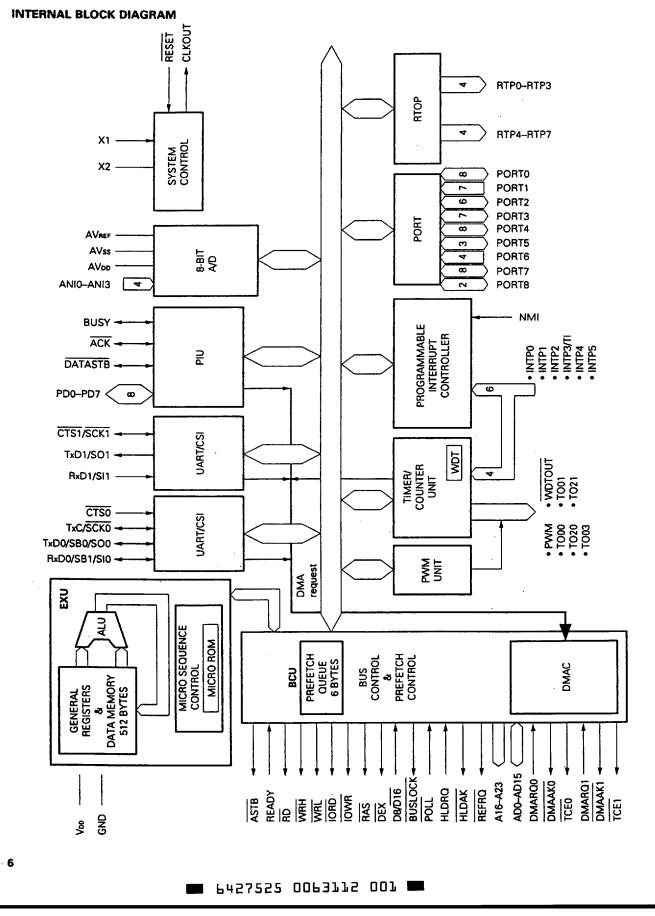
No.	Signal Nane	Port	No.	Signal Name	Port	No.	Signal Name	Port
F1	RTPT7	P77	КЗ	AD2		N3	AD9	
F2	RTPT6	P76	K12	POLL		N4	AD11	—
F3	RTPT4	P74	K13	WDTOUT	—	N5	AD14	_
F12		P07	K14	X1 ·		N6	A18	
F13		P05	L1	AD0		N7	A21	—
F14		P04	L2	AD3		N8	A23	
G1	NC	—	L3	AD6	—	N9	D8/D16	
G2	DMARQO	P80	L12	BUSLOCK	—	N10	ASTB	_
G3	Vod	—	L13	READY		N11	IOWR	
G12	—	P03	L14	RESET		N12	DEX	—
G13		P02	M1	AD1		N13	Voo	_
G14		P01	M2	AD5		N14	HLDRQ	
H1	DMARQ1	P81	М3	NC		P1	AD7	—
H2	DMAAKO	—	M4	AD8		P2	AD10	—
НЗ	DMAAK1	—	M5	AD12	—	P3	AD13	—
H12	REFRO	—	M6	A16	—	P4	AD15	
H13		P00	M7	A20		P5	A17	-
H14	NC	—	M8	Voo		P6	A19	—
J1	TCEO		M9	WRH		P 7	NC	
J2	TCE1		M10	IORD	_	P8	A22	
J3	GND		М11	NC		P9	GND	
J12	Voo		M12	NC		P10	IC (L)	_
J13	X2		М13	HLDAK	-	P11	WRL	
J14	GND		M14	CLKOUT	—	P12	RD	—
К1	Voo		N1	AD4		P13	RAS	_
К2	IC (H)		N2	NC		P14	OPEN	_

Remark IC: Internally Connected NC: Non-Connection

- 2. The IC (L) pin should be connected to GND with an external resistor (1 to 10 k Ω).
 - 3. No connection should be made to the OPEN pin.

🎟 6427525 0063lll 175 🎟

Notes 1. The IC (H) pin should be connected to Vop with an external resistor (1 to 10 k Ω).



CONTENTS

1.1 LIST OF PIN FUNCTION 10 1.1.1 Port Pins 10 1.1.2 Non-Port Pins 11 2. BLOCK CONFIGURATION 14 2.1 BUS CONTROL UNIT (BCU) 14 2.2 EXECUTION UNIT (BCU) 14 2.3 INTERRUPT CONTROLLER (INTC) 14 2.4 DMA CONTROLLER (INTC) 14 2.5 UART/CLOCKED SERIAL INTERFACE (UART/CSI) 14 2.6 PARALLEL INTERACE UNIT (PUI) 14 2.7 A/D CONVERTER UNIT (8-BIT A/D) 14 2.8 TIMER/COUNTER UNIT (TCU) 14 2.9 PUW (PULSE WIDT MODULATION) UNIT (PWM) 14 2.10 WATCHDOG TIMER (WOT) 14 2.11 PORTS (PORT) 14 2.12 CLOCK GENERATOR (CG) 15 2.13 CLOCK GENERATOR (CG) 15 2.14 SOFTWARE INTERVAL TIMER (SIT) 15 3.1 FEATURES 16 3.1 FEATURES 16 3.1 FEATURES 16 3.2.1 Registers (SP, BP) end Index Registers	1.	PIN I	FUNCTIONS	10
1.1.2 Non-Port Pins 11 2. BLOCK CONFIGURATION 14 2.1 BUS CONTROL UNIT (BCU) 14 2.2 EXECUTION UNIT (ECU) 14 2.3 INTERRUPT CONTROLLER (INTC) 14 2.4 DMA CONTROLLER (INTC) 14 2.5 UART/CLOCKED SERIAL INTERFACE (UART/CSI) 14 2.6 PARALLEL INTERFACE (UART/CSI) 14 2.7 A/D CONVERTER UNIT (8-BIT A/D) 14 2.8 PARALLEL INTERFACE UNIT (PUI) 14 2.7 A/D CONVERTER UNIT (8-BIT A/D) 14 2.8 PM (PULSE WIDTH MODULATION) UNIT (PWM) 14 2.9 PWM (PULSE WIDTH MODULATION) UNIT (PWM) 14 2.10 WATCHDOG TIMER (WDT) 14 2.11 PORTS (PORT) 14 2.12 PORTS (WIDT MODULATION) 14 2.13 CLOCK GENERATOR (CG) 15 2.14 SOFTWARE INTERVAL TIMER (SIT) 14 2.13 CLOCK GENERATOR (CG) 15 3.2 POINTEROL (FUNCTIONS 16 3.1 FEATURES 16 <t< th=""><th></th><th>1.1</th><th>LIST OF PIN FUNCTION</th><th></th></t<>		1.1	LIST OF PIN FUNCTION	
2. BLOCK CONFIGURATION 14 2.1 BUS CONTROL UNIT (BCU) 14 2.2 EXECUTION UNIT (EXU) 14 2.3 INTERRUPT CONTROLLER (INTC) 14 2.4 DMA CONTROLLER (INTC) 14 2.5 UART/CLOCKED SERIAL INTERFACE (UART/CSI) 14 2.6 PARALLEL INTERFACE UNIT (PUU) 14 2.7 A/D CONVERTE UNIT (FILI) 14 2.8 TIMER/COUNTRE UNIT (FILI) 14 2.10 WATCHDOG TIMER (WDT) 14 2.11 PORTS (FORT) 14 2.12 REAL-TIME OUTPUT PORT (RTOP) 14 2.13 CLOCK GENERATOR (CG) 15 2.14 SOFTWARE INTERVAL TIMER (SIT) 15 3.1 FEATURES 16 3.1 FEATURES 16 3.1 FEATURES 16 3.2 REGISTERS 17 3.2.3 Pointers (SP, BP) and Index Registers (IX, IY) 20 3.2.4 Segment Registers (DS2, DS1) 20 3.2.5 Extended Segment Registers (IX, ER) 21 3.2.6 Special Function Registers (ISR) 22			1.1.1 Port Pins	10
2.1 BUS CONTROL UNIT (BCU) 14 2.2 EXECUTION UNIT (EXU) 14 2.3 INTERRUPT CONTROLLER (INTC) 14 2.4 DMA CONTROLLER (IMTC) 14 2.5 UART/CLOCKED SERIAL INTERFACE (UART/CSI) 14 2.6 PARALLEL INTERFACE UNIT (PU) 14 2.7 A/D CONVERTER UNIT (B-BIT A/D) 14 2.8 TIMER/COUNTER UNIT (B-BIT A/D) 14 2.9 PWM (PULSE WIDTH MOULLATION) UNIT (PWM) 14 2.9 PWM (PULSE WIDTH MOULLATION) UNIT (PWM) 14 2.10 WATCHDOG TIMER (WDT) 14 2.11 PORTS (PORT) 14 2.12 CLOCK GENERATOR (CG) 15 2.14 SOFTWARE INTERVAL TIMER (SIT) 15 3.1 FEATURES 16 3.1 FEATURES 16 3.2 Pointers (SP, BP) and Index Registers (IX, IY) 20 3.2.1 Registers (SP, S, DSO, DS1) 20 3.2.2 Signert Registers (SF, S) 20 3.2.4 Segment Registers (SF, BP) 21 3.2.5 Extended Segment Registers			1.1.2 Non-Port Pins	11
2.1 BUS CONTROL UNIT (BCU) 14 2.2 EXECUTION UNIT (EXU) 14 2.3 INTERRUPT CONTROLLER (INTC) 14 2.4 DMA CONTROLLER (IMTC) 14 2.5 UART/CLOCKED SERIAL INTERFACE (UART/CSI) 14 2.6 PARALLEL INTERFACE UNIT (PU) 14 2.7 A/D CONVERTER UNIT (B-BIT A/D) 14 2.8 TIMER/COUNTER UNIT (B-BIT A/D) 14 2.9 PWM (PULSE WIDTH MOULLATION) UNIT (PWM) 14 2.9 PWM (PULSE WIDTH MOULLATION) UNIT (PWM) 14 2.10 WATCHDOG TIMER (WDT) 14 2.11 PORTS (PORT) 14 2.12 CLOCK GENERATOR (CG) 15 2.14 SOFTWARE INTERVAL TIMER (SIT) 15 3.1 FEATURES 16 3.1 FEATURES 16 3.2 Pointers (SP, BP) and Index Registers (IX, IY) 20 3.2.1 Registers (SP, S, DSO, DS1) 20 3.2.2 Signert Registers (SF, S) 20 3.2.4 Segment Registers (SF, BP) 21 3.2.5 Extended Segment Registers	2.	BLO	CK CONFIGURATION	14
2.2 EXECUTION UNIT (EXU) 14 2.3 INTERRUPT CONTROLLER (INTC) 14 2.4 DMA CONTROLLER (INTC) 14 2.5 UART/CLOCKED SERIAL INTERFACE (UART/CSI) 14 2.6 PARALLEL INTERFACE UNIT (PU) 14 2.7 A/D CONVERTER UNIT (BIT A/D) 14 2.8 TIMER/COUNTER UNIT (TCU) 14 2.8 TIMER/COUNTER UNIT (TCU) 14 2.9 PWM (PULSE WIDT H MODULATION) UNIT (PWM) 14 2.10 WATCHOOG TIMER (WDT) 14 2.11 PORTS (PORT) 14 2.12 REALTIME OUTPUT PORT (RTOP) 14 2.13 CLOCK GENERATOR (CG) 15 3.1 CPU FUNCTIONS 15 3.2 REGISTERS 17 3.2.1 REGISTERS 17 3.2.2 General Registers (AW, BW, CW, DW) 19 3.2.3 Pointers (SP, BP) and Index Registers (IX, IY) 20 3.2.4 Segment Registers (DS2, DS3) 21 3.2.5 Extended Segment Registers (IX, IY) 20 3.2.4 Special Function Registers (SFR) <td></td> <td></td> <td></td> <td></td>				
2.3 INTERRUPT CONTROLLER (INTC) 14 2.4 DMA CONTROLLER (IMAC) 14 2.5 UART/CLOCKED SERIAL INTERFACE (UART/CSI) 14 2.6 PARALLEL INTERFACE UNIT (PIU) 14 2.7 A/D CONVERTER UNIT (BUT A/D) 14 2.8 TIMER/COUNTER UNIT (FU) 14 2.8 TIMER/COUNTER UNIT (FU) 14 2.9 PWM (PULSE WIDTH MODULATION) UNIT (PWM) 14 2.10 WATCHOOG TIMER (WDT) 14 2.11 PORTS (PORT) 14 2.12 REAL-TIME OUTPUT PORT (RTOP) 14 2.13 CLOCK GENERATOR (CG) 15 2.14 SOFTWARE INTERVAL TIMER (SIT) 15 3. CPU FUNCTIONS 16 3.1 FEATURES 16 3.1 FEATURES 16 3.2 Register Banks 17 3.2.3 Pointers (SF, PP) and Index Registers (IX, IY) 20 3.2.4 Segment Registers (PS, SS, DS, DS, DS1) 20 3.2.5 Extended Segment Registers (DS2, DS3) 21 3.2.6 Special Function Register (SSR)		2.2		
2.4 DMA CONTROLLER (DMAC) 14 2.5 UART/CLOCKED SERIAL INTERFACE (UART/CSI) 14 2.6 PARALLEL INTERFACE (UART/CSI) 14 2.6 PARALLEL INTERFACE (UART/CSI) 14 2.7 A/D CONVERTER UNIT (PU) 14 2.7 A/D CONVERTER UNIT (BU) 14 2.8 TIMER/COUNTER UNIT (CU) 14 2.9 PWM (PULSE WIDTH MODULATION) UNIT (PWM) 14 2.10 WATCHDOG TIMER (WDT) 14 2.11 PORTS (PORT) 14 2.12 REAL-TIME OUTPUT PORT (RTOP) 14 2.13 CLOCK GENERATOR (CG) 15 2.14 SOFTWARE INTERVAL TIMER (SIT) 15 3. CPU FUNCTIONS 16 3.1 FEATURES 16 3.2 Registers (AW, BW, CW, DW) 19 3.2.3 Pointers (SF, BP) and Index Registers (IX, IY) 20 3.2.4 Segment Registers (DS2, DS3) 21 3.2.5 Extended Segmert Registers (IX, IY) 20 3.2.4 Segment Registers (DS2, DS3) 21 3.2.5 Extended Segmert Regis		2.3		
2.5 UART/CLOCKED SERIAL INTERFACE (UART/CSI) 14 2.6 PARALLEL INTERFACE UNIT (PU) 14 2.7 A/D CONVERTER UNIT (AD) 14 2.8 TIMER/COUNTER UNIT (TCU) 14 2.9 PWM (PULSE WIDTH MODULATION) UNIT (PWM) 14 2.10 WATCHDOG TIMER (WDT) 14 2.11 PORTS (PORT) 14 2.12 REAL-TIME OUTPUT PORT (RTOP) 14 2.13 CLOCK GENERATOR (CG) 15 2.14 SOFTWARE INTERVAL TIMER (SIT) 15 3.1 FEATURES 16 3.1 FEATURES 16 3.2.1 Register Banks 17 3.2.2 General Registers (AW, BW, CW, DW) 19 3.2.3 Pointers (SP, BP) and Index Registers (IX, IY) 20 3.2.4 Segment Registers (DS2, DS3) 21 3.2.5 Extended Segment Registers (DS2, DS3) 21 3.2.6 Special Function Registers (SFR) 22 3.2.7 Special Function Registers (SFR) 22 3.2.8 Special Function Registers (SFR) 23 3.4 P				
2.6 PARALLEL INTERFACE UNIT (PIU) 14 2.7 A/D CONVERTER UNIT (FBIT A/D) 14 2.8 TIMER/COUNTER UNIT (CU) 14 2.9 PWM (PULSE WIDTH MODULATION) UNIT (PWM) 14 2.10 WATCHDOG TIMER (WDT) 14 2.11 PORTS (PORT) 14 2.12 REAL-TIME OUTPUT PORT (RTOP) 14 2.13 CLOCK GENERATOR (CG) 15 2.14 SOFTWARE INTERVAL TIMER (SIT) 15 3. CPU FUNCTIONS 16 3.1 FEATURES 16 3.2 REGISTERS 17 3.2.1 REGISTERS 17 3.2.2 General Registers (AW, BW, CW, DW) 19 3.2.3 Pointers (SP, BP) and Index Registers (IX, IY) 20 3.2.4 Segment Registers (DS2, DS3) 21 3.2.5 Extended Segment Registers (SFR) 22 3.3 PROGRAM COUNTER (PC) 23 3.4 PROGRAM STATUS WORDS (FSW) 22 3.5.1 Besic Memory Space 24 3.5.4 Vector Table Area 36		2.5		
2.7 A/D CONVERTER UNIT (8-BIT A/D) 14 2.8 TIMER/COUNTER UNIT (TCU) 14 2.9 PWM (PULSE WIDTH MODULATION) UNIT (PWM) 14 2.10 WATCHDOG TIMER (WDT) 14 2.11 PORTS (PORT) 14 2.12 REAL-TIME OUTPUT PORT (RTOP) 14 2.13 SOFTWARE INTERVAL TIMER (SIT) 15 3. CPU FUNCTIONS 16 3.1 FEATURES 16 3.1 FEATURES 16 3.2 REGISTERS 16 3.2 REGISTERS 17 3.2.1 Register Banks 17 3.2.2 General Registers (AW, BW, CW, DW) 19 3.2.3 Pointers (SP, BP) and Index Registers (IX, IY) 20 3.2.4 Segment Registers (DS2, DS3) 21 3.2.5 Steil Function Registers (DS2, DS3) 21 3.2.6 Special Function Registers (DS2, DS3) 21 3.2.6 Special Function Registers (DS2, DS3) 22 3.3 PROGRAM COUNTER (PC) 23 3.4 PROGRAM STATUS WORDS (PSW) 22				
2.8 TIMER/COUNTER UNIT (TCU) 14 2.9 PWM (PULSE WIDTH MODULATION) UNIT (PWM) 14 2.10 WATCHDOG TIMER (WDT) 14 2.11 PORTS (PORT) 14 2.12 REAL-TIME OUTPUT PORT (RTOP) 14 2.13 CLOCK GENERATOR (CG) 15 2.14 SOFTWARE INTERVAL TIMER (SIT) 15 3. CPU FUNCTIONS 16 3.1 FEATURES 16 3.1 FEATURES 16 3.2.1 Register Banks 17 3.2.2 General Registers (PS, SD, DSO, DS1) 20 3.2.4 Segment Registers (DS2, DS3) 21 3.2.5 Extended Segment Registers (SFR) 22 3.2.6 Special Function Registers (SFR) 23 3.4 PROGRAM COUNTER (PC) 23 3.5.1 Basic Memory Space 24 3.5.2 Extended Memory Space 24 3.5.3 Special Function Register Area 36 3.5.4 Vector Table Area 36 3.5.7 Extended Memory Space 24 3.5.8 <td></td> <td>2.7</td> <td></td> <td></td>		2.7		
2.9 PWM (PULSE WIDTH MODULATION) UNIT (PWM) 14 2.10 WATCHDOG TIMER (WDT) 14 2.11 PORTS (PORT) 14 2.12 REAL-TIME OUTPUT PORT (RTOP) 14 2.13 CLOCK GENERATOR (CG) 15 2.14 SOFTWARE INTERVAL TIMER (SIT) 15 3. CPU FUNCTIONS 16 3.1 FEATURES 16 3.1 FEATURES 16 3.2 REGISTERS 16 3.2.1 Register Banks 17 3.2.2 General Registers (AW, BW, CW, DW) 19 3.2.3 Pointers (SP, BP) and Index Registers (IX, IY) 20 3.2.4 Segment Registers (DS2, DS3) 21 3.2.5 Extended Segment Registers (ISR) 22 3.3 PROGRAM COUNTER (PC) 23 3.4 PROGRAM STATUS WORDS (PSW) 23 3.5 MEMORY SPACE 24 3.5.1 Basic Memory Space 26 3.5.2 Extended Memory Space 26 3.5.3 Special Function Register Area 36 3.5.4		2.8		
2.11 PORTS (PORT) 14 2.12 REAL-TIME OUTPUT PORT (RTOP) 14 2.13 CLOCK GENERATOR (CG) 15 2.14 SOFTWARE INTERVAL TIMER (SIT) 15 3. CPU FUNCTIONS 16 3.1 FEATURES 16 3.1 FEATURES 16 3.2 REGISTERS 17 3.2.1 Register Banks 17 3.2.2 General Registers (AW, BW, CW, DW) 19 3.2.3 Pointers (SP, BP) and Index Registers (IX, IY) 20 3.2.4 Segment Registers (DS2, DS3) 21 3.2.6 Special Function Registers (DS2, DS3) 21 3.2.6 Special Function Registers (SFR) 22 3.3 PROGRAM COUNTER (PC) 23 3.4 PROGRAM STATUS WORDS (PSW) 23 3.5 MEMORY SPACE 24 3.5.1 Basic Memory Space 25 3.5.3 Special Function Register Area 36 3.5.4 Vector Table Area 36 3.5.1 Basic Memory Space 36 3.5.3		2.9		
2.11 PORTS (PORT) 14 2.12 REAL-TIME OUTPUT PORT (RTOP) 14 2.13 CLOCK GENERATOR (CG) 15 2.14 SOFTWARE INTERVAL TIMER (SIT) 15 3. CPU FUNCTIONS 16 3.1 FEATURES 16 3.1 FEATURES 16 3.2 REGISTERS 17 3.2.1 Register Banks 17 3.2.2 General Registers (AW, BW, CW, DW) 19 3.2.3 Pointers (SP, BP) and Index Registers (IX, IY) 20 3.2.4 Segment Registers (DS2, DS3) 21 3.2.6 Special Function Registers (DS2, DS3) 21 3.2.6 Special Function Registers (SFR) 22 3.3 PROGRAM COUNTER (PC) 23 3.4 PROGRAM STATUS WORDS (PSW) 23 3.5 MEMORY SPACE 24 3.5.1 Basic Memory Space 25 3.5.3 Special Function Register Area 36 3.5.4 Vector Table Area 36 3.5.1 Basic Memory Space 36 3.5.3		2.10		
2.12 REAL-TIME OUTPUT PORT (RTOP) 14 2.13 CLOCK GENERATOR (CG) 15 2.14 SOFTWARE INTERVAL TIMER (SIT) 15 3. CPU FUNCTIONS 16 3.1 FEATURES 16 3.2 REGISTERS 17 3.2.1 Register Banks 17 3.2.2 General Registers (AW, BW, CW, DW) 19 3.2.3 Pointers (SP, BP) and Index Registers (IX, IY) 20 3.2.4 Segment Registers (DS, DS, DS1) 20 3.2.5 Extended Segment Registers (DS2, DS3) 21 3.2.6 Special Function Registers (SFR) 22 3.3 PROGRAM COUNTER (PC) 23 3.4 PROGRAM STATUS WORDS (PSW) 23 3.5 MEMORY SPACE 24 3.5.1 Basic Memory Space 24 3.5.2 Extended Memory Space 26 3.5.4 Vector Table Area 36 3.5.1 Basic Memory Space 36 3.5.2 Extended Memory Space 36 3.5.4 Vector Table Area 36 3.5.7<		2.11	PORTS (PORT)	
2.14 SOFTWARE INTERVAL TIMER (SIT) 15 3. CPU FUNCTIONS 16 3.1 FEATURES 16 3.2 REGISTERS 17 3.2.1 Register Banks 17 3.2.2 General Registers (AW, BW, CW, DW) 19 3.2.3 Pointers (SP, BP) and Index Registers (IX, IY) 20 3.2.4 Segment Registers (DS2, DS3) 21 3.2.5 Extended Segment Registers (DS2, DS3) 21 3.2.6 Special Function Registers (DS2, DS3) 21 3.2.6 Special Function Registers (SFR) 22 3.3 PROGRAM COUNTER (PC) 23 3.4 PROGRAM STATUS WORDS (PSW) 23 3.5 MEMORY SPACE 24 3.5.1 Basic Memory Space 25 3.5.3 Special Function Register Area 36 3.5.4 Vector Table Area 36 3.5.4 Vector Table Area 36 3.5.4 Vector Table Area 36 3.5 MUCTION 35 4. BUS CONTROL FUNCTIONS 32 4.		2.12	REAL-TIME OUTPUT PORT (RTOP)	14
3. CPU FUNCTIONS 16 3.1 FEATURES 16 3.2 REGISTERS 17 3.2.1 Register Banks 17 3.2.2 General Registers (AW, BW, CW, DW) 19 3.2.3 Pointers (SP, BP) and Index Registers (IX, IY) 20 3.2.4 Segment Registers (PS, SS, DSO, DS1) 20 3.2.5 Extended Segment Registers (DS2, DS3) 21 3.2.6 Special Function Registers (DS2, DS3) 21 3.2.6 Special Function Registers (SFR) 22 3.3 PROGRAM COUNTER (PC) 23 3.4 PROGRAM STATUS WORDS (PSW) 23 3.5 MEMORY SPACE 24 3.5.1 Basic Memory Space 24 3.5.2 Extended Memory Space 26 3.5.4 Vector Table Area 36 3.5.4 Vector Table Area 36 3.5.4 Vector Table Area 36 3.5.1 WAIT FUNCTIONS 35 3.5.4 Vector Table Area 36 3.5.4 Vector Table Area		2.13	CLOCK GENERATOR (CG)	15
3.1 FEATURES 16 3.2 REGISTERS 17 3.2.1 Register Banks 17 3.2.2 General Registers (AW, BW, CW, DW) 19 3.2.3 Pointers (SP, BP) and Index Registers (IX, IY) 20 3.2.4 Segment Registers (PS, SS, DS0, DS1) 20 3.2.5 Extended Segment Registers (DS2, DS3) 21 3.2.6 Special Function Registers (SFR) 22 3.3 PROGRAM COUNTER (PC) 23 3.4 PROGRAM STATUS WORDS (PSW) 23 3.5 MEMORY SPACE 24 3.5.1 Basic Memory Space 24 3.5.2 Extended Memory Space 26 3.5.4 Vector Table Area 36 3.5 REGISTER FILE SPACE 36 3.7 I/O SPACE 36 3.6 REGISTER FILE SPACE 36 3.7 I/O SPACE 36 3.6 REGISTER FILE SPACE 36 3.7 I/O SPACE 36 3.6 REGISTER FILE SPACE 36 3.7 I/O SPACE 36		2.14	SOFTWARE INTERVAL TIMER (SIT)	15
3.1 FEATURES 16 3.2 REGISTERS 17 3.2.1 Register Banks 17 3.2.2 General Registers (AW, BW, CW, DW) 19 3.2.3 Pointers (SP, BP) and Index Registers (IX, IY) 20 3.2.4 Segment Registers (PS, SS, DS0, DS1) 20 3.2.5 Extended Segment Registers (DS2, DS3) 21 3.2.6 Special Function Registers (SFR) 22 3.3 PROGRAM COUNTER (PC) 23 3.4 PROGRAM STATUS WORDS (PSW) 23 3.5 MEMORY SPACE 24 3.5.1 Basic Memory Space 24 3.5.2 Extended Memory Space 26 3.5.4 Vector Table Area 36 3.5 REGISTER FILE SPACE 36 3.7 I/O SPACE 36 3.6 REGISTER FILE SPACE 36 3.7 I/O SPACE 36 3.6 REGISTER FILE SPACE 36 3.7 I/O SPACE 36 3.6 REGISTER FILE SPACE 36 3.7 I/O SPACE 36				
3.2 REGISTERS 17 3.2.1 Register Banks 17 3.2.2 General Registers (AW, BW, CW, DW) 19 3.2.3 Pointers (SP, BP) and Index Registers (IX, IY) 20 3.2.4 Segment Registers (PS, SS, DS0, DS1) 20 3.2.5 Extended Segment Registers (DS2, DS3) 21 3.2.6 Special Function Registers (SFR) 22 3.3 PROGRAM COUNTER (PC) 23 3.4 PROGRAM STATUS WORDS (PSW) 23 3.5 MEMORY SPACE 24 3.5.1 Basic Memory Space 24 3.5.2 Extended Memory Space 24 3.5.4 Vector Table Area 34 3.6 REGISTER FILE SPACE 36 3.7 I/O SPACE 36 3.7 I/O SPACE 36 4.1 WAIT FUNCTION 35 4.2 REFRESH FUNCTION 35 4.2.1 Refresh Mode Register (RFM) 41 4.2.2 Wait Control in Refresh Cycle 41	3.	CPU		-
3.2.1 Register Banks 17 3.2.2 General Registers (AW, BW, CW, DW) 19 3.2.3 Pointers (SP, BP) and Index Registers (IX, IY) 20 3.2.4 Segment Registers (PS, SS, DS0, DS1) 20 3.2.5 Extended Segment Registers (DS2, DS3) 21 3.2.6 Special Function Registers (SFR) 22 3.3 PROGRAM COUNTER (PC) 23 3.4 PROGRAM STATUS WORDS (PSW) 23 3.5 MEMORY SPACE 24 3.5.1 Basic Memory Space 24 3.5.2 Extended Memory Space 25 3.5.3 Special Function Register Area 26 3.5.4 Vector Table Area 34 3.6 REGISTER FILE SPACE 36 3.7 I/O SPACE 36 3.7 I/O SPACE 36 4.1 WAIT FUNCTIONS 33 4.2 REFRESH FUNCTION 34 4.2 Refresh Mode Register (RFM) 41 4.2.2 Wait Control in Refresh Cycle 41		3.1		
3.2.2 General Registers (AW, BW, CW, DW) 19 3.2.3 Pointers (SP, BP) and Index Registers (IX, IY) 20 3.2.4 Segment Registers (PS, SS, DS0, DS1) 20 3.2.5 Extended Segment Registers (DS2, DS3) 21 3.2.6 Special Function Registers (SFR) 22 3.3 PROGRAM COUNTER (PC) 23 3.4 PROGRAM STATUS WORDS (PSW) 23 3.5 MEMORY SPACE 24 3.5.1 Basic Memory Space 24 3.5.2 Extended Memory Space 25 3.5.3 Special Function Register Area 26 3.5.4 Vector Table Area 34 3.6 REGISTER FILE SPACE 36 3.7 I/O SPACE 36 4.1 WAIT FUNCTIONS 35 4.2 REFRESH FUNCTION 36 4.2 Refresh Mode Register (RFM) 41 4.2.2 Wait Control in Refresh Cycle 41		3.2	REGISTERS	17
3.2.3 Pointers (SP, BP) and Index Registers (IX, IY) 20 3.2.4 Segment Registers (PS, SS, DS0, DS1) 20 3.2.5 Extended Segment Registers (DS2, DS3) 21 3.2.6 Special Function Registers (SFR) 22 3.3 PROGRAM COUNTER (PC) 23 3.4 PROGRAM STATUS WORDS (PSW) 23 3.5 MEMORY SPACE 24 3.5.1 Basic Memory Space 24 3.5.2 Extended Memory Space 25 3.5.3 Special Function Register Area 26 3.5.4 Vector Table Area 34 3.6 REGISTER FILE SPACE 36 3.7 I/O SPACE 38 4. BUS CONTROL FUNCTIONS 39 4.1 WAIT FUNCTION 39 4.2 REFRESH FUNCTION 39 4.2 Register (RFM) 41 4.2.2 Wait Control in Refresh Cycle 41			-	
3.2.4 Segment Registers (PS, SS, DS), DS1) 20 3.2.5 Extended Segment Registers (DS2, DS3) 21 3.2.6 Special Function Registers (SFR) 22 3.3 PROGRAM COUNTER (PC) 23 3.4 PROGRAM STATUS WORDS (PSW) 23 3.5 MEMORY SPACE 24 3.5.1 Basic Memory Space 24 3.5.2 Extended Memory Space 24 3.5.3 Special Function Register Area 26 3.5.4 Vector Table Area 34 3.6 REGISTER FILE SPACE 36 3.7 I/O SPACE 38 4. BUS CONTROL FUNCTIONS 39 4.1 WAIT FUNCTION 39 4.2 REFRESH FUNCTION 39 4.2 Refresh Mode Register (RFM) 41 4.2.1 Refresh Mode Register (RFM) 41 4.2.2 Wait Control in Refresh Cycle 41			•	
3.2.5 Extended Segment Registers (DS2, DS3) 21 3.2.6 Special Function Registers (SFR) 22 3.3 PROGRAM COUNTER (PC) 23 3.4 PROGRAM STATUS WORDS (PSW) 23 3.5 MEMORY SPACE 24 3.5.1 Basic Memory Space 24 3.5.2 Extended Memory Space 25 3.5.3 Special Function Register Area 26 3.5.4 Vector Table Area 34 3.6 REGISTER FILE SPACE 36 3.7 I/O SPACE 38 4. BUS CONTROL FUNCTIONS 39 4.1 WAIT FUNCTION 39 4.2 REFRESH FUNCTION 39 4.2 Refresh Mode Register (RFM) 41 4.2.1 Refresh Cycle 41			-	
3.2.6 Special Function Registers (SFR) 22 3.3 PROGRAM COUNTER (PC) 23 3.4 PROGRAM STATUS WORDS (PSW) 23 3.5 MEMORY SPACE 24 3.5.1 Basic Memory Space 24 3.5.2 Extended Memory Space 25 3.5.3 Special Function Register Area 26 3.5.4 Vector Table Area 34 3.6 REGISTER FILE SPACE 36 3.7 I/O SPACE 38 4. BUS CONTROL FUNCTIONS 39 4.1 WAIT FUNCTION 39 4.2 REFRESH FUNCTION 41 4.2.1 Refresh Mode Register (RFM) 41 4.2.2 Wait Control in Refresh Cycle 41				
3.3 PROGRAM COUNTER (PC) 23 3.4 PROGRAM STATUS WORDS (PSW) 23 3.5 MEMORY SPACE 24 3.5.1 Basic Memory Space 24 3.5.2 Extended Memory Space 25 3.5.3 Special Function Register Area 26 3.5.4 Vector Table Area 34 3.6 REGISTER FILE SPACE 36 3.7 I/O SPACE 36 4. BUS CONTROL FUNCTIONS 39 4.1 WAIT FUNCTION 39 4.2 REFRESH FUNCTION 41 4.2.1 Refresh Mode Register (RFM) 41 4.2.2 Wait Control in Refresh Cycle 41				
3.4 PROGRAM STATUS WORDS (PSW) 23 3.5 MEMORY SPACE 24 3.5.1 Basic Memory Space 24 3.5.2 Extended Memory Space 25 3.5.3 Special Function Register Area 26 3.5.4 Vector Table Area 34 3.6 REGISTER FILE SPACE 36 3.7 I/O SPACE 36 4. BUS CONTROL FUNCTIONS 39 4.1 WAIT FUNCTION 39 4.2 REFRESH FUNCTION 41 4.2.1 Refresh Mode Register (RFM) 41 4.2.2 Wait Control in Refresh Cycle 41			3.2.6 Special Function Registers (SFR)	
3.5 MEMORY SPACE 24 3.5.1 Basic Memory Space 24 3.5.2 Extended Memory Space 25 3.5.3 Special Function Register Area 26 3.5.4 Vector Table Area 34 3.6 REGISTER FILE SPACE 36 3.7 I/O SPACE 38 4. BUS CONTROL FUNCTIONS 39 4.1 WAIT FUNCTION 39 4.2 REFRESH FUNCTION 41 4.2.1 Refresh Mode Register (RFM) 41 4.2.2 Wait Control in Refresh Cycle 41				
3.5.1 Basic Memory Space 24 3.5.2 Extended Memory Space 25 3.5.3 Special Function Register Area 26 3.5.4 Vector Table Area 34 3.6 REGISTER FILE SPACE 36 3.7 I/O SPACE 36 4. BUS CONTROL FUNCTIONS 39 4.1 WAIT FUNCTION 39 4.2 REFRESH FUNCTION 41 4.2.1 Refresh Mode Register (RFM) 41 4.2.2 Wait Control in Refresh Cycle 41				
3.5.2 Extended Memory Space 25 3.5.3 Special Function Register Area 26 3.5.4 Vector Table Area 34 3.6 REGISTER FILE SPACE 36 3.7 I/O SPACE 36 4. BUS CONTROL FUNCTIONS 39 4.1 WAIT FUNCTION 39 4.2 REFRESH FUNCTION 41 4.2.1 Refresh Mode Register (RFM) 41 4.2.2 Wait Control in Refresh Cycle 41		3.5		
3.5.3 Special Function Register Area 26 3.5.4 Vector Table Area 34 3.6 REGISTER FILE SPACE 36 3.7 I/O SPACE 38 4. BUS CONTROL FUNCTIONS 39 4.1 WAIT FUNCTION 39 4.2 REFRESH FUNCTION 41 4.2.1 Refresh Mode Register (RFM) 41 4.2.2 Wait Control in Refresh Cycle 41				
3.5.4 Vector Table Area 34 3.6 REGISTER FILE SPACE 36 3.7 I/O SPACE 38 4. BUS CONTROL FUNCTIONS 39 4.1 WAIT FUNCTION 39 4.2 REFRESH FUNCTION 41 4.2.1 Refresh Mode Register (RFM) 41 4.2.2 Wait Control in Refresh Cycle 41				
3.6 REGISTER FILE SPACE 36 3.7 I/O SPACE 38 4. BUS CONTROL FUNCTIONS 39 4.1 WAIT FUNCTION 39 4.2 REFRESH FUNCTION 41 4.2.1 Refresh Mode Register (RFM) 41 4.2.2 Wait Control in Refresh Cycle 41			• • •	
3.7 I/O SPACE 38 4. BUS CONTROL FUNCTIONS 39 4.1 WAIT FUNCTION 39 4.2 REFRESH FUNCTION 41 4.2.1 Refresh Mode Register (RFM) 41 4.2.2 Wait Control in Refresh Cycle 41				
4. BUS CONTROL FUNCTIONS 39 4.1 WAIT FUNCTION 39 4.2 REFRESH FUNCTION 41 4.2.1 Refresh Mode Register (RFM) 41 4.2.2 Wait Control in Refresh Cycle 41				
4.1 WAIT FUNCTION 39 4.2 REFRESH FUNCTION 41 4.2.1 Refresh Mode Register (RFM) 41 4.2.2 Wait Control in Refresh Cycle 41		3.7	I/O SPACE	38
4.1 WAIT FUNCTION 39 4.2 REFRESH FUNCTION 41 4.2.1 Refresh Mode Register (RFM) 41 4.2.2 Wait Control in Refresh Cycle 41	4.	BUS	CONTROL FUNCTIONS	39
4.2 REFRESH FUNCTION 41 4.2.1 Refresh Mode Register (RFM) 41 4.2.2 Wait Control in Refresh Cycle 41				
4.2.2 Wait Control in Refresh Cycle41		4.2		
			4.2.1 Refresh Mode Register (RFM)	41
4.2.3 Refresh Address				
			4.2.3 Refresh Address	41

🔲 6427525 0063113 T48 📖

5.	INTE	RRUPT FUNCTIONS	42
	5.1	FEATURES	42
	5.2	INTERRUPT RESPONSE METHODS	
		5.2.1 Vectored Interrupts	45
		5.2.2 Register Bank Switching Function	46
		5.2.3 Macro Service Function	47
6.	DMA	FUNCTION (DMA CONTROLLER)	48
	6.1	FEATURES	48
-	050	AL INTERFACE FUNCTIONS	50
7.		FEATURES	
	7.1	PROTOCOLS	
	7.2		
	7.3	UART	
		7.3.1 Features CLOCKED SERIAL INTERFACE (CSI)	
	7.4		
		7.4.1 Features	5Z
8.	PAR	ALLEL INTERFACE FUNCTIONS	53
	8.1	FEATURES	
	0.1		
9.	TIM	R FUNCTION	55
	9.1	FEATURES	55
	9.2	TIMER UNIT CONFIGURATION	
	9.3	REAL-TIME OUTPUT PORT FUNCTION	57
		9.3.1 Real-Time Output Port Configuration	57
		9.3.2 Real-Time Output Port Operation	59
10	-	A UNIT	61
10			
	10.1		
	10.2	PWM UNIT CONFIGURATION	0(
-11	. WA I	CHDOG TIMER FUNCTION	63
	11.1	FEATURES	
	11.2		
12	-	CONVERTER FUNCTION	
	12.1	FEATURES	64
13	. STA	NDBY FUNCTION	
	13.1	HALT MODE	
	13.2	STOP MODE	67
4 4	~~~	CK GENERATOR	60
14			
	14.1	CLOCK GENERATOR CONFIGURATION AND OPERATION	00

6427525 0063114 984 🖬

15.	SOFT	WARE INTERVAL TIMER FUNCTION)
	15.1	SOFTWARE INTERVAL TIMER CONFIGURATION)
16.	CODE	C INSTRUCTION	1
	16.1	FEATURES	1
	16.2	MEMORY MAP	ł
	16.3	PROCESSING FLOW	;
17.	INST	RUCTION SET	3
	17.1	INSTRUCTIONS NEWLY ADDED TO V20/V30 AND V25/35	ł
	17.2	INSTRUCTION SET OPERATIONS	,
	17.3	INSTRUCTION SET TABLE	>
18.	ELEC	TRICAL SPECIFICATIONS128	}
19.	CHAF	RACTERISTIC CURVES (FOR REFERENCE ONLY)165	;
20.	PACK	AGE DRAWINGS	,
21.	RECO	DMMENDED SOLDERING CONDITIONS)

1. PIN FUNCTIONS

1.1 LIST OF PIN FUNCTIONS

1.1.1 Port Pins

Pin Name	Input/Output	Function	Alternate Function
P 00 to P07	Input/output	Port 0 Input/output specifiable bit-wise 8-bit input/output port	
P10*			NMI
P11			INTPO
P12			INTP1
P13	Input	Port 1 7-bit input port	INTP2
P14			INTP3/TI
P15			INTP4
P16			INTP5
P20			PWM
P21			TO00
P22		Port 2	TO01
P23		Input/output specifiable bit-wise 6-bit input/output port Port 3 put/output Input/output specifiable bit-wise 7-bit input/output port	TO20
P24			TO21
P25	-		TO30
P30			TxD0/SB0/SO0
P31			RxD0/SB1/SI0
P32			TxC/SCK0
P33	Input/output		CTS0
P34			TxD1/SO1
P35			RxD1/SI1
P36			CTS1/SCK1
P40 to P47		.Port 4 Input/output specifiable bit-wise 8-bit input/output port	PD0 to PD7
P50		Port 5	DATASTB
P51		Input/output specifiable bit-wise	ACK
P52		3-bit input/output port	BUSY
P60 to P63	Input	Port 6 Input/output specifiable bit-wise 4-bit input/output port	ANIO to ANI3
P70 to P77		Port 7 Input/output specifiable bit-wise 8-bit input/output port	RTP0 to RTP7
P80	Input/output	Port 8 Input/output specifiable bit-wise	DMARQ0
P81		2-bit input/output specifiable bit-wise	DMARQ1

* Unusable as general-purpose port (non-maskable interrupt)

10

🖬 6427525 OO63116 757 📰

1.1.2 Non-Port Pins

(1) Bus control pins

Pin Name	Input/ Output	Function	Alternate Function
ASTB		External bus cycle address strobe signal output in external bus	
RD		External memory cycle data read strobe signal output in external bus	
WRL	Output	External memory cycle lower byte data write strobe signal output in external bus	
WRH		External memory cycle upper byte data write strobe signal output in external bus	
READY	Input	External bus cycle ready signal input in external bus	
DEX		External bus cycle upper byte data enable signal output	
RAS	Output	DRAM low address latch timing signal output	
D8/D16	Input	External bus data bus width selection signal input	
BUSLOCK	Output	External bus bus lock signal output	
POLL		Input of POLL signal (sampled in POLL instruction execution)	
HLDRQ	Input	External bus hold request signal input	:
HLDAK		External bus hold acknowledge signal output	
REFRO	Output	Refresh pulse signal output	
AD0 to AD15	3-state input/output	External bus cycle address/data multiplex signal input/output in external bus	
A16 to A23	3-state output	External bus cycle address signal output in external bus	
IORD	0.1.1	External I/O cycle data read strobe signal output	
IOWR	Output	External I/O cycle data write strobe signal output	
DMARQ0	Ic 4	DMA request signal input (channel 0)	P80
DMARQ1	Input	DMA request signal input (channel 1)	P81
DMAAKO		DMA acknowledge signal output (channel 0)	
DMAAK1		DMA acknowledge signal output (channel 1)	1
TCEO	Output	DMA termination signal output (channel 0)	
TCEI	-	DMA termination signal output (channel 1)	1

🖿 6427525 0063117 693 📟

(2) Other pins

Pin Name	Input/ Output	Function	Alternate Function	
GND		GND potential		
Voo		Positive power supply		
AVes	_	A/D converter GND potential		
AVoo		A/D converter analog power supply		
AVREF		A/D converter reference voltage input		
RESET	Input	System reset signal input		
X1		Connection pins of crystal resonator/ceramic resonator for system clock generation. In case of external clock supply, input		
X2		to X1 and leave X2 open.		
CLKOUT	Output	Internal system clock ø output		
WDTOUT		Watchdog timer overflow signal output		
NMI		Non-maskable interrupt request input *1	P10	
INTP0		nput External interrupt request input *2	P11	
INTP1			P12	
INTP2			P13	
INTP3	Input		P14/TI	
INTP4		•	P15	
INTP5			P16	
ТІ		External event clock input	P14/INTP3	
PWM		PWM output	P20	
TO00, TO01, TO20, TO21, TO30	Output	Timer unit output	P21 to P25	
TxD0		UART transmission data output	P30/SB0/SO0	
RxD0	Input	UART reception data input	P31/SB1/SI0	
TxC	Output	UART transmission clock output	P32/SCK0	
CTSO			P33	
CTS1	Input	UART transmission enable signal input	P36/SCK1	
SB0			P30/TxD0/SO0	
SB1	Input/output	SBI transmission/reception data input/output	P31/RxD0/SI0	

- Because NMI interrupt is unmaskable, NMI interrupt is always initiated by detecting a valid edge (when reading from port 1, the pin level is read).
 - 2. By masking or disabling (IE = 0) these interrupts, these pins can be used as general-purpose input/output ports, respectively.

12

🖬 6427525 0063118 52T 🎟

Pin Name	Input/ Output	Function	Alternate Function
SO0	0		P30/TxD0/SB0
SO1	Output	CSI transmission data output	P34/T×D1
SI0			P31/RxD0/SB1
SI1	Input	CSI reception data input	P35/RxD1
SCKO			P32/TxC
SCK1	-	CSI serial clock input/output	P36/CTS1
PD0 to PD7		Parallel interface — Data input/output	P40 to P47
DATASTB	Input/output	Parallel interface — Data strobe signal	P50
ACK		Parallel interface — Acknowledge signal	P51
BUSY		Parallel interface — Busy signal	P52
ANIO to ANI3	Input	Analog input signal to A/D converter	P60 to P63
RTP0 to RTP7	Output	Real-time output port	P70 to P77

🖿 6427525 OO63119 466 🖿

2. BLOCK CONFIGURATION

2.1 BUS CONTROL UNIT (BCU)

The BCU performs control of the main bus. The BCU starts the necessary internal/external bus cycle on the basis of the physical address obtained from the execution unit (EXU).

2.2 EXECUTION UNIT (EXU)

The EXU controls address calculation, arithmetic and logical operations, data transfer, etc., by means of a microprogram (firmware for controlling the microsequencer on the basis of decoded op code). The EXU contains 512 bytes of RAM (corresponding to the register file space).

2.3 INTERRUPT CONTROLLER (INTC)

The INTC services hardware interrupt requests generated by on-chip peripheral hardware and interrupt requests generated externally with vectored interrupts, bank switching, or macro service. It can also control the programmable 4-level interrupt priority order, and can also perform multiprocessing control for interrupt.

2.4 DMA CONTROLLER (DMAC)

The DMAC is a general-purpose DMA controller, capable of handling the 16M-byte memory space in a linear fashion. Operating modes comprise memory-to-memory transfer mode, intelligent DMA (ring buffer method and counter control method) mode, next address specification mode, and 2-channel operation.

2.5 UART/CLOCKED SERIAL INTERFACE (UART/CSI)

This block supports the asynchronous interface (UART) in which data synchronization is achieved by means of start/stop bits, and the clocked serial interface (CSI), allowing either to be used.

For the clocked serial interface there is a further choice of serial bus interface mode (SBI) or 3-wire serial I/O mode.

2.6 PARALLEL INTERFACE UNIT (PIU)

This performs input/output using strobe signal synchronization in 8-bit units, and supports the Centronics interface and general-purpose parallel data communication functions.

2.7 A/D CONVERTER UNIT (8-BIT A/D)

This is an A/D converter with 4 analog inputs, and provided with 4 A/D conversion result registers.

2.8 TIMER/COUNTER UNIT (TCU)

The timer/counter unit incorporates a 16-bit timer/counter, and can be used as an interval timer, free-running counter, or event counter.

2.9 PWM (PULSE WIDTH MODULATION) UNIT (PWM)

An 8-bit precision PWM (pulse width modulation) signal output function.

2.10 WATCHDOG TIMER (WDT)

The WDT incorporates an 8-bit watchdog timer for detection of inadvertent program looping, system errors, etc. The WDTOUT pin is provided to give external notification of the generation of watchdog timer interrupts.

2.11 PORTS (PORT)

53 port pins are provided, allowing port pin and control pin functions to be selected.

2.12 REAL-TIME OUTPUT PORT (RTOP)

This is a real-time output port which uses an interrupt from timer 0 as a trigger. It can output the contents of the 8-bit buffer register at programmable intervals in 4-bit or 8-bit units.

14

6427525 0063120 188 📟

2.13 CLOCK GENERATOR (CG)

The CG generates a clock at a frequency of 1/2, 1/4, 1/8 or 1/16 that of the crystal and oscillator connected to the X1 and X2 pins and supplies it as the CPU operating clock.

2.14 SOFTWARE INTERVAL TIMER (SIT)

The SIT incorporates a 16-bit software interval timer as a software timer function and watch function timer. Interval interrupts can be set by input clock (count clock) selection and software timer/counter compare register setting.

🔲 6427525 006**3121 01**4 📖

3. CPU FUNCTIONS

The CPU of the V55PI is software upword compatible with the V20 and V30 (native mode), and the V25 and V35.

3.1 FEATURES

- Software upward compatible with V20 & V30 (native mode) and V25 & V35 (includes additional instructions)
- Minimum instruction cycle: 160 ns/12.5 MHz (external 25 MHz clock)

125 ns/16 MHz (external 32 MHz clock)

- ----- 1M-byte basic memory (program) space ----- 16M-byte extended memory (data) space
- Register file space (in on-chip RAM): 512 bytes/16 register banks
- I/O space: 64K bytes
- Register configuration (compared with V20/V30 and V25/V35)

	ltem	V20, V30	V25, V35	V55PI
Extend	led segment register	None	None	D\$2, D\$3
Regist	er bank	None	8 banks (in memory space)	16 banks (in register file space)
	Mode flag	MD	None	None
	Register bank flags	None	RB0 to RB2	RB0 to RB3
PSW	Input/output instruction trap flag	None	IBRK	IBRK
	User flag	None	F0, F1	None
Specia	I function register area	None	240 bytes (memory mapping onto FFF00H to FFFEFH)	496 bytes (memory mapping onto FFE00H to FFFEFH)

- Internal 16-bit architecture, switchable external data bus width (16/8 bits)
- Automatic wait control with memory divided in variable sizes (max. 6 blocks)
- Programmable wait function
- Wait function using READY pin
- Refresh function
 - Automatic generation of refresh cycle (RAS only)
- RAS pin functions
 - RAS pin
 → DRAM RAS timing

 RD, WRH, WRL pins
 → DRAM CAS timing

 ASTB pin
 → DRAM row/column address switching timing

16

🖬 6427525 0063122 T50 페

3.2 REGISTERS

The V55PI CPU has general register sets compatible with the V20 and V30 (native mode), and the V25 and V35. The general register sets are mapped onto the register file space. These general register sets are also used as onchip RAM, and there can be a maximum of 16 register sets in bank form.

In addition, the V55PI has various special function registers for controlling on-chip peripheral hardware. These special function registers are mapped onto memory space addresses 0FFE00H to 0FFFEFH.

3.2.1 Register Banks

The general register sets are mapped onto the register file space (in on-chip RAM). The general register sets are used in a bank arrangement; each bank consists of 32 bytes and up to 16 banks can be set.

The CPU normally uses register bank 15 for program execution, and it is possible to switch to another bank automatically by means of maskable hardware interrupt or software interrupt (BRKCS instruction). It is possible to return from the switched-to register bank to the original register bank by means of the instruction for returning from an interrupt (RETRBI).

The register bank configuration is shown in Figure 3-1. The general register sets are mapped onto the area with an offset of (+08H) to (+1FH) from the start address of each register bank. The word address from the start in a register bank is the extended segment register (DS2) area. The vector PC/DS3 area is used to set the value to be loaded into the PC when the register bank is switched, that is, the offset value of the start address of the interrupt service routine. This area is also used as the extended segment register (DS3) area. The PSW save area is used to save the PSW when the register bank is switched, and the PC save area is used to save the PC when the register bank is switched.

After a reset, register bank 15 is selected automatically. Also, segment register initialization after a reset is performed for register bank 15 only.

The register file space onto which these general register sets are mapped can also be accessed as data memory by addition of a special prefix instruction (IRAM:) to a memory manipulation instruction.

Of the 16 set register banks, banks 0 and 1 have macro service channels (parameter and work area for macro service) allocated in duplicate.

6427525 0063123 997 🚥

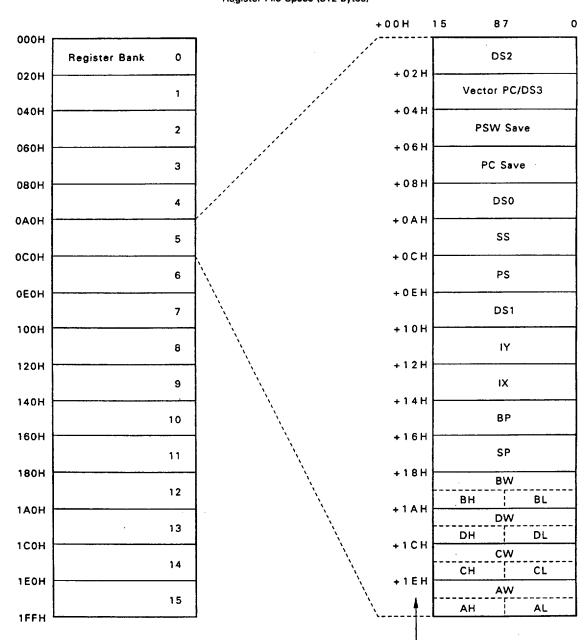


Figure 3-1. Register Bank Configuration

(Offset from the starting address of each register bank)



18

GA 6427525 0063124 823 MM

3.2.2 General Registers (AW, BW, CW, DW)

There are four 16-bit general registers. In addition to being accessed as 16-bit registers, these registers can also be accessed as 8-bit registers by dividing each register into upper and lower 8-bit halves (AH, AL, BH, BL, CH, CL, DH, DL).

These registers are used as 8-bit or 16-bit registers with a wide range of instructions including transfer, arithmetic and logical operation instructions.

Each register is also used as the default register for specific instruction processing, as shown below.

AW: Word multiplication/division, word input/output, data conversion

- AL : Byte multiplication/division, byte input/output, BCD rotation, data conversion
- AH : Byte multiplication/division
- BW : Data conversion

CW : Loop control branch, repeat prefix

CL : Shift instructions, rotate instructions, BCD operations

DW: Word multiplication/division, indirect addressing input/output

These registers are mapped onto the register file space (in on-chip RAM). The address is the value obtained by adding the offset for each register to (register bank number x 32).

Register	Offset	Register	Offset
AW	1EH	AL	1EH
AVV		АН	1FH
BW	18H	BL	18H
BVV	1011	ВН	19H
cw	1СН	CL	1CH
		СН	1DH
DW	1.411	DL	1AH
DW	1AH	DH	1BH

Table 3-1. General Register Offsets

🔳 6427525 0063125 76T 🔳

3.2.3 Pointers (SP, BP) and Index Registers (IX, IY)

These are 16-bit registers used as base pointers or index registers in memory accesses using based addressing (BP), indexed addressing (IX, IY), based indexed addressing (BP, IX, IY), etc. The SP is also used as the pointer in stack operations. As with general registers, these are used with transfer instructions, arithmetic operation instructions, etc., but in this case they cannot be used as 8-bit registers. Each register is also used as the fixed address pointer for specific instruction processing, as shown below.

- SP : Stack manipulation
- IX : Block transfers, BCD operation source side address specification
- IY : Block transfers, BCD operation destination side address specification

These registers are mapped onto the register file space (in on-chip RAM). The address is the value obtained by adding the offset for each register to (register bank number x 32).

Register	Offset
SP	16H
BP	14H
IX	12H
IY	10H

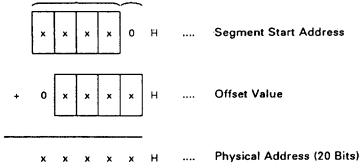
Table 3-2. Pointer and Index Register Offsets

3.2.4 Segment Registers (PS, SS, DS0, DS1)

The CPU manages the 1M-byte basic memory space by dividing it into 64K-byte units. The CPU specifies the start address of each segment with a segment register, and uses another register or effective address for the specification of phyiscal address, with the relative address from the start address as the offset.

The physical address is created as shown below.

Segment Register 4-Bit Fixed



There are four segment registers: PS (Program Segment), SS (Stack Segment), DS0 (Data Segment 0), and DS1 (Data Segment 1). The respective segments are used in the following cases.

- PS : Program fetch
- SS : Stack manipulation instructions, addressing using BP as base register
- DS0: General variable accesses, source block data accesses such as block transfer instructions, etc.
- DS1: Destination block data accesses such as block transfer instructions, etc.
- 20

🖬 6427525 0063126 6T6 🖿

However, using a segment override prefix instruction makes it possible for access of general variables to change from DS0 to another segment register. Also, in addressing which uses BP as the base register, another segment register can be used instead of SS.

Example MOV AW, 1000H MOV DS1 : AW MOV BL, DS1, BYTE PTR (IX); DSI : Byte data read from IX

When a reset is performed, PS of register bank 15 is initialized to FFFFH, and SS, DS0 and DS1 are initialized to 0000H.

These registers are mapped onto the register file space (in on-chip RAM). The address is the value obtained by adding the offset for each register to (register bank number x 32).

Register	Offset
DS0	08H
DS1	0EH
SS	0AH
PS	осн

Table 3-3. Segment Register Offsets

3.2.5 Extended Segment Registers (DS2, DS3)

In addition to the segment registers for accessing the 1M-byte basic memory space, the V55PI is provided with extended segment registers which specify the start address of each 64K-byte segment of the 16M-byte extended memory space. There are two extended segment registers, DS2 (Data Segment 2) and DS3 (Data Segment 3), which are used as shown below.

- DS2: Extended memory space general variable accesses (by segment override prefix instructions), source block data accesses in extended memory space block transfer instructions, etc.
- DS3: Extended memory space general variable accesses (by segment override prefix instructions), destination block data accesses in extended memory space block transfer instructions, etc.

The data access using an extended semgnet register is performed by using the segment override prefix. Especially, in the block transfer instruction, DS2 and DS3 can be specified simultaneously by segment override prefix. (In this case, the order for DS2 and DS3 is optional.)

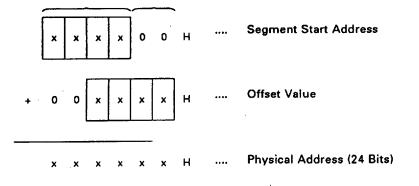
Example REP DS2: DS3: MOVBKW ; Word memory block transfer from DS2 : IX to DS3 : IY.

The CPU specifies the start address of each segment with an extended segment register, and performs an access by using another register or effective address for the specification of physical address, with the relative address from the start address as the offset value.

The physical address is created as shown in the next page.

🖬 6427525 OO63127 532 🖿

Extended Segment Register 8-Bit Fixed



When a reset is performed, DS2 and DS3 of register bank 15 are initialized to 0000H.

These registers are mapped onto the register file space (in on-chip RAM). The address is the value obtained by adding the offset for each register to (register bank number x 32).

Register	Offset
DS2	00H
DS3	02H (Also used as vectored PC)

Table 3-4. Extended Segment Register Offsets

3.2.6 Special Function Registers (SFR)

22

The V55PI has a group of registers with the function of controlling on-chip peripheral hardware.

A number of registers are provided according to the type of cotrol for each peripheral hardware unit, and the actual operation can be set using the individual bits in the registers. These registers are mapped onto the memory space, and are read and written to using the same method as for ordinary memory (see 3.5.3 "Special Function Register Area").

Example MOV AW, 0FFE0H MOV DS1, AW MOV BL, DS1 : BYTE PTR [1EFH]; 0FFE0H : 1EFH (PRC register) Read

There are also two instructions, BTCLR and BTCLRL, which are only valid for special function registers. Of these, BTCLRL is an instruction newly provided in the V25 or V35.

The BTCLR instruction is valid for registers in the upper 240 bytes (0FFF00H to 0FFFEFH) of the special function register area, and the BTCLRL instruction is valid for registers in the lower 256 bytes (0FFE00H to 0FFEFFH).

6427525 0063128 479 📖

3.3 PROGRAM COUNTER (PC)

This is a 16-bit binary counter which holds the offset value of the program memory address on which the CPU is to perform execution.

The PC is incremented each time an instruction code is fetched from the instruction queue, and is also loaded with the new location address value when a branch, call, return or break instruction is executed.

When a reset is performed, 0000H is loaded into the PC. Because the PS register is initialized to FFFFH in a reset, after a reset the CPU begins execution at physical address 0FFFF0H.

3.4 PROGRAM STATUS WORDS (PSW)

The PSW consists of 6 status flags and 5 control flags.

 Status flags 	
• V (Overflow)	Overflow detection flag
•S (Sign)	Sign bit detection flag
•Z (Zero)	All zero detection flag
 AC (Auxiliary Carry) 	4-bit carry/borrow detection flag
• P (Parity)	Parity detection flag
• CY (Carry)	Carry/borrow detection flag
Control flags	
• RB0 to RB3 (Register Banks	0 to 3) Register bankspecification flags
• DIR (Direction)	Block transfer/input/output instruction direction control flag
•IE (Interrupt Enable)	Interrupt enabled state control flag
• BRK (Break)	Single-step interrupt control flag
• IBRK (I/O Break)	Input/output instruction trap control flag

The status flags are set (1) or reset (0) automatically according to the result (data value) of execution of various kinds of instructions. The CY flag can be directly set, reset or inverted by an instruction.

The control flags are set or reset by instructions, and control the operation of the CPU. The IE and BRK flags are always reset when interrupt servicing is initiated.

The contents of the PSW can be saved to and restored from the stack by the PUSH and POP instructions. However, when the contents are restored by the POP PSW instruction, bits 12 to 15 (RB0 to RB3) are not returned to the PSW.

The low-order 8 bits of the PSW can also be saved to or restored from the AH register by an MOV instruction. The PSW bit configuration is shown below.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RB3	RB2	RB1	RBO	Y	DIR	١E	BRK	S	Z	0	AC	0	Р	IBRK	CY

3.5 MEMORY SPACE

The V55PI has a 16M-byte memory space. Of this, using lowest 1M bytes (000000H to 0FFFFFH) as the basic memory space, the 16M bytes including the basic memory space (000000H to FFFFFH) can be accessed as the extended memory space. The basic memory space can be accessed using the segment registers (PS, SS, DS0, DS1) in the same way as in the V25 and V35. The extended memory space can be accessed using the extended segment registers (DS2, DS3), and has the basic memory space mapped onto the lowest 1M bytes. See 3.2.4 "Segment Registers (PS, SS, DS0, DS1)" and 3.2.5 "Extended Segment Registers (DS2, DS3)" for the physical addresses.

The 496-byte space 0FFE00H to 0FFFEFH has mapped onto it a group of registers to which specific functions are allocated such as on-chip peripheral hardware registers, control registers, etc., and these are manipulated by memory accesses.

In addition, independent of these, there is a 512-byte register file space (in on-chip RAM). In addition to being accessed by using register manipulation instructions as in the V25 and V35, the register file space can also be accessed as data memory by adding a special prefix instruction (IRAM:) to a memory manipulation in.

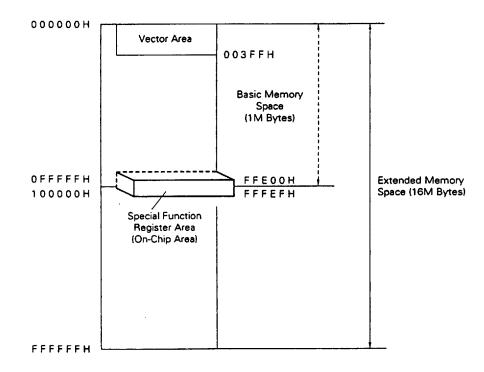


Figure 3-2. Memory Space

3.5.1 Basic Memory Space

The memory space comprises a 1M-byte basic memory space and 16M-byte extended memory space. The basic memory space is mapped onto the lowest 1M bytes (000000H to 0FFFFFH) of the extended memory space.

The 1M-byte basic memory space is shown in Figure 3-3.

Conditions for accessing the basic memory space by software are the same as for the V20/V30 and V25/V35.

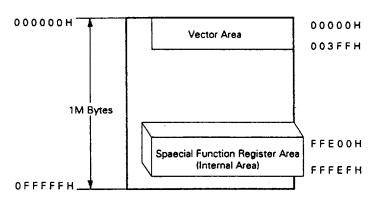
A basic memory space physical address is specified by the segment start address indicated by the segment register (PS, SS, DS0, DS1) and the offset value from the segment start position indicated by another register or immediate data.

The basic memory space has the vectored interrupt vector area and special function register area mapped onto it. For an area in which special function registers are mapped, data accesses cannot be made to external memory (program fetches are possible.)

24

🖬 6427525 0063130 027 🖬





0FFF0H to 0FFFFFH is a program area used for the system boot, and PS and PC become 0FFFH and 0H, respectively, therefore the program execution starts from 0FFFF0H.

3.5.2 Extended Memory Space

The 16M-byte extended memory space is shown in Figure 3-4.

The only accesses that can be performed on the extended memory space are data accesses.

The basic memory space is mapped onto the lowest 1M bytes (000000H to 0FFFFFH) of the extended memory space, and can be accessed using the segment registers PS, SS, DS0 and DS1.

Data accesses can be performed in the extended memory space using the extended segment registers DS2 and DS3. With DS2 and DS3 it is possible to use a specification as a segment override prefix instruction added to a memory manipulation instruction.

An extended memory space physical address is specified by the segment start address indicated by the extended segment register and the offset value from the segment start position indicated by another register or immediate data. If the generated address indicates the lowest 1M-byte area (000000H to 0FFFFFH), the basic memory space is accessed.

🎟 6427525 8863131 T63 🎟

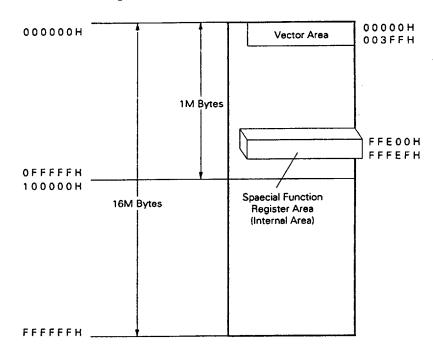


Figure 3-4. Extended Memory Space

3.5.3 Special Function Register Area

The 496-byte space 0FFE00H to 0FFFEFH has mapped onto it a group of registers to which functions such as on-chip peripheral hardware operation specification, status monitoring, etc., are assigned.

Program fetches cannot be performed from these areas.

Special function register manipulation is performed by accesses by means of memory manipulation instructions. If the special function register area is accessed, RD, WRH, WRL, IORD, IOWR and other control signals do not become active.

A list of special function registers is given in Table 3-5. The meaning of the items in the table is explained below.

- Symbol The symbol used to indicate the special function register name. Corresponds to the operand description format (symbol name) in a memory manipulation instruction.
- R/W Indicates whether this special function register is read/write enabled.
 - R/W: Read/write enabled
 - R : Read only
 - W : Write only
- Manipulation Method ... Indicates which of the following can be used on the register: bit manipulation, 8-bit manipulation, 16-bit manipulation, 32-bit manipulation.
- RESET Indicates the status of the register after RESET input.
- Note Addresses which are not listed are the reserved area, therefore, they should not be accessed by the user program.

26

🎟 6427525 0063132 9TT 📟

(1/1)
Registers
Function
Special
Table 3-5.

	Consist Europian Basister Name	Ű	Sumbol	Ŵa	Ma	nipulabl	Manipulable Bit Units	s	Aftar Rosat
		i Ao	100		1 Bit	8 Bits	16 Bits 32 Bits	2 Bits	
OFFEOOH	A/D conversion result register 0	ADCR0		ъ		۲			Undefined
0FFE02H	A/D conversion result register 1	ADCR1		æ		•			Undefined
OFFE04H	A/D conversion result register 2	ADCR2		œ		•			Undefined
OFFE06H	A/D conversion result register 3	ADCR3		œ		•			Undefined
OFFE 10H	Parallel interface buffer	PAD		RW *1		•			Undefined
OFFE 18H	Parallel interface control register 0	PAC0		Å	•	•			H06
OFFE 19H	Parallel interface control register 1	PAC1		RW	•	•			03H
OFFE1AH	Parallel interface status register	PAS		R.W *2		•			40H
OFFE1CH	Parallel interface acknowledge interval register 1	PAI1		≥		•			Undefined
OFFE1DH	Parallel interface acknowledge interval register 2	PAI2		3		•			Undefined
OFFE20H	A/D converter mode register	ADM		RM	•	•			HOO
OFFECOH	Interrupt mask flag register 0 (Iow)	MKD	MKOL	RW	•	•	•		FFH
OFFEC1H	Interrupt mask flag register 0 (high)		МКОН	Ŵ	•	•	•		FFH
0FFEC2H	Interrupt mask flag register 1 (low)	MK1	MK1L	Ŵ	•	•	•	I	FFH
OFFEC3H	Interrupt mask flag register 1 (high)		MK1H	ÅÅ	•	•			FFH
0FFEC4H	In-service priority register	ISPR		æ	•	•			HOD
OFFECSH	Interrupt mode control register	IMC		N ^R		•			80H
0FFEC9H	Interrupt request control register 09	1C09		R/W	•	•			43H
OFFECAH	Interrupt request control register 10	IC10	-	Ŵ	•	•		_	43H
OFFECBH	Interrupt request control register 11	1011		RW	•	•			43H
OFFECCH	Interrupt request control register 12	IC12		RW	•	•			43H
OFFECDH	Interrupt request control register 13	IC13		RW	•	•			43H

🔳 6427525 OO63133 836 🔳

1. Varies according to input/output mode.
2. Some bits R, others R/W (possible).

		C. mbol	Ž	Mar	Ideludir	Manipulable Bit Units		After Beest
Address	Special Function Register Name	ionilike		1 Bit	8 Bits	16 Bits 32	Bits	אונפו המשפו
OFFECEH	Interrupt request control register 14	IC14	RW	•	•			43H
OFFEDOH	Interrupt request control register 16	IC16	RW	•	•			43H
OFFED1H	Interrupt request control register 17	IC17	RW	•	•			43H
OFFED2H	interrupt request control register 18	IC18	RN	•	•			4 3H
0FFED3H	Interrupt request control register 19	IC19	RN	•	•			43H
OFFED4H	Interrupt request control register 20	IC20	RW	•	•			43H
OFFEDSH	Interrupt request control register 21	IC21	RW	•	•			43H
OFFEDGH	Interrupt request control register 22	IC22	RM	•	•			43H
0FFED7H	Interrupt request control register 23	1C23	R/W	•	•			43H
0FFED8H	Interrupt request control register 24	IC24	R/W	•	•			43H
0FFED9H	Interrupt request control register 25	1C25	RW	•	•			43H
OFFEDAH	Interrupt request control register 26	IC26	RW	•	•			43H
OFFEDBH	Interrupt request control register 27	IC27	RW	•	•			43H
OFFEDCH	Interrupt request control register 28	1C28	R/W	•	•			43H
OFFEDDH	Interrupt request control register 29	IC29	R/W	•	•			43H
OFFEDEH	Interrupt request control register 30	1C30	RW	•	•			43H
OFFEDFH	Interrupt request control register 31	IC31	RW	•	•			43H
OFFEEOH	Interrupt request control register 32	IC32	RW	•	•			43H
OFFEE4H	Interrupt request control register 36	IC36	RM	•	•			43H
OFFEESH	Interrupt request control register 37	IC37	R/W	•	•			43H
OFFFOOH	Port 0	PO	RW	•	•			Undefined
0FFF01H	Port 1	14	æ	•	•			Undefined
0FFF02H	Port 2	P2	Ŵ	•	•			Undefined
0FFF03H	Port 3	P3	RW	•	•			Undefined

Table 3-5. Special Function Registers (2/7)

NEC

28

🖬 6427525 0063134 772 🔳

(17)
Registers
Function
Special
3-5.
Fable

		- - - - - - - - - - - - - - - - - - -	NV a	Ma	nipulabl	Manipulable Bit Units	lits	Aftar Racat
Address	Special Function register name	inniiike		1 Bit	8 Bits	16 Bits	32 Bits	
0FFF04H	Port 4	þ4	RW	•	•			Undefined
OFFFOSH	Port 5	P5	R/W	•	•			Undefined
OFFFOGH	Port 6	P6	ж	•	•			Undefined
OFFF07H	Port 7	P7	R/W	•	•			Undefined
OFFF08H	Port 8	P8	Ŵ	•	•			Undefined
0FFF0CH	Port read control register	PRDC	R.W	٠	•			HOO
OFFFOEH	Real-time output port	RTP	R.W	•	•			Undefined
OFFF10H	Port 0 mode register	PMO	R/W	•	•			FFH
0FFF12H	Port 2 mode register	PM2	R/W	•	•			FFH
OFFF13H	Port 3 mode register	PM3	R/W	•	•			FFH
OFFF14H	Port 4 mode register	PM4	R/W	•	•			FFH
OFFF15H	Port 5 mode register	PM5	R/W	•	•			FFH
OFFF17H	Port 7 mode register	PM7	ΜŅ	•	•			FFH
OFFF18H	Port 8 mode register	PM8	RW	•	•			FFH
0FFF22H	Port 2 mode conrol register	PMC2	RW	•	•			HOO
0FFF23H	Port 3 mode control register	PMC3	R.W	•	•			HOO
OFFF24H	Port 4 mode control register	PMC4	RW	•	•			H00
OFFF25H	Port 5 mode control register	PMC5	RW	•	•			H00
0FFF27H	Port 7 mode control register	PMC7	ΜN	•	•			HOO
OFFF28H	Port 8 mode control register	PMC8	RW	•	•			HOO
0FFF2CH	Real-time output port control register	RTPC	R M	•	•			40H
0FFF2DH	Real-time output port delay specification register	ктрр	RW	•	•			Undefined
OFFF2EH	Port 7 buffer (low)	P7L	ΝN	•	•			Undefined
0FFF2FH	Port 7 buffer (high)	P7H	RM	•	•			Undefined

_
E
Ē
Registers
Function
Special
Table 3-5.

AddrassSpecial Function Register Name $Special Function Register Name\operatorname{Special Function Register Name$						Mai	nipulabl	Manipulable Bit Units	lits	
Timer control register 0 TMC0 RW • • • ·	Address	Special Function Register Name	Sym Sym	log	¥	1 Bit		16 Bits	32 Bits	After Keset
Imme control register 1 Imme control register 1 Imme control register 0 Imme control register 10 Imme control register 20 Imme control register 20 Imme control register 20 Imme control register 20 <	OFFF30H	Timer control register 0	TMC0		R M	•	•			H00
Inner output control register 0 IOC $\mathbb{R}W$ \mathbb{C}	OFFF31H	Timer control register 1	TMC1		RW	•	٠	•		HOO
Imme output control register 1 IOC1 \mathbb{RW} \mathbb{K} <	0FFF32H	Timer output control register 0	TOCO		RW	•	•	•		HOO
External interrupt mode register 0 MTM RM e	OFFF33H	Timer output control register 1	TOCI		RW	٠	٠	•		HOO
External interrupt mode register 1 IVIM1 R/W e </td <td>OFFF34H</td> <td>External interrupt mode register 0</td> <td></td> <td>NTMO</td> <td>RM</td> <td>•</td> <td>•</td> <td>•</td> <td></td> <td>HOO</td>	OFFF34H	External interrupt mode register 0		NTMO	RM	•	•	•		HOO
Inter register 0Import <td>OFFF35H</td> <td>External interrupt mode register 1</td> <td></td> <td>NTM1</td> <td>R/W</td> <td>•</td> <td>•</td> <td>•</td> <td></td> <td>HOO</td>	OFFF35H	External interrupt mode register 1		NTM1	R/W	•	•	•		HOO
Timer register 1 M1 MW M ·	OFFF40H	Timer register 0	TMO		R/W			٠		HOO
Titner register 2 Time register 2 MM	0FFF42H	Timer register 1	TM1		RW			•		Hoo
Timer register 3 Timer register 3 Timer register 3 Timer register 3 Timer register 0 R/W N	OFFF44H	Timer register 2	TM2		RW			•		HOO
Timer capture register 00 Timer capture register 01 CT01 RW I I Timer capture register 01 CT01 R/W I I I I Timer capture register 01 CM00 R/W R/W I I I I Timer compare register 01 CM01 R/W R/W I I I I Timer compare register 10 CM10 R/W I	OFFF46H	Timer register 3	TM3		R.W			•		HOO
Timer capture register 01 CT01 R,W · · · Timer compare register 01 CM00 R,W ·	OFFF48H	Timer capture register 00	CT00		RW			•		Undefined
Timer compare register 00 RW N • • • I Timer compare register 01 CM01 RW P • • 1 1 Timer compare register 01 CM01 RW P 1 • 1 1 Timer compare register 10 CM10 RW P 1	OFFF4AH	Timer capture register 01	CT01		RW			•		Undefined
Timer compare register 01 CM01 RW ·	OFFF4CH	Timer compare register 00	CM00		RW			•		Undefined
Timer capture register 10 CT10 R/W · · · Timer compare register 10 CM10 R/W ·	OFFF4EH	Timer compare register 01	CM01		RW			٠		Undefined
Timer compare register 10 CM10 R/W N N N Timer compare register 11 CM11 R/W N	OFFF50H	Timer capture register 10	CT10		RW			•		Undefined
Timer compare register 11 CM11 R/W N • • · <th< td=""><td>OFFF52H</td><td>Timer compare register 10</td><td>CM10</td><td></td><td>RW</td><td></td><td></td><td>•</td><td></td><td>Undefined</td></th<>	OFFF52H	Timer compare register 10	CM10		RW			•		Undefined
Timer compare register 20 CM20 R/W • <	OFFF54H	Timer compare register 11	CM11		RW			•		Undefined
Timer compare register 21 CM21 R/W r · <th< td=""><td>OFFF58H</td><td>Timer compare register 20</td><td>CM20</td><td></td><td>RW</td><td></td><td></td><td>٠</td><td></td><td>Undefined</td></th<>	OFFF58H	Timer compare register 20	CM20		RW			٠		Undefined
Timer compare register 22 CM22 R/W • <	OFFF5AH	Timer compare register 21	CM21		RW			•		Undefined
Timer compare register 23 CM23 R/W • <	OFFF5CH	Timer compare register 22	CM22		R/W			٠		Undefined
Watchdog timer mode register WDM R/W* •	OFFFSEH	Timer compare register 23	CM23		RW			•		Undefined
Timer compare register 30 CM30 R/W e	0FFF60H	Watchdog timer mode register	MDM		R.W*	•	•			HOO
	OFFF64H	Timer compare register 30	CM30		R/W			•		Undefined

🗰 6427525 0063136 545 🔳

				IVVa	Mai	nipulabl	Manipulable Bit Units	its	After Deet
Address	Special Function negister Name	innilike	00	A A / VI	1 Bits	8 Bits	16 Bits 32 Bits	32 Bits	
OFFF66H	Timer compare register 31	CM31		R/W			•		Undefined
0FFF6CH	PWM register	PWM		RΜ	. •	•			HOO
0FFF6DH	PWM control register	PWMC		RW	•	•			HOO
OFFF70H	Transmit baud rate generator register 0	TxBRG0		RW	•	•			Undefined
OFFF71H	Receive baud rate generator register 0	R×BRG0		Ŵ	•	•			Undefined
OFFF72H	Prescaler register 0	PRS0		RŴ	•	•			HOO
OFFF73H	UART mode register 0 / clocked serial interface mode register 0	UARTM0/CSIM0	SIMO	Å	•	•	i		Hoo
OFFF74H	UART status register 0 / SBI control register 0	UARTS0/SBIC0	CC	*1/*2	•	•			HOO
OFFF75H	UART transmit buffer 0 / clocked serial I/O shift register 0	TxB0/SIO0		3		•			Undefined
OFFF76H	Receive buffer 0	R×B0		æ		•			Undefined
OFFF78H	Transmit baud rate generator register 1	T×BRG1	_	Μ	•	•			Undefined
OFFF79H	Receive baud rate generator register 1	RxBRG1		RМ	•	•			Undefined
OFFF7AH	Prescaler register 1	PRS1		ΜR	•	•			HOO
OFFF7BH	UART mode register 1 / clocked serial interface mode register 1	UARTM1/CSIM1	1M1	Ŵ	•	•			HOO
0FFF7CH	UART status register 1	UARTS1		*1/*2	•	•			H00
0FFF7DH	UART transmit buffer 1 / clocked serial I/O shift register 1	TxB1/SI01		3		•			Undefined
0FFF7EH	Receive buffer 1	R×B1		æ		•			Undefined
OFFF7FH	Protocol selection register	ASP		Å	•	•			H00
OFFF80H	Terminal counter 0 (low)		TCOL	RW			•	1	Undefined
0FFF82H	Terminal counter 0 (high)		тсон	RW		•	•	,	Undefined

Table 3-5. Special Function Registers (5/7)

6427525 0063137 481 📟

Remark (): Depends on the mode.

Some bits R, others R/W. R or W in bit units.

÷ 2 ٠

31

(6/7)
legisters
Function A
Special I
Table 3-5.

Transial counter modulo register o (high) TCMOL PW Isit B fits Is B fits </th <th></th> <th></th> <th></th> <th>1040</th> <th>WVa</th> <th>Ma</th> <th>Manipulable Bit Units</th> <th>e Bit Un</th> <th>nits</th> <th>After Beest</th>				1040	WVa	Ma	Manipulable Bit Units	e Bit Un	nits	After Beest
Terminal counter modulo register 0 low) $TCM0$ RW N N N Terminal counter modulo register 0 low) $TCM0$ RW N	Address	Special Function Register Name	ĥ	100		1 Bit	8 Bits	16 Bits	32 Bits	
Terminal counter modulo register 0 (high) Towold Kw w <thw< td=""><td>OFFF84H</td><td>Terminal counter modulo register 0 (low)</td><td>TOMO</td><td>TCMOL</td><td>RW</td><td></td><td></td><td>•</td><td></td><td>Undefined</td></thw<>	OFFF84H	Terminal counter modulo register 0 (low)	TOMO	TCMOL	RW			•		Undefined
$ \begin{array}{ c c c c c c c c c c c c c c c c c c c$	OFFF86H	Terminal counter modulo register 0 (high)		TCMOH	RW		٠	•	•	Undefined
$ \begin{array}{ c c c c c c c c c c c c c c c c c c c$	0FFF88H	DMA up/down counter 0 (low)		UDCOL	RW			•	•	Undefined
$ \begin{array}{ $	OFFFBAH	DMA up/down counter 0 (high)	0000	UDC0H	RW		•	•	•	Undefined
$ \begin{array}{ $	OFFF8CH	DMA compare register 0 (low)	DCMD	DCM0L	RW			•	•	Undefined
$ \begin{array}{ c c c c c c c c c c c c c c c c c c c$	OFFFBEH	DMA compare register 0 (high)		рсмон	R/W		•	•	•	Undefined
DMA memory address register 0 (high) MAR0H R/W R/W • </td <td>OFFF90H</td> <td>DMA memory address register 0 (low)</td> <td></td> <td>MAROL</td> <td>RW</td> <td></td> <td></td> <td>•</td> <td></td> <td>Undefined</td>	OFFF90H	DMA memory address register 0 (low)		MAROL	RW			•		Undefined
$ \begin{array}{ c c c c c c c c c c c c c c c c c c c$	OFFF92H	DMA memory address register 0 (high)		MAROH	RW		٠	٠	•	Undefined
DMA read/write pointer 0 (high) DMA PPTC0H R/W • </td <td>OFFF94H</td> <td>DMA read/write pointer 0 (low)</td> <td>COTOC</td> <td>DPTCOL</td> <td>R/W</td> <td></td> <td></td> <td>•</td> <td>•</td> <td>Undefined</td>	OFFF94H	DMA read/write pointer 0 (low)	COTOC	DPTCOL	R/W			•	•	Undefined
DMA mode register 0 DMA mode register 0 RM \bullet <	OFFF96H	DMA read/write pointer 0 (high)	2	DPTCOH	RW		•	•	•	Undefined
DMA control register 0 DMAC R/W • <td< td=""><td>OFFF9CH</td><td>DMA mode register 0</td><td>DMAMO</td><td></td><td>RW</td><td>•</td><td>٠</td><td></td><td></td><td>EOH</td></td<>	OFFF9CH	DMA mode register 0	DMAMO		RW	•	٠			EOH
$ \begin{array}{cccccccccccccccccccccccccccccccccccc$	OFFF9DH	DMA control register 0	DMACO		R/W	•	•			HOO
Terminal counter 1 (low) TC1 RW •	OFFF9EH	DMA status register	DMAS		R.W	*	•			HOO
Terminal counter 1 (high) TCIH RW • <	OFFFAOH	Terminal counter 1 (low)	TC1	тсіг	R/W			•	•	Undefined
Terminal counter modulo register 1 (low) TCM1 R/W \bullet	0FFFA2H	Terminal counter 1 (high)		тстн	R/W		•	•		Undefined
Terminal counter modulo register 1 (high)TCM1HR/W•• </td <td>OFFFA4H</td> <td></td> <td>TCM1</td> <td>TCM1L</td> <td>RW</td> <td></td> <td></td> <td>•</td> <td></td> <td>Undefined</td>	OFFFA4H		TCM1	TCM1L	RW			•		Undefined
DMA up/down counter 1 (low) UDC1L R/W • • DMA up/down counter 1 (high) UDC1 R/W • • • DMA compare register 1 (low) DCM1 R/W • • • • DMA compare register 1 (low) DCM1 R/W • • • • • DMA compare register 1 (low) DCM1 R/W • • • • • • DMA memory address register 1 (low) MAR1 R/W • <td>OFFFAGH</td> <td></td> <td>1.5.1</td> <td>тсмін</td> <td>R.W</td> <td></td> <td>•</td> <td>•</td> <td>-</td> <td>Undefined</td>	OFFFAGH		1.5.1	тсмін	R.W		•	•	-	Undefined
DMA up/down counter 1 (high) UDC1H R/W •	OFFFA8H	DMA up/down counter 1 (iow)		UDC1L	R/W			•	•	Undefined
DMA compare register 1 (low) DCM1 R/W • • DMA compare register 1 (high) DCM1 R/W • • • DMA memory address register 1 (low) MAR1 R/W • • • • DMA memory address register 1 (high) MAR1 R/W • • • •	OFFFAAH	DMA up/down counter 1 (high)		UDC1H	RW		•	•	- - -	Undefined
DMA compare register 1 (high) DCM1H R/W •	OFFFACH	DMA compare register 1 (low)	1WJQ	DCM1L	RM			•	•	Undefined
DMA memory address register 1 (low) MAR1 R/W • DMA memory address register 1 (high) MAR1 R/W • •	OFFFAEH	DMA compare register 1 (high)	1000	DCM1H	R.W		•	•	,	Undefined
DMA memory address register 1 (high)	OFFFBOH	DMA memory address register 1 (low)		MAR1L	RW			•	•	Undefined
	0FFFB2H	DMA memory address register 1 (high)		MAR1H	RM		•	•		Undefined

■ 6427525 0063138 318 ■

_
5
E
5
ü
Ť
Ë
ĕ
E ,
5
÷
ဥ
5
Ľ
a
Ξ.
ă
S
3-2
ble
a,
F

DMA read/write po DMA read/write po DMA mode regist DMA control regis Software timer/co Programmable wa Programmable wa Programmable wa Refresh mode regi		Constinue Radister Nama	2vn	Svmbol	MA	Ма	Manipulable Bit Units	e Bit Ur	nits	After Reset
DMA read/write pointer 1 (low)DMA read/write pointer 1 (high)DMA mode register 1DMA control register 1DMA control register 1Software timer/counterSoftware timer/counterProgrammable wait control registerProgrammable wait control registerRefresh mode registerStandby control register	Address		5			1 Bit	8 Bits	16 Bits	16 Bits 32 Bits	
DMA read/write pointer 1 (high)DMA mode register 1DMA control register 1Software timer/counterSoftware timer/counter compare reiProgrammable wait control registerProgrammable wait control registerRefresh mode registerStandby control register	OFFFB4H	DMA read/write pointer 1 (low)		DPTC1L	R/W			•		Undefined
DMA mode register 1 DMA control register 1 Software timer/counter Software timer/counter compare re Programmable wait control register Programmable wait control register Memory block control register Refresh mode register Standby control register	OFFFB6H	DMA read/write pointer 1 (high)	חרוכו	DPTC1H	R/W		•	•	•	Undefined
DMA control register 1Software timer/counterSoftware timer/counter compare reProgrammable wait control registerProgrammable wait control registerMemory block control registerRefresh mode registerStandby control register	DFFFBCH	DMA mode register 1	DMAM1		RW	•	•			EOH
Software timer/counter Software timer/counter compare rei Programmable wait control register Programmable wait control register Memory block control register Refresh mode register Standby control register	DEFFBDH	DMA control register 1	DMAC1		Ϋ́	•	•			HOO
Software timer/counter compare re Programmable wait control register Programmable wait control register Memory block control register Refresh mode register Standby control register	1	Software timer/counter	STC		Я			•		Undefined
Programmable wait control register Programmable wait control register Memory block control register Refresh mode register Standby control register	1	Software timer/counter compare register	STMC		R/W			•		FFFH
┼╌┤╴┟╾┼╌┤			PWC0		RN	•	•			EAH
	1	Programmable wait control register 1	PWC1		RN	•	•			ААН
	1	Memory block control register	MBC		Å	•	•			FCH
	DFFFECH	Refresh mode register	RFM		RW	•	•			77H
	DFFFEEH	Standby control register	STBC		R/W *1	•	•			Undefined *2
OFFFEFH Processor control register		Processor control register	PRC		R/W	•	•			EEH

*1 The SFB bit of the standby control register can be set (1) by instruction, but cannot be cleared (0). (Only '1' can be written.) *2 After power-on reset: 00H, otherwise: no change

NEC

6427525 0063139 254 🔳

3.5.4 Vector Table Area

The 1K-byte area 00000H to 003FFH in the memory space holds 256 vectors (4 bytes used per vector) for the start addresses of interrupt routines initiated by interrupt requests, break instructions, etc.

In the initial state, vectors 0 to 47 are reserved as V55PI family dedicated on-chip peripheral and software interrupt vectors. For vectors 8 to 47, the vector address of hardware interrupts except NMI can be changed by means of bits V0 and V1 of the interrupt mode control register (IMC).

Vector 0	(00000H)	:	Divide error
Vector 1	(00004H)	:	Single step
, Vector 2	(00008H)	:	NMI instruction
Vector 3	(0000CH)	:	BRK 3 instruction
Vector 4	(00010H)	:	BRKV instruction
Vector 5	(00014H)	:	CHKIND instruction
Vector 6	(00018H)	:	Input/output instruction
Vector 7	(0001CH)	:	FPO instruction/exception trap
When V1 = V0	= 0 :		
Vector 8	(00020H)	:	INTWDT
Vector 9	(00024H)	:	INTPO
Vector 10	(00028H)	:	INTP1
Vector 11	(0002CH)	:	INTP2
Vector 12	(00030H)	:	INTP3
Vector 13	(00034H)	:	INTP4
Vector 14	(00038H)	:	INTP5
Vector 15	(0003CH)	:	System reserved
Vector 16	(00040H)	:	INTCM00
Vector 17	(00044H)	:	INTCM01
Vector 18	(00048H)	:	INTCM10
Vector 19	(0004CH)	:	INTCM11
Vector 20	(00050H)	:	INTCM21
Vector 21	(00054H)	:	INTCM31
Vector 22	(00058H)	:	INTD0 DMA#0_MAIN
Vector 23	(0005CH)	:	INTDOS DMA#0_SUB
Vector 24	(00060H)	:	INTD1 DMA#1_MAIN
Vector 25	(00064H)	:	INTD1S DMA#1_SUB
Vector 26	(00068H)	:	INTSERO
Vector 27	(0006CH)	:	INTSER1
Vector 28	(00070H)	:	INTSRO/INTCSI0
Vector 29	(00074H)	:	INTSR1/INTCSI1
Vector 30	(00078H)	:	INTST0
Vector 31	(0007CH)	:	INTST1
Vector 32	(00080H)	:	INTSIT
Vector 33	(00084H)	:	System reserved
Vector 34	(00088H)	:	System reserved
Vector 35	(0008CH)	:	System reserved
Vector 36	(00090H)	:	INTPAI
Vector 37	(00094H)	:	INTAD
Vector 38	(00098H)	:	System reserved
Vector 39	(0009CH)	:	System reserved

34

🖿 6427525 0063140 Т76 🎞

Vector 40	(000A0H)	: System reserved
Vector 41	(000A4H)	: System reserved
Vector 42	(000A8H)	: System reserved
Vector 43	(000ACH)	: System reserved
Vector 44	(000B0H)	: System reserved
Vector 45	(000B4H)	: System reserved
Vector 46	(000B8H)	: System reserved
Vector 47	(000BCH)	: System reserved
When V1 = 0,	V0 = 1 :	
Vector 72	(00120H)	: INTWDT
Vector 73	(00124H)	: INTPO
•	•	•
•	•	•
Vector 110	(001B8H)	: System reserved
Vector 111	(001BCH)	: System reserved
When V1 = 1,	V0 = 0 :	
Vector 136	(00220H)	: INTWDT
Vector 137	(00224H)	: INTPO
•	•	•
•	•	•
Vector 174	(002B8H)	: System reserved
Vector 175	(002BCH)	: System reserved
When V1 = 1,	V0 = 1 :	
Vector 200	(00320H)	: INTWDT
Vector 201	(00324H)	: INTPO
•	•	•
•	•	
Vector 238	(003B8H)	: System reserved
Vector 239	(003BCH)	: System reserved
		•

🔲 6427525 0063141 902 📖

3.6 REGISTER FILE SPACE

The register file space is shown in Figure 3-5.

The size of the register file space is 512 bytes, and a maximum 16-bank register set can be set.

The register file space is separate from the memory space, and in addition to accesses using a register manipulation instruction as with the V25 and V35, the register file space can be accessed as data memory by adding a special prefix instruction (IRAM:) to a memory manipulation instruction. (Access is performed asynchronously independently of the external bus cycle.

When the IRAM: prefix instruction is added to a memory manipulation instruction, the CPU performs a data access with the low-order 9 bits of the memory address offset value as the register file address. In this case, segment register and physical address addition is not performed, and an external bus cycle is not initiated.

Example

Label1: MOV IRAM : [0024H], AW(1) MOV [0056H], BW(2)

- (1) This shows the case where data is transferred to the register file space using an "IRAM:" prefix instruction. The AW register value is stored in address 24H of the register file.
- (2) This shows the case where an instruction for data transfer to the memory space is used.

If the IRAM prefix instruction is added to the primitive block transfer instruction and BCD operation instruction, which specify the source block and destination block, it becomes effective for the destination block.

Also, the macro service conrol word area (008H to 03FH), the macro service work area (000H to 007H), and the area used by the macro service channel (008H to 0FFH) are allocated in overlapping fashion in the file space. If a specific macro service which requires work area (RTOPTRN) is not used, these work areas can be used as data space.

6427525 0063142 849 📟

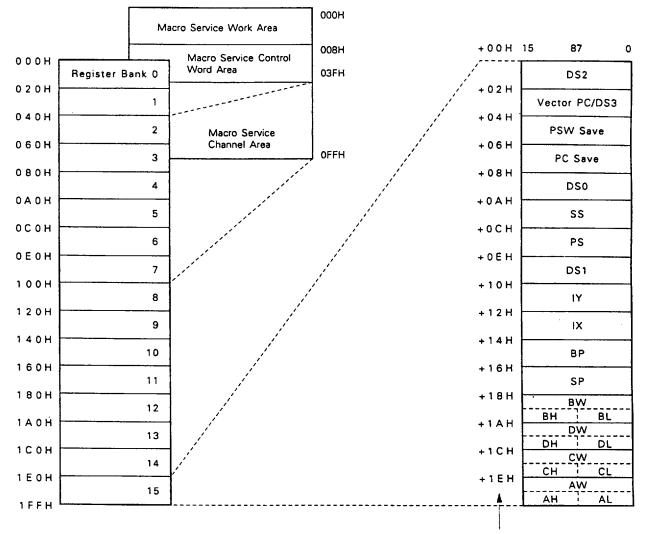


Figure 3-5. Register File Space

(Offset from the starting address of each register bank)

🖬 6427525 OO63143 785 📰

3.7 I/O SPACE

The V55PI has a 64K-byte I/O space.

The I/O space map is shown in Figure 3-6.

The I/O space is accessed using address bus/data bus and control signals (IORD, IOWR, etc).

0 is output from the unused high-order 8 bits of the address bus.

Wait cycles can be inserted in an I/O cycle by software and the READY pin.

The area FF80H to FFFFH of the I/O space is a reserved area, in which two V55PI on-chip peripheral DMA input/ output read/write pointers (IOP) are allocated. The address of IOP0 is FF94H, and the address of IOP1 is FFB4H.

When the CPU executes an input/output instruction with an IOP address as an operand, the DMA controller performs a read/write of data in the DMA controller transfer buffer, with the IOP contents as the address value, and increments (or decrements) the IOP value automatically in accordance with the contents of the DMA control register. Therefore, data written by the DMA controller can be referenced by an input/output instruction, and conversely, data written by an input/output instruction can be transferred by the DMA controller.

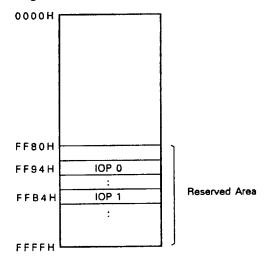


Figure 3-6. I/O Map (64K Bytes)

Remark IOPn corresponds to the DMA read/write pointer (DPTCn).



4. BUS CONTROL FUNCTIONS

With the V55PI pin, refer to 1.1.2 (1) "Pin function for bus control".

As regards pins which have an alternate function as port pins, when that function is used, the corresponding function must be selected by means of the port mode control register (PMCn).

4.1 WAIT FUNCTION

The V55PI divides the basic memory space (000000H to 0FFFFH) into a maximum of 4 blocks with a variable memory size, divides the uppermost extended memory space area (100000H to FFFFFH) into two areas with a variable memory size, and performs wait control for each block. The memory size of each block in the basic memory space is specified by the memory block control register (MBC).

Figure 4-1 shows the memory block configuration when A9H has been set for the MBC register value.

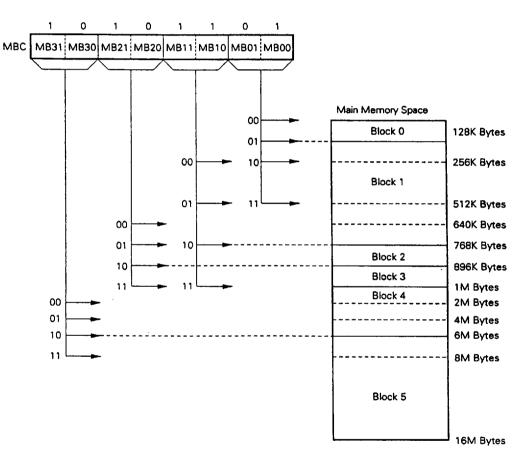


Figure 4-1. Partitioned Memory Control

🔳 6427525 0063145 558 🔳

	7	6	5	4	3	2	1	0
	(BLC	скз)	(BLC	CK2)	(BLC	оск1)	(BLC	СК0)
PWC1	DW31	DW30	DW21	DW20	DW11	DW10	DW01	DW00
	7	6	5	4	3	2	1	0
DALCO	(BLOCK4)	(BLOCK1)	(1/0 S	ipace)	(BLC	OCK5)	(BLC)CK4)
PWC0	AW1	AWO	IOW1	IOWO	DW51	DW50	DW41	DW40

Figure 4-2. Memory Wait Control

Data Wait (DW, IOW)

DWn1/IOW1	DWn0/IOW0	Wait State
0	0	0 *1
0	1	1 *2
1	0	2 *2
1	1	3 *2

* 1. READY signal is ignored.

2. Additional control by means of READY signal is also possible.

Address Wait (AW)

AW	n	Wait State
	0	Not inserted (block 1)
AW0	1	Inserted (block 1)
	0	Not inserted (block 4)
AW1	1	Inserted (block 4)

40

6427525 0063146 494 **m**

4.2 REFRESH FUNCTION

The following functions are provided to refresh DRAM and pseudo-SRAM.

- · Function to insert periodically a refresh cycle in a series of bus cycles
- Refresh address output function to refresh DRAM and pseudo-SRAM
- Function to generate a refresh cycle in hold mode and HALT mode.
- Function to insert a wait state in a refresh cycle

4.2.1 Refresh Mode Register (RFM)

The RFM register is an 8-bit register to control refresh operation.

A refresh cycle can be selected from the time base counter output tap.

While a refresh request is held by another bus cycle if the next refresh request is generated, only the latter is valid. The RFM register value after a reset is 77H.

4.2.2 Wait Control in Refresh Cycle

A wait state can be inserted in a refresh cycle. The specified number of wait states is inserted for memory block 4 by the programmable wait control register (PWC0) or READY pin.

4.2.3 Refresh Address

Bus pins AD0 to AD15 and A16 to A19 are activated in a refresh cycle.

For each refresh cycle, the count is performed in one-address increments from x00000 to x1FFFFF in the case of the external 8-bit bus width, and in two-address increments from x00001H to xFFFFF in the case of the external 16-bit bus width (the minimum address is returned to after the maximum address).

After initialization by a reset, count-up is started from x00000H in the case of the external 8-bit bus width and x00001H in the case of the external 16-bit bus width.

In the case of the external 16-bit bus width, the refresh address minimum address bit (A0) is fixed at "1" and the DEX pin output is also fixed at "1".

A20 to A23 are undefined in a refresh cycle.

🖬 6427525 006**3147 320**

5. INTERRUPT FUNCTIONS

The V55PI incorporates a powerful interrupt controller (INTC) which controls multiple-interrupt servicing for a total of 25 maskable hardware interrupt requests: 19 internal and 6 external. The interrupt controller controls multiple-interrupt servicing based on programmable priority.

The following functions are provided as interrupt servicing modes: vectored interrupt function, macro service function, register bank switching function.

5.1 FEATURES

V55PI interrupt functions offer the following features:

- Comprehensive servicing states for interrupt requests
 - Vectored interrupt function
 Branch to interrupt service routine specified by vector table
- Register bank switching function : High-speed interrupt response by automatic register bank switching
- Macro service function
 High-speed interrupt servicing by microprogram (firmware)
- 4-level programmable priority order control
- Interrupt multiprocessing control according to the priority
- Rich variety of macro service functions (following 7 modes) closely tied to V55PI on-chip peripheral hardware

EVTCNT BLKTRS	: Event count processing : Data transfer between special function register and external memory buffer
	: Data transfer between special function register and external memory buffer (with transfer data
	detection function)
DTACMP	: Special function register status detection
DTADIF	: Time measurement by timer capture function
RTOPTRN	: Automatic control of real-time output port
DTACMP-N	A : Data transfer between external I/O and memory

- 7 external interrupt request inputs (NMI, INTP0 to INTP5)
- Maskable interrupt requests are individually maskable.

A list of interrupt sources is given in Table 5-1.

42

🔳 6427525 006**3148** 267 🔳

(1/2)
Sources
Interrupt
Table 5-1.

$ \begin{array}{ c c c c c c c c c c c c c c c c c c c$			Interrupt		Interrupt Source		Default			Register	Macro Service
	Interrupt Classification	Priority	Request Signal	Interrupt Request Control Register	Generating Source	Generating Unit	Vector Table Number	Address	Service	Bank Switching	Control Word Address
atual z WDT B 00x20H No No No 3 INTPO IC9 INTPO pin input 9 00x24H Yes Yes Yes 4 INTP1 IC10 INTP1 pin input 10 00x28H Yes Yes Yes 5 INTP2 IC11 INTP2 pin input 11 00x28H Yes Yes Yes 6 INTP3 IC13 INTP4 pin input 12 00x28H Yes Yes Yes 7 INTP4 IC13 INTP4 pin input 12 00x28H Yes Yes Yes 8 INTP6 IC14 INTP6 pin input 12 00x28H Yes Yes Yes 9 INTCM01 IC13 INTP4 pin input 12 00x28H Yes Yes Yes 10 INTEM01 IC14 INTP6 Yes Yes Yes Yes Yes 11 INTCM01 IC15			IWN		NMI pin input		2	00008H	Ňo	No	
3 INTPO IC30 INTPO pin input. 9 00x24H Yes Yes Yes 4 INTP1 IC10 INTP1 pin input. 10 00x24H Yes Yes Yes 5 INTP2 IC11 INTP2 pin input. 11 00x24H Yes Yes Yes 7 INTP3 IC12 INTP4 pin input. 13 00x34H Yes Yes Yes 9 INTCM00 IC16 INTP6 pin input. 13 00x34H Yes Yes Yes 9 INTCM01 IC17 INTP6 pin input. 14 00x34H Yes Yes Yes 10 INTCM01 IC16 CM01 match detection 17 00x34H Yes Y	Nonmaskable	2	WDT	1	Watchdog timer overflow	WDT	8	00×20H	Na	No	I
4 INTP1 IC0 INTP1 pin input 10 0x2BH Yes Yes 5 INTP2 IC11 INTP2 pin input 11 0x2CH Yes Yes 7 INTP4 IC12 INTP4 pin input 12 0x3CH Yes Yes 8 INTP4 IC13 INTP4 pin input 13 0x3CH Yes Yes 9 INTFM0 IC16 INTP4 pin input 14 0x3CH Yes Yes 9 INTFM0 IC16 CM00 match detection 14 0x3CH Yes Yes 10 INTCM1 IC19 CM11 match detection 17 00x3H Yes Yes 11 INTCM1 IC19 CM11 match detection 17 00x4H Yes Yes 12 INTCM1 IC19 CM11 match detection 17 00x4H Yes Yes 13 INTCM1 IC19 CM11 match detection 17 00x4H Yes Yes		e	INTPO	1C9	INTPO pin input	4	σ	00×24H	Yes	Yes	012H
5 NHP2 IC11 NHP2 pin input External 11 00x2CH Yes Yes 7 NTP4 IC12 INTP4 pin input 12 00x3H Yes Yes Yes 8 NTP6 IC14 INTP5 pin input 13 00x3H Yes Yes Yes 9 NTCM00 IC16 CM00 match detection 14 00x3H Yes Yes Yes 10 NTCM01 IC17 CM01 match detection 17 00x4H Yes Yes Yes 11 NTCM1 IC19 CM1 match detection 17 17 00x4H Yes Yes Yes 12 INTCM1 IC19 CM1 match detection 17 18 00x4H Yes Yes Yes Yes Yes 13 INTCM1 IC19 CM1 match detection 17 19 00x4H Yes Yes Yes Yes Yes Yes Yes Yes Yes Yes </td <td></td> <td>*</td> <td>INTP1</td> <td>IC10</td> <td>INTP1 pin input</td> <td></td> <td>10</td> <td>00×28H</td> <td>Yes</td> <td>Yes</td> <td>014H</td>		*	INTP1	IC10	INTP1 pin input		10	00×28H	Yes	Yes	014H
6 INTP3 ICT2 INTP3 in input Extention 12 00x30H Yes Yes 7 INTP4 ICT3 INTP4 pininput 13 00x3H Yes Yes Yes 8 INTP5 ICT4 INTP5 pin input 14 00x3H Yes Yes 9 INTCM00 ICT6 CM00 match detection 17 00x4H Yes Yes 10 INTCM10 ICT8 CM10 match detection 17 00x4H Yes Yes 11 INTCM11 ICT9 CM10 match detection 17 00x4H Yes Yes 12 INTCM11 ICT9 CM11 match detection 19 00x4H Yes Yes 13 INTCM21 ICZ1 CM31 match detection 20 00x5H Yes Yes 14 INTCM21 ICZ2 CM31 match detection 21 00x5H Yes Yes 15 INTCM21 ICZ2 CM31 match detection 21		ъ	INTP2	IC11	INTP2 pin input		11	00×2CH	Yes	Yes	016H
7 INTP4 IC13 INTP4 pininput 13 00x3H1 Yes Yes 8 INTP5 IC14 INTP5 pin input 14 00x3H1 Yes Yes 9 INTCM00 IC16 CM00 match detection 16 00x4H1 Yes Yes 10 INTCM10 IC18 CM10 match detection 17 00x4H Yes Yes 11 INTCM11 IC19 CM10 match detection 18 00x4H Yes Yes 12 INTCM11 IC19 CM11 match detection 18 00x4H Yes Yes 13 INTCM21 IC20 CM21 match detection 20 00x5H Yes Yes 14 INTCM21 IC20 CM21 match detection 21 00x5H Yes Yes 15 INTCM21 IC21 CM31 match detection 22 00x5H Yes Yes 16 INTCM21 IC22 DMA channel 0_main 23 00x5H Yes		9	INTP3	IC12	INTP3 pin input	External	12	00×30H	Yes	Yes	018H
8 NTP5 IC14 INTP5 pin input 14 00x38H Yes Yes 9 INTCM00 IC16 CM00 match detection 15 00x40H Yes Yes 10 INTCM01 IC17 CM01 match detection 17 00x40H Yes Yes 11 INTCM10 IC18 CM10 match detection 18 00x40H Yes Yes 12 INTCM11 IC19 CM11 match detection 18 00x40H Yes Yes 13 INTCM21 IC20 CM11 match detection 19 00x40H Yes Yes 14 INTCM21 IC21 CM31 match detection 21 00x50H Yes Yes 15 INTO0 IC22 DMA channel 0_main 22 00x50H Yes Yes 16 INTD0 IC23 DMA channel 0_main 23 00x50H Yes Yes 17 INTD0 IC23 DMA channel 0_main 24 00x60H Yes </td <td></td> <td>7</td> <td>INTP4</td> <td>IC13</td> <td>INTP4 pininput</td> <td></td> <td>13</td> <td>00×34H</td> <td>Yes</td> <td>Yes</td> <td>01AH</td>		7	INTP4	IC13	INTP4 pininput		13	00×34H	Yes	Yes	01AH
9INTCM00IC16CM00 match detection1600×40HYesYesYes10INTCM01IC17CM01 match detection1700×44HYesYesYes11INTCM10IC18CM10 match detection1800×48HYesYesYes12INTCM11IC19CM11 match detection1900×46HYesYesYes13INTCM21IC20CM11 match detection2000×60HYesYesYes14INTCM31IC21CM31 match detection2100×60HYesYesYes15INTCM31IC22CM31 match detection2200×60HYesYesYes16INTD0IC22DMA channel 0_main2300×50HYesYesYes17INTD1IC24DMA channel 0_main2400×60HYesYesYes18INTD1IC25DMA channel 1_main2400×60HYesYesYes		8	INTPS	IC14	INTP5 pin input		14	00×38H	Yes	Yes	01CH
10INTCM01IC17CM01 match detection17 $00\times44H$ YesYesYes11INTCM10IC18CM10 match detection18 $00\times46H$ YesYesYes12INTCM11IC19CM11 match detection19 $00\times46H$ YesYesYes13INTCM21IC20CM11 match detection20 $00\times46H$ YesYesYes14INTCM31IC21CM31 match detection21 $00\times56H$ YesYesYes15INTD0IC22DMA channel 0_main22 $00\times56H$ YesYesYes16INTD0IC23DMA channel 0_subDMA23 $00\times66H$ YesYesYes17INTD1IC24DMA channel 0_subDMA24 $00\times60H$ YesYesYes18INTD1SIC25DMA channel 1_main25 $00\times60H$ YesYesYes		6	INTCM00	IC16	CM00 match detection		16	00×40H	Yes	Yes	020H
11INTCM10IC18CM10 match detection1800x4BHYesYes12INTCM11IC19CM11 match detection1900x4CHYesYesYes13INTCM21IC20CM21 match detection2000x50HYesYesYes14INTCM31IC21CM31 match detection2100x54HYesYesYes15INTD0IC22DMA channel 0_main2200x58HYesYesYes16INTD0SIC23DMA channel 0_subDMA2300x56HYesYesYes17INTD1IC24DMA channel 1_main2400x60HYesYesYesYes18INTD1SIC25DMA channel 1_sub2500x64HYesYesYesYes	-	9	INTCM01	IC17	CM01 match detection		17	00×44H	Yes	Yes	022H
	Maskable	1	INTCM 10	IC18	CM10 match detection	1 	18	00×48H	Yes	Yes	024H
INTCM21 IC20 CM21 match detection 20 00×50H Yes Yes INTCM31 IC21 CM31 match detection 21 00×50H Yes Yes Yes INTD0 IC22 DMA channel 0_main 22 00×50H Yes Yes Yes INTD0 IC23 DMA channel 0_main 23 00×50H Yes Yes Yes INTD0S IC23 DMA channel 0_sub DMA 23 00×50H Yes Yes Yes INTD1 IC24 DMA channel 1_main 24 00×60H Yes		12	INTCM11	IC19	CM11 match detection		19	00×4CH	Yes	Yes	026H
INTCM31 IC21 CM31 match detection 21 00x54H Yes Yes INTD0 IC22 DMA channel 0_main 22 00x58H Yes Yes Yes INTD0 IC23 DMA channel 0_sub 23 00x56H Yes Yes Yes INTD1 IC24 DMA channel 1_main 24 00x60H Yes Yes INTD1 IC25 DMA channel 1_main 25 00x60H Yes Yes Yes		13	INTCM21	IC20	CM21 match detection		20	00×50H	Yes	Yes	028H
INTD0 IC22 DMA channel 0_main 22 00×58H Yes Yes INTD0S IC23 DMA channel 0_sub DMA 23 00×5CH Yes Yes Yes Yes INTD1 IC24 DMA channel 1_main 24 00×60H Yes Yes Yes Yes Yes INTD1S IC25 DMA channel 1_sub 25 00×64H Yes		14	INTCM31	IC21	CM31 match detection		21	00×54H	Yes	Yes	02AH
INTDOS IC23 DMA channel 0_sub 23 00x5CH Yes Yes INTD1 IC24 DMA channel 1_main 24 00x60H Yes Yes INTD15 IC25 DMA channel 1_sub 25 00x64H Yes Yes		15	INTDO	IC22	DMA channel 0_main		22	00×58H	Yes	Yes	02CH
INTD1 IC24 DMA channel 1_main DMA INTD1S IC25 DMA channel 1_sub 25 00×60H Yes		16	INTDOS	IC23	DMA channel 0_sub		23	00×5CH	Yes	Yės	02EH
INTD1S IC25 DMA channel 1_sub 25 00×64H Yes Yes		17	INTD1	IC24	DMA channel 1_main	ζ₩Ο	24	H09×00	Yes	Yes	030H
		18	INTD1S	IC25	DMA channei 1_sub		25	00×64H	Yes	Yes	032H

🖬 6427525 0063149 1T3 🎟

ନ
(2/2
ŝ
e
Ċ,
3
•
c n
-
<u> </u>
- E
-
- =
–
_
-
-
٠.
Óت.
<u> </u>
Ā
- 72
<u> </u>
-

	4	Interrupt		Interrupt Sourse		Default Vertor	Vectored	Macro	Register	Macro Service
Classification	Priority		Interrupt Request Control Register	Generating Source	Generating Unit	Table Number	Address	Service	Bank Switching	Control Word Address
	19	INTSERO	IC26	UART reception error (ch0)		26	00×68H	No	Yes	034H
	20	INTSER1	IC27	UART reception error (ch1)		27	00×6CH	No	Yes	036H
		INTSR0/		UART reception (ch0)/	·	ę		Yes	Yes	
	51	INTCS10	IC28	Serial transmission/reception (ch0)		58		Yes	Yes	038H
		INTSR1/		UART reception (ch1)/		ç	002311	Yes	Yes	1464
Maskable	5	INTCS11	ICZ3	LDMA channel 5		ß		Yes	Yes	
	23	INTSTO	IC30	UART transmission (ch0)	•	30	00×78H	Yes	Yes	03CH
	24	INTST1	IC31	UART transmission (ch1)		31	00×7CH	Yes	Yes	03EH
	25	INTSIT	IC32	STM match detection	SIT	32	H08×00	No	Yes	l
	26	INTPAI	1C36	Parallel VF	Parallet I/F	36	H06×00	89 人	Yes	1
	27	INTAD	IC37	A/D converter	AD converter	37	00×94H	Yes	Yes	H800
				Divide error		0	H00000	No	No	
				BRK flag (single-step)		-	00004H	٥N	No	
				BRK3 instruction		ε	0000СН	٥N	No	
				BRKV instruction		4	00010H	No	No	
Software				CHKINĎ instruction		2	00014H	No	No	1
	I	i	1	Input/output instruction (IBRK flag)	1	9	00018H	No	No	•
				BRK imm8		*	H*×00	No	٥N	
				BRKCS instruction		1	1	No	Yes	 1
				FP0 instruction/						
Exception trap				Exception trap		7	0001CH	No	°N N	

Indicates that the value is variable in the range 0 to 255 (0 to FFH).
 Remarks "x" indicates that the value is determined by the V0 and V1 bits of the IMC register.

44

6427525 0063150 915 🔳

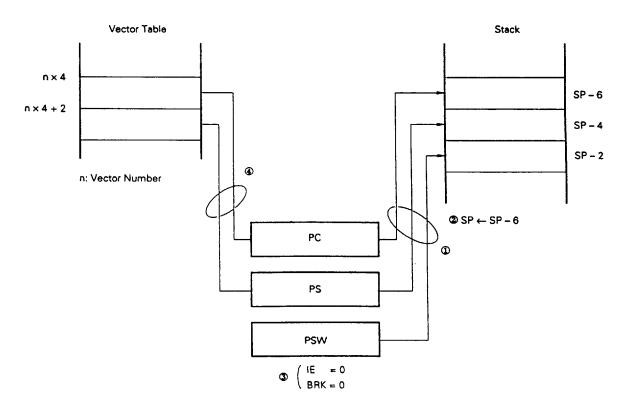
5.2 INTERRUPT RESPONSE METHODS

The V55PI has three interrupt response methods: a vectored interrupt function, register bank switching function, and macro service function. In the case of a maskable interrupt request, one of these functions can be selected by means of the interrupt request control register (ICxx) for each interrupt source according to the purpose of the interrupt. The on-chip interrupt controller handles interrupt requests according to the set response method.

5.2.1 Vectored Interrupts

A vectored interrupt can only be acknowledged in the interrupt enabled state (El state). When a vectored interrupt is acknowledged, the CPU enters the interrupt disabled state (DI state), and the current PSW contents and PC and PS contents are saved to the stack. Then the corresponding vector is selected from the vector table, and the interrupt service routine is started at the address indicated by that vector. Vector numbers are fixed for each interrupt source. In the DI state, interrupts are held pending, and are acknowledged when the El state is set again.

The return from the interrupt is performed by an RETI instruction. In the case of a hardware interrupt other than a non-maskable interrupt, an FINT instruction must be executed before the return instruction. When a return is made from an interrupt, the PC, PS and PSW are restored from the stack.





🖿 6427525 OO63151 851 💻

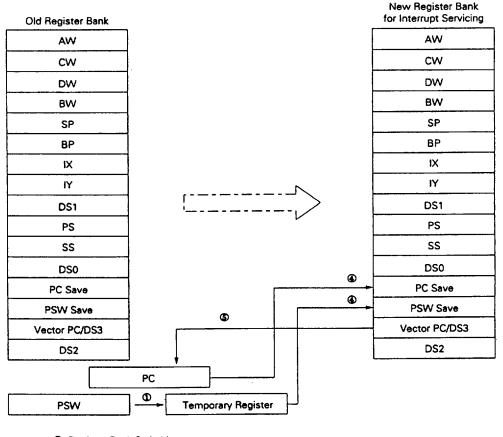
5.2.2 Register Bank Switching Function

In the V55PI, general register sets are mapped onto on-chip RAM, and register sets can be held in up to 16 banks. Interrupt servicing is performed by automatically switching the register bank when a BRKCS or TSKSW instruction is executed or when an interrupt is responded to. Because saving of registers to the stack previously performed by software is not required, high-speed switching of the program execution environment is possible.

The register bank switching sequence is performed as follows (See Figure 5-2).

- ① The contents of PSW is saved to temporary register.
- 2 The register bank is switched.
- ③ IE and BRK are set to 0.
- The contents of PSW which is saved to the PC and the temporary register are saved to the saving area, respectively.
- The interrupt service routine start address offset value is loaded from the vector PC area in the register bank to PC.

Figure 5-2. Register Bank Switching Sequence (In Case of Register Bank Switching by Interrupt)



Register Bank Switching

③ 1E = 0, BRK = 0

46

🖬 6427525 OO63152 798 🔜

5.2.3 Macro Service Function

The macro service function performs processing of simple data transfers, etc., by means of a microprogram (CPU internal dedicated firmware) started by generation of an interrupt request. The simple, standardized interrupt servicing which was coded and executed by a user program is performed automatically.

Macro service processing is caused by an interrupt request and is performed. Macro service is designed to minimize as far as possible the frequency of generation of interrupts consisting mainly of software processing, hold down the software overhead due to a series of processes used in an interrupt (register saving, initialization, register restoration, return from the interrupt routine), and improve the CPU efficiency.

Processing performed by the macro service is transparent in terms of software, and it is possible to process as a single mass of data what was previously processed by software byte by byte, allowing more efficient programming.

The V55PI macro service supports not only the simple data transfers used in the V25 and V35, but also various operating modes closely linked to the on-chip V55PI peripheral hardware, as shown below.

(a) EVTCNT (EVENT COUNTER)

The counter is updated each time the macro service are generated, and when the counter reaches 0 the macro service for the corresponding interrupt source is terminated and a vectored interrupt or a register bank switching is generated.

(b) DTACMP (DATA COMPARE)

The interrupt source specific SFR and preset byte data are compared, and if they match, the macro service for the corresponding interrupt source is terminated and a vectored interrupt or register bank switching is generated.

(c) DTADIF (DATA DIFFERENCE)

The difference in using the timer/counter unit capture register is calculated. This is initiated by a timer interrupt: the value of the capture register latched last time is subtracted from the value of the capture register latched this time, and the result is stored in the previously specified memory buffer.

When processing has been performed the previously set number of times, the corresponding interrupt source macro service is terminated, and a vectored interrupt or register bank switching is generated.

(d) BLKTRS (BLOCK TRANSFER)

A data transfer is performed between the previously specified memory buffer and SFR.

When the previously set number of data transfers have been performed, the corresponding interrupt source macro service is terminated, and a vectored interrupt or register bank switching is generated.

(e) BLKTRS-C (BLOCK TRANSFER WITH CHARACTER SEARCH)

A data transfer is performed between the previously specified memory buffer and SFR. When the previously set number of data transfers have been completed, or when the transfer data matches the previously set character data, the corresponding interrupt source macro service is terminated, and a vectored interrupt or register bank switching is generated.

(f) RTOPRTN (RTOP TRANSFER)

Data to be output to the real-time output port is transferred to the port 7 buffer (P7H, P7L), and data which specifies interval for output to the real-time output port is transferred to the timer compare register (CM00, CM01).

(g) DTACMP-M (DATA COMPARE WITH CHARACTER MASK)

The logical product of the status data read from the external I/O and the previously set mask data is performed. The previously set byte data is compared with the result. If it matches, a data transfer is performed between the external I/O and memory. If it does not match, or if the previously set number of data transfers have been performed, the corresponding interrupt source macro service is terminated, and a vectored interrupt or register bank switching is generated.

6427525 0063153 624 🔳

6. DMA FUNCTION (DMA CONTROLLER)

The V55PI incorporates a 2-channel DMA controller which controls execution of memory-to-I/O or memory-tomemory DMA transfers on the basis of DMA requests generated by an on-chip peripheral hardware (serial interface, parallel interface, or timer), the external DMARO pin or a software trigger.

Each channel of the DMA controller further comprises a main channel and a sub-channel: the operating mode determines whether the main channel and sub-channel are used as a single channel or as separate channels. When used as separate channels, function for a maximum of 4 channels can be constructed.

6.1 FEATURES

- Two independent DMA channels (max. 4-channel configuration possible)
- Four transfer modes
 - Single transfer mode ... One DMA transfer cycle is executed in response to one DMA request.
 - Demand release mode ... Consecutive DMA transfer cycles are executed while DMA request is active.
 - Single-step mode ... DMA transfer cycles and CPU bus cycles are executed alternately after DMA request generation.
 - Burst mode ... For each DMA request, the specified number of DMA transfer cycles are executed consecutively.
- Five operating modes
- ... DMA transfers to ring buffer are controlled. Intelligent DMA mode-1 (ring buffer system) · Intelligent DMA mode-2 (counter control system) ... Transfer data is transferred consecutively, divided into an arbitrary number of bytes. ... Consecutive transfers are possible between different Next address specification mode transfer buffers. ... Main channel and subchannel are used as independent · 2-channel operating mode channels. ... Two bus cycles are started for one DMA transfer cycle, Memory-to-memory transfer mode and memory-to-memory transfer is executed. 3 clocks/1 bus cycle (no wait case) Transfer objects ... 1 DMA transfer cycle/1 bus cycle External I/O ←→ memory SFR (internal I/O) ←→ memory ... 1 DMA transfer cycle/1 bus cycle Memory ←→ memory (memory includes SFR) ... 1 DMA transfer cycle/2 bus cycles
- Byte transfer/word transfer selectable
- Transfer address increment/decrement/non-update selectable
- DMA transfer end signal (TCE0, TCE1) output
- 24-bit DMA memory address registers (MAR0, MAR1)
- 21-bit terminal counters (TC0, TC1)
- External DMA request signal input pins (DMARQ0, DMARQ1: alternate function as port P80 and P81 pins)
- External DMA acknowledge signal output pins (DMAAK0, DMAAK1)

🖬 6427525 0063154 560 🛲

⁴⁸

T	C	MA Start Sourc	e			
Transfer Mode	On-Chip Peripheral	Software Trigger	DMARQ Pin	STOP Method	Interrupt	
Single transfer mode	Available	Available	Available	Reset of EDMA bit of DMAMn register	Acknowledged	
Demand release mode	Not Available	Not Available	Available	Stops when the DMARQ pin is driven low during the transfer. Reset of EDMA bit of DMAMn register	Not acknowledged during transfer. Acknowledged at other times.	
Single step mode	Available*	Available	Available	Reset of EDMA bit of DMAMn	Acknowledged register	
Burst mode	Available*	Available	Available	None (stop disabled during the transfer)	Not acknowledged	

Table 6-1. Transfer Modes

* The DMA start source is an on-chip timer interrupt, and transfer is possible only when the transfer I/O specification is external.

•			·····	Possible Tran	sfer Modes*	
Operating	Mode	Transfer Type	1	2	3	٩
Intelligent DMA mod (ring buffer method		I/O (SFR) → Memory	Yes	Yes	No	No
Intelligent DMA mo (counter control me		Memory \rightarrow I/O (SFR)	No	No	Yes	Yes
Next address specif	ication mode	I/O (SFR) ←→ Memory	Yes	Yes	No	No
2-channel	(Stop at end)	I/O ←→ Memory	Yes	Yes	Yes	Yes
operating-mode	(Repetition)	I/O ←→ Memory	Yes	Yes	Yes	No
Memory-memory	(Stop at end)	Memory ←→ Memory	Yes	No	Yes	Yes
transfer mode	(Repetition)	Memory ←→ Memory	Yes	No	Yes	No

* Transfer modes

- ① Single transfer mode
- 2 Demand release mode
- ③ Single step mode
- Burst mode

🗰 6427525 OO63155 4T? 🎟

7. SERIAL INTERFACE FUNCTIONS

The V55PI is equipped with a 2-channel serial interface unit (ch0, ch1). The two communication protocols supported by the V55PI are as follows:

(1) Asynchronous UART
 (2) Clocked CSI SBI: 2-wire serial bus interface
 IOE: I/O expansion 3-wire serial interface

7.1 FEATURES

- Two communication protocols supported
- Two serial channels
- Wake-up function
- On-chip dedicated baud rate generator
- DMA request generated by completion of transmission/reception (transmit/receive data DMA transfer is capable)

7.2 PROTOCOLS

The UART is an asynchronous serial interface which achieves data synchronization by means of start/stop bits, and is functionally enhanced UART functions compared with previous single-chip microcontroller.

The CSI (clocked serial interface) is a clocked serial interface which achieves synchronization by transmission/ reception of a clock. The CSI is a subset of the standard serial bus interface specification for NEC single-chip microcontrollers, and I²C functions are not supported. The wake-up release function is implemented by using macro service.

		Supporte	d Protocols	
Serial Interface Unit	Clocke	d (CSI)	Asynchronous	
	SBI	IOE	(UART)	
Channel 0	Yes	Yes	Yes	
Channel 1	No	Yes	Yes	

Table 7-1. Supported Protocols

The UART function or CSI function can be programmably selected for each channel. Protocol selection is performed by means of the protocol selection register (ASP).

50

7.3 UART

7.3.1 Features

- Transfer rate: 95 to 390 Kbps (with 12.5 MHz system clock \$\phi\$)
 - 123 to 500 Kbps (with 16 MHz system clock ø)
- Full-duplex operation capability
- On-chip dedicated (transmission and reception) baud rate generators
- Wake-up function
- Zero parity function
- Parity error detection
- Framing error detection
- Overrun error detection
- Three dedicated UART interrupt sources
 - UART receive error interrupts (INTSER0, INTSER1)
 - UART reception interrupts (INTSR0, INTSR1)
 - UART transmisstion interrupts (INTST0, INTST1)
- Macro service function
 - UART reception interrupts (INTSR0, INTSR1)
 - UART transmission interrupts (INTST0, INTST1)

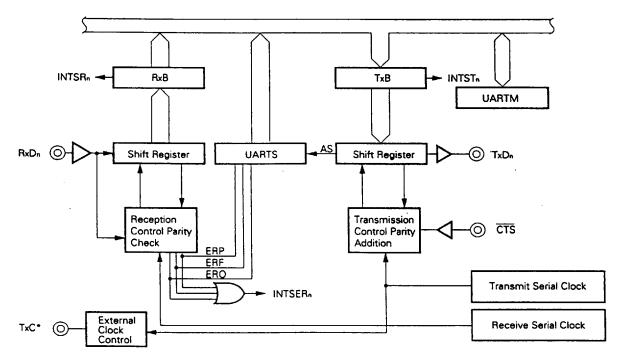


Figure 7–1. UART Block Diagram

* Channel 0 only

🖿 6427525 OO63157 27T 🛤

7.4 CLOCKED SERIAL INTERFACE (CSI)

7.4.1 Features

- Transfer speed: Max. 3.125 Mbps (with 12.5 MHz system clock φ) Max. 4.0 Mbps (with 16 MHz system clock φ)
- Half-duplex communication
- Data length: 8-bit unit
- External/internal clock selection function
- Data MSB-first/LSB-first selection function
- SBI mode (2-wire NEC type serial bus) ... ch0 only
 - Address/command/data identification function
 - · Function for chip selection by address
 - Wake-up function
 - Acknowledge signal (ACK) control function
 - Busy signal (BUSY) control function

The V55PI clocked serial interface has the following two operating modes.

(1) 3-wire serial I/O mode (IOE mode)

In this mode, 8-bit data transfer is performed using three lines: the serial clock (\overline{SCK}), and serial data input and output (SI, SO). This mode is useful when connecting an I/O device, display controller, etc., which incorporates a conventional clocked serial interface.

The functions of the V25 and V25+The have been enhanced, and data MSB-first/LSB-first selection is possible.

(2) Serial bus interface mode (SBI mode)

In the SBI mode, communication is performed with multiple devices by means of two lines: the serial clock (SCK) and the serial bus interface (SB0 or SB1).

This mode conforms to the NEC serial bus format.

In the SBI mode, the sender can output to the serial data bus an address to select the target device for serial communication, a command which gives a directive to the target device, and actual data. Thus there is no need for the line for handshaking required when multiple devices are connected with a conventional clocked serial interface, allowing input/output ports to be used efficiently.

In addition, wake-up release is performed using macro service.

52

6427525 OO63158 lO6

8. PARALLEL INTERFACE FUNCTIONS

The V55Pl incorporates a parallel interface unit for data input on a Centronics specification interface, and general data input/output.

8.1 FEATURES

The following features are provided as parallel interface functions:

- · Centronics specification interface compatibility
- Input/output mode switchable by software
- BUSY signal manipulable by software
- BUSY signal and ACK signal output timing settable
- Initialization by external interrupt
- Dedicated parallel interface interrupt source
- Parallel interface interrupt (INTPAI)
- DMA request signal generation in parallel transmission/reception
- INTPAI functions as a DMA start trigger.
- Signal pin input/output characteristic is TTL level (Centronics specification interface)

🖿 6427525 OO63159 O42 페

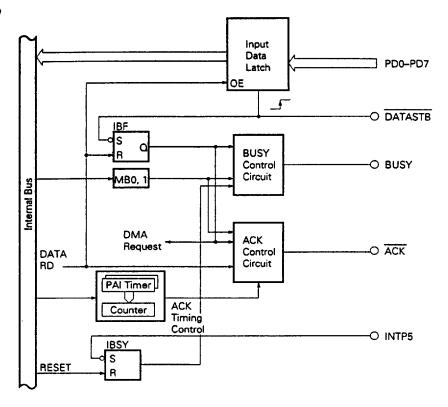
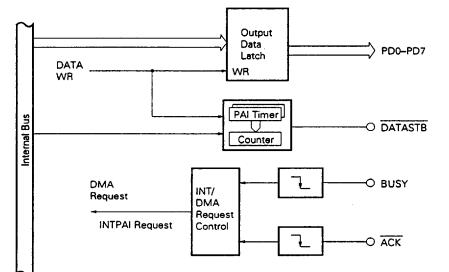


Figure 8-1. Parallel Interface Block Diagram

(a) Input mode

(b) Output mode



54

🔳 6427525 0063160 864 🔳

9. TIMER FUNCTION

The V55PI timer unit can be used as an interval timer, free-running timer and event counter. It is also possible to manipulate P7 as a real-time output port, synchronized with interrupt requests generated by the timer. The normal timer function and real-time output port function are described here.

9.1 FEATURES

The timer function offers the following features.

- 16-bit timer x 4
- Two count clock sources are selectable
 - System clock scaled output selectable (φ/8, φ/32: system clock φ)
 - External input pulses from TI pin
- External count output signal (TOn output)
- Three 16-bit capture registers on chip (external interrupt input signals INTPO to INTP2 as triggers)
- Six dedicated timer unit interrupt source (INTCM00, INTCM01, INTCM10, INTCM11, INTCM21, INTCM31)
- · Real-time output port function synchronized with timer interrupts

9.2 TIMER UNIT CONFIGURATION

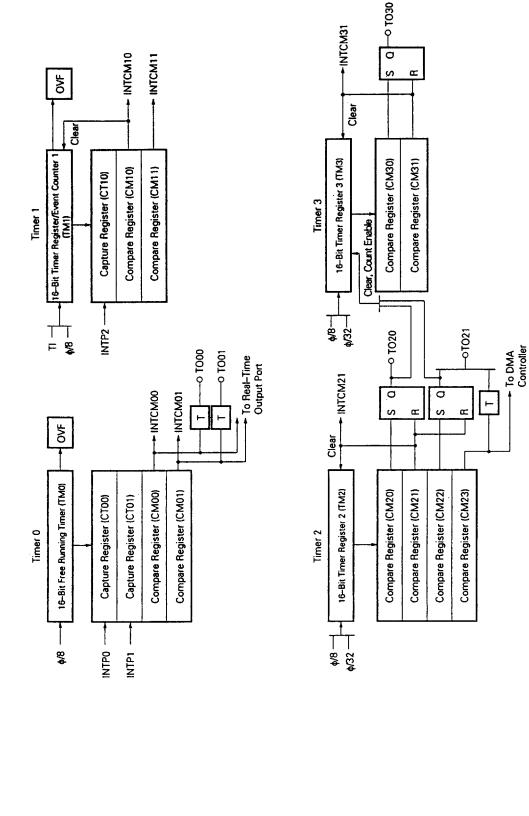
The timer unit configuration is shown in Figure 9-1, and the function of each timer in Table 9-1.

Table 9-1. Timer Functions.

			Timer 0	Timer 1	Timer 2	Timer 3
Function	Count function		Available	Available	Available	Available
	Capture function		Available	Available	Not Available	Nót Available
	Compare function		Available	Available	Available	Available
	Timer output function	Toggle output	Available	Not Available	Available	Not Available
		Set/reset output	Not Available	Not Available	Available	Available
	Cascading		Not Available	Not Available	Available	

🎫 6427525 0063161 7TO 🛤

56



6427525 0063162 637 💻

9.3 REAL-TIME OUTPUT PORT FUNCTION

Port 7 of the V55Pl incorporates a real-time output port function, and can output the contents of the port 7 buffer (P7H, P7L) at programmable intervals from timer 0 bit-wise.

9.3.1 Real-Time Output Port Configuration

The real-time output port configuration is shown in Figure 9-2. It comprises the following buffer registers, output and control registers.

(1) Port 7 buffer (P7H, P7L)

The buffer registers hold the data to be output next when port 7 is set to the real-time output port mode. The port 7 buffer contents are not affected by reset input.

(2) Real-time output port (RTP)

Real-time output port output data is held in this port after being taken from the port 7 buffer, and output from the pins.

RTP can be read or written to by an 8-bit or single-bit manipulation instruction (unlike the port 7 output port).

(3) Real-time output port delay specification regiser (RTPD) and delay counter

This register is set and used when using the mode in which a delay time is inserted in the timing for output from the real-time output port (RTP) to the output pins.

If the P7L bit is set to "0", "0" is output to the corresponding output pin bit after the elapse of the delay time equivalent to the count clock cycle time set in the real-time output port delay specification register after the time at which the transfer trigger is generated. The delay time in this case is counted by the delay counter.

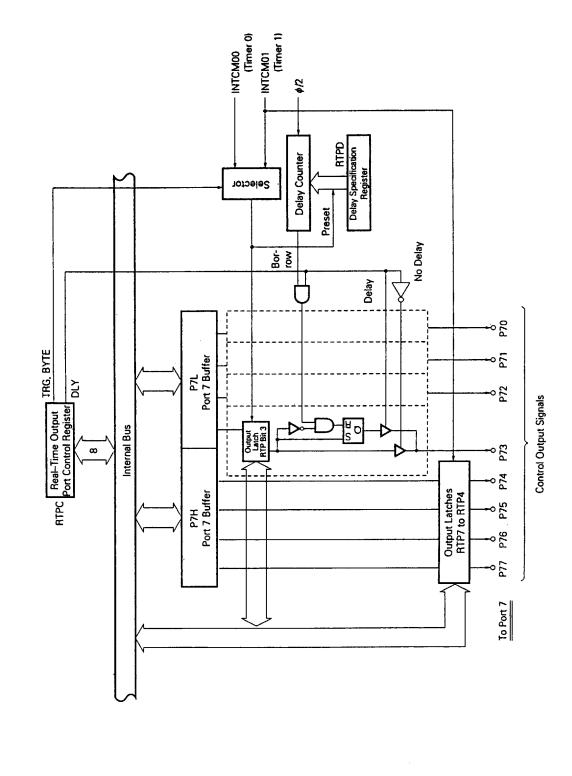
(4) Real-time output port control register (RTPC)

RTPC specifies the operating mode of the real-time output port. It is possible to specify whether or not a delay is to be inserted when data is output, the timing for transferring data to the port 7 buffer, the transfer timing trigger, and so on.



Figure 9-2. Real-Time Output Port Operation

58



🖬 6427525 0063164 40T 🖿

9.3.2 Real-Time Output Port Operation

Real-time output port specification is performed bit-wise by the port 7 mode control register (PCM7). Port 7 (P7), the port 7 buffer (P7H, P7L) and the real-time output port can be accessed as real-time output ports. Data output is performed as described below.

When output data is written in the port 7 buffer (P7H, P7L), the port 7 buffer contents are transferred to the realtime output port (RTP) and output to the pins in synchronization with the timing of an interrupt request from timer 0 (INTCM00, INTCM01), or a write to the TRG bit in the control register (RTPC).

An example of the direct control of the output pattern for a real-time output port and the output interval is shown in Figure 9-3.

Update data is transferred from the two data storage areas set beforehand in the external memory space to the real-time output function buffer registers (P7H, P7L) and compare registers (CM00, CM01).

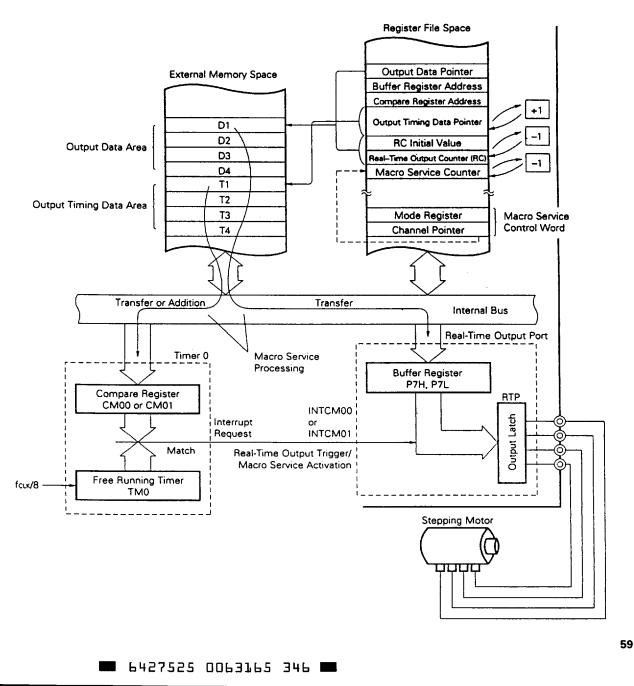


Figure 9-3. Real-Time Output Port Stepping Motor Control

In particular, it is possible to insert a delay time in the timing for output by setting the real-time output port delay specification register (RTPD) pins. If the P7L bit is changed from "1" to "0", it is possible to perform output after inserting a delay time of 2 × the system clock set in the RTPD from the timing at which the transfer trigger is generated. In this case, "0" is output from the corresponding output pin. This delay is counted by the delay counter.

60

10. PWM UNIT

The V55PI is provided with an 8-bit precision PWM (pulse width modulation) signal output function.

PWM output can be used as a digital-to-analog conversion output by connecting a low-pass filter, etc., externally. This is ideal for the actuator control signal for motors, etc.

10.1 FEATURES

The PWM unit offers the following features:

- PWM output pulse active level selectable
- Frequency: 25 MHz (with 12.5 MHz system clock φ)
 - \rightarrow PWM cycle: 40.96 μ s

: 32 MHz (with 16 MHz system clock ϕ)

- \rightarrow PWM cycle: 32.00 μ s
- Output pulse width (duty): 0, 1/256,, 255/256
 - → Resolution: 160 ns (with 12.5 MHz system clock ϕ) 125 ns (with 16 MHz system clock ϕ)

10.2 PWM UNIT CONFIGURATION

The configuration of the PWM unit is shown in Figure 10-1.

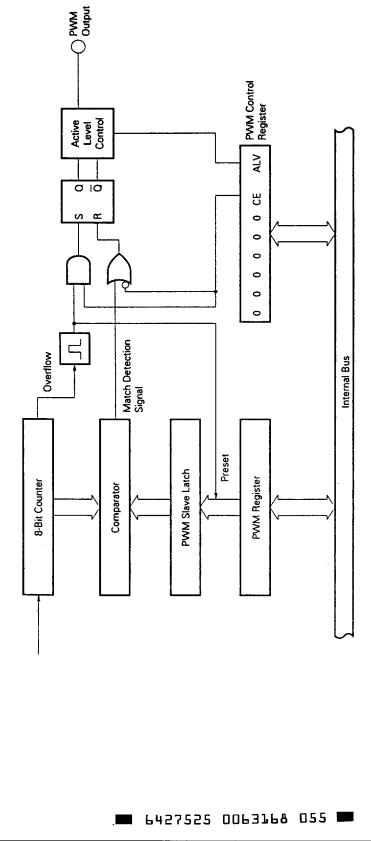
The PWM unit consists of the PWM register (PWM) and PWM control register (PWMC), and an 8-bit counter. The PWM register controls the pulse width (duty) in the PWM output mode. The 8-bit counter is set to 00H by

reset input. The PWM register is not affected by reset input.





62



11. WATCHDOG TIMER FUNCTION

The watchdog timer is a function for preventing inadvertent program looping and deadlocks.

11.1 FEATURES

- Three overflow times settable (8.1, 32.7, 131.0 [ms]: system clock φ = 16 MHz) (10.4, 41.9, 167.7 [ms]: system clock φ = 12.5 MHz)
- Output pin provided (WDTOUT pin) which can be directly linked to the RESET pin

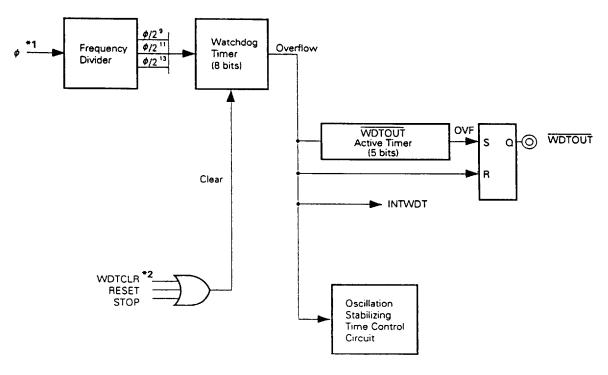
11.2 WATCHDOG TIMER CONFIGURATION AND OPERATION

Non-generation of a watchdog timer interrupt enables normal operation of the program or system to be confirmed. To use the watchdog function, an instruction (RSTWDT) to clear the watchdog timer (start the count) must be included in at fixed intervals in the program execution time, at the start of a subroutine, etc.

If the instruction which clears the watchdog timer is not executed within the set time and the watchdog timer overflows, a watchdog timer interrupt (INTWDT) is generated and the low-level signal is output to the WDTOUT pin to report a program error.

The watchdog timer configuration is shown in Figure 11-1.





* 1. ¢: System clock

2. WDTCLR: Watchdog timer clearance by instruction

💻 6427525 0063169 T91 📖

12. A/D CONVERTER FUNCTION

The V55PI incorporates a high-speed, high-precision 8-bit analog/digital (A/D) converter with four analog inputs (ANI0 to ANI3). The A/D converter uses the successive approximation method, and is provided with four A/D conversion result registers (ADCR0 to ADCR3) which hold the conversion results.

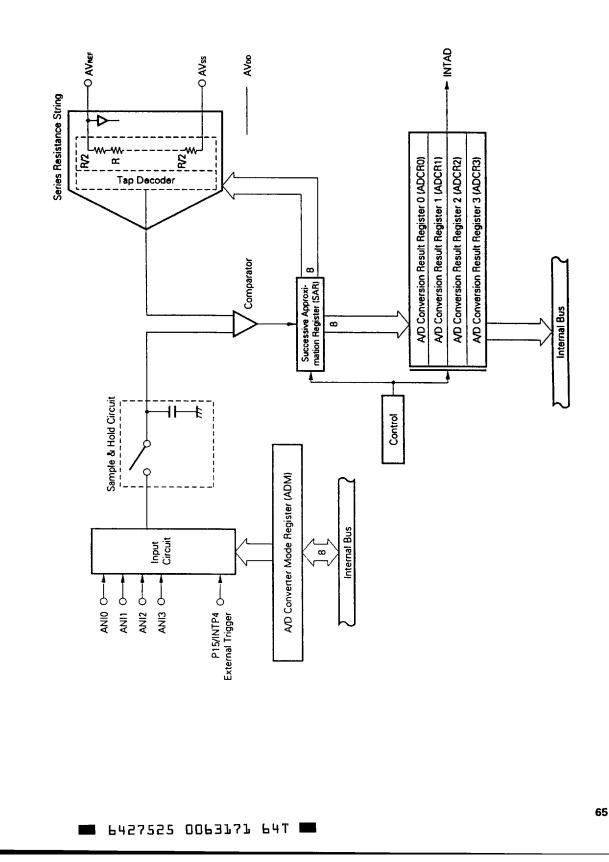
12.1 FEATURES

64

The A/D converter offers the following features:

- Incorporates four 8-bit A/D conversion result registers.
- Four analog input pins (ANI0 to ANI3)
- Two A/D converter conversion operating modes
 - Scan mode : Performs conversion by selecting multiple analog inputs in sequence.
 - Select mode: Performs continuous conversion with only one pin used as the analog input.
- Two conversion start methods
 - Hardware start : Started by trigger input (INTP4)
 - · Software start : Started by A/D converter mode register (ADM) bit setting
- Generation of conversion end interrupt request (INTAD)

Figure 12-1. A/D Converter Block Diagram



13. STANDBY FUNCTIONS

The V55PI has two methods for controlling the operating clock as standby functions designed to reduce power dissipation. Transition to either of these standby modes is possible by means of a dedicated instruction.

Parameter		HALT Mode	STOP Mode		
Clock generator		Operating			
Internal system clock		Stopped			
16-bit timer					
Watchdog timer			Stopped		
Hold circuit					
Serial interface					
Parallel interface		Operating			
A/D Converter					
Interrupt request controller					
DMA controller					
l/O line		High level	High level		
Bus lines	AD0 to AD15	Change according to DMAC operating	Retained		
bus imes	A16 to A23	status			
R/W output		High level	High level		
Refresh operation		Operating	Stopped		
Data retention		All internal data retained (CPU status, RAM contents, etc.)	All internal data retained (CPU status, RAM contents, etc.)		
Release method		• NMI • INTWDT • <u>Maska</u> ble interrupt request • RESET input	• <u>NMI</u> • RESET input		

Table 13-1. HALT/STOP Mode Operating Status

13.1 HALT MODE

In this mode, the CPU operating clock is halted.

Setting the CPU idle time to the HALT mode enables overall system power dissipation to be reduced. The HALT mode is entered by executing the HALT instruction.

In the HALT mode the CPU clock and program execution are stopped, and all register and on-chip RAM contents immediately prior to the stoppage are retained. The status of each hardware unit is shown in Table 13-1.

When the HALT instruction is executed during a DMA transfer, transition to the HALT mode is deferred until the transfer bus cycle for one DMA request is completed.

66

6427525 0063172 586 📖

13.2 STOP MODE

In this mode, clock oscillation is stopped.

This is effective when the entire application system is stopped, and offers extremely low power dissipation. The STOP mode is entered by executing the STOP instruction. In this mode all clocks are stopped. Program execution is stopped, and all register and on-chip RAM contents immediately prior to the stoppage are retained. The status of each hardware unit is shown in Table 13-1.

When the STOP instruction is executed during a DMA transfer, transition to the STOP mode is deferred until the transfer bus cycle for one DMA request is completed. If there is contention between a refresh cycle and STOP instruction execution, transition to the STOP mode is deferred until the refresh cycle is completed.

🔲 6427525 OO63173 412 🖿

14. CLOCK GENERATOR

The clock generator supplies various clocks to the CPU and peripheral hardware, and controls the CPU operating mode.

14.1 CLOCK GENERATOR CONFIGURATION AND OPERATION

The clock generator is configured as shown in Figure 14-1.

The clock generator clock is generated by a crystal resonator or ceramic resonator connected to the X1 and X2 pins. The clock generator output is subjected to waveform shaping (dividing frequency by 2) and selection of the scaling factor by means of the processor control register (PRC), and is then used as the system clock ϕ .

The system clock ø scaling factor is specified by the PCK1 and PCK0 bits of the PRC register, and can be selected as 1/2, 1/4, 1/8 or 1/16 the oscillator frequency (fxx).

Selecting a low-speed system clock ø reduces the current consumption of internal circuit, allowing extended operation of a battery-driven system even when the voltage drops.

An external clock can be input. In this case, the clock signal should be input to the X1 pin, and leave the X2 pin open.

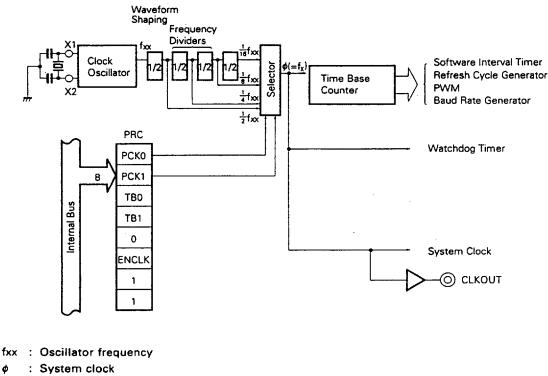


Figure 14-1. Clock Generator

PRC: Processor control register

68

■ 6427525 0063174 359 **■**

In the V55PI, the frequency divider (time base counter: TBC) which divides the internal system clock ϕ is shared by each timer unit.

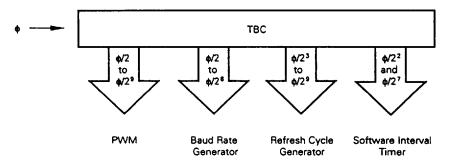
The TBC cannot be read or written to by an instruction.

The TBC tap output (divide-by-2ⁿ clock) is supplied to the units shown below as a count clock.

- (1) Refresh cycle generator
- (2) Software interval timer
- (3) PWM unit
- (4) Baud rate generator

The TBC is cleared to 00H only by reset input, after which it is constantly incremented. TBC operation is stopped in the STOP mode. The configuration of the TBC is shown in Figure 14-2.





➡ 6427525 0063175 295 ➡

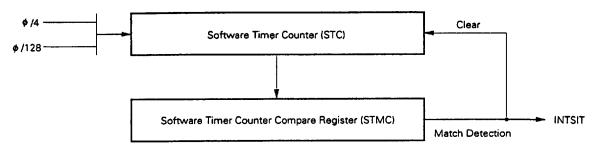
15. SOFTWARE INTERVAL TIMER FUNCTION

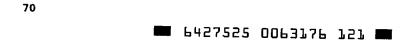
The V55PI incorporates a 16-bit software interval timer as a timer for software timer functions and watch functions.

15.1 SOFTWARE INTERVAL TIMER CONFIGURATION

The configuration of the software interval timer is shown in Figure 15-1.







16. CODEC INSTRUCTIONS

The V55PI has 9 codec instructions.

Using these special instructions on the V55PI enables not only image information MH encoding but also MR encoding which previously required the use of a special device such as an ACEE (advanced compression/expansion engine) to be implemented by means of a small-scale, high-speed codec.

16.1 FEATURES

The V55PI has the following 9 codec instructions (4 for compression, 5 for expansion):

- Compression instructions
- (1) Change point table creation instruction: COLTRP
- (2) Data transmission instruction (transmission of EOL *1, FILL, RTC *2, etc.): ALBIT
- (3) MH encoding instruction: MHENC
- (4) MR encoding instruction: MRENC
- Expansion instructions
- (5) EOL detection instruction: SCHEOL
- (6) 1-bit (tag) detection instruction: GETBIT
- (7) MH decoding change point table creation instruction: MHDEC
- (8) MR decoding change point table creation instruction: MRDEC
- (9) Pixel data creation instruction: CNVTRP

MH/MR encoding and MH/MR decoding using these instructions are performed as shown in Figures 16-1 and 16-2.

- 1. EOL: End Of Line
 - 2. RTC: Return To Control
- Note When compression/expansion processing is performed using the V55PI codec instructions, the following should be specified as preconditions.
 - · Compression/expansion is to be performed line by line.
 - · Consideration must be given to task switching and interrupt generation during compression processing.
 - The number of bits processed per line must not be changed during processing of one page.
 - The segment value must be changed for data over 64 Kbytes that straddles segments during processing.

💻 6427525 0063177 068 📰

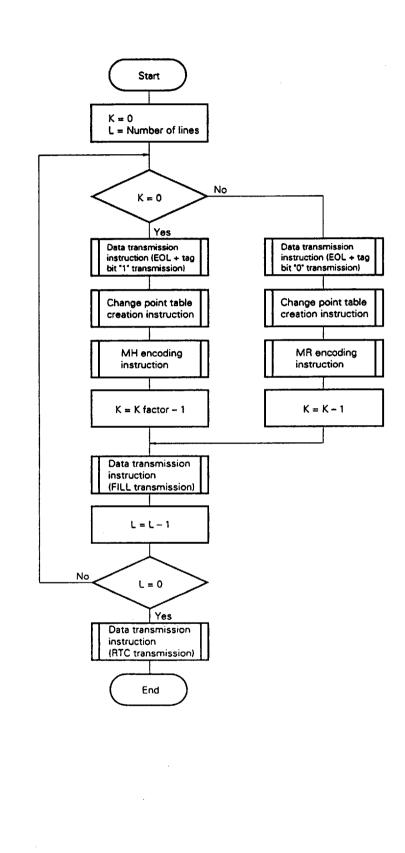


Figure 16-1. MH/MR Encoding Processing Flow

💻 6427525 0063178 TT4 🖿

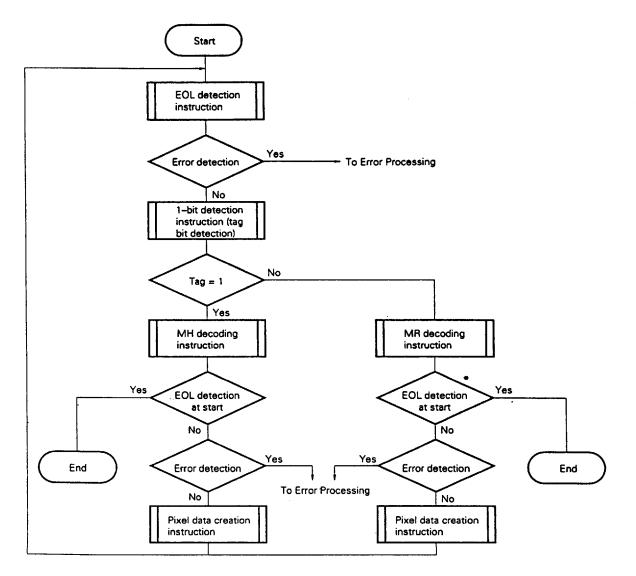


Figure 16-2. MH and MR Decoding Processing Flow

* RTC is detected by two EOLs.

16.2 MEMORY MAP

The data memory areas required by the V55PI's codec instructions are shown below.

(1) Register file space

This is the register bank for parameter setting.

(2) User RAM

Encoding line change point table	Storage area for change point information required for performing enco In the case of n bit/lines, a maximum area of 2n + 4 bytes is require	
Reference line table	Reference line change point information storage area	
Image data buffer	Storage area for pixel data read from scanner in encoding, or encoding received from modem in decoding) data
Transmit/receive buffer	Buffer for transferring encoded data to modem/scanner	
Print buffer	Buffer for transferring decoded pixel data to recording system	

(3) User ROM

Encoding conversion table	Conversion table for MH/N	AR encoding
Decoding conversion table	Conversion table for MH/N	AR decoding

(4) Access to Expanded Memory Space

The 16-Mbyte expanded memory space can be accessed by using the expanded segment override prefix instruction (DS2: or DS3:).

However, the segment registers DS2 and DS3 that are used during instruction execution are DS2 and DS3 in the parameter setting register banks of each instruction.

Table 16-1. Instructions to which Expanded Segment Override Prefix Can Be Attached

DS2:	DS3:	CODEC Instruction
Yes	Yes	COLTRP
Yes	No	MHENC
Yes	Yes	MHDEC
Yes	No	MRENC
Yes	Yes	MRDEC
Yes	No	SCHEOL
Yes	No	GETBIT
Yes	Yes	CNVTRP

Example

DS2 : DS3 : COLTRP DS2 : SCHEOL

The relationship between encoding instructions and data in memory is shown in Figure 16-3, and the relationship between decoding instruction and data in memory is shown in Figure 16-4.

🖬 6427525 0063180 652 페

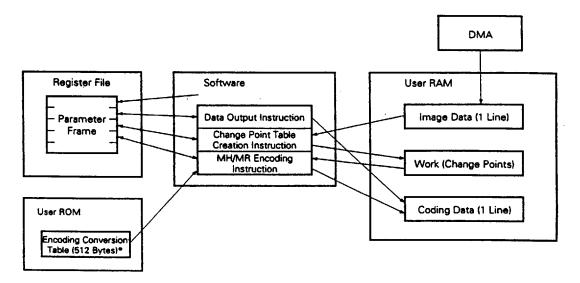
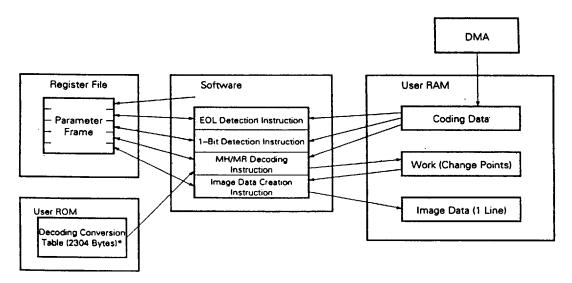


Figure 16-3. Encoding Instructions and Data in Memory

* In case of MH/MR encoding instructions





In case of MH/MR decoding instructions

🖿 6427525 OO63181 599 🛲

16.3 PROCESSING FLOW

The instructions shown in 16.1 "FEATURES" are used in the order shown in Figures 16-5 and 16-6 in encoding/ decoding processing.

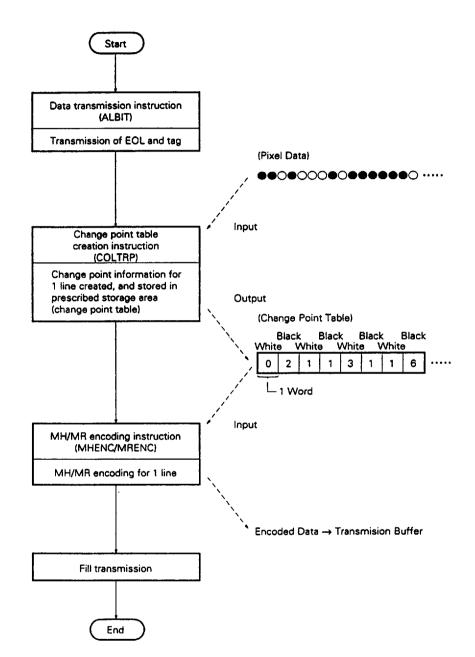
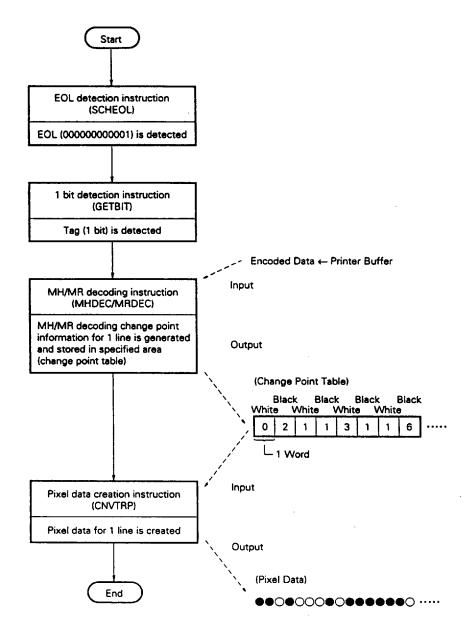


Figure 16-5. Processing Flow for Encoding of One Line

76

📕 6427525 0063182 425 📕





📰 6427525 OO63183 361 📟

17. INSTRUCTION SET

The V55PI instruction set is upward compatible with the V20/V30 (native mode) and V25/V35 instruction sets.

17.1 INSTRUCTIONS NEWLY ADDED TO V20/V30 AND V25/V35

Instructions which have been added to the V20/V30 and V25/V35 instruction sets, and instructions whose application range has been extended, are shown below.

(1) instructions added to V20/V30.

Mnemonic	Operand	Instruction Group	
BRKCS	reg 16		
TSKSW	reg 16		
MOVSPA	None	Data transfer instruction	
MOVSPB	reg 16		
BTCLR	sfr, imm3, short-label	Conditional branch instruction	
RETRBI	None	Interrupt instruction	
FINT	None		
STOP	None	CPU control instruction	

6427525 OO63184 2T8

(2) Instructions added to V25/V35.

Mnemonic	Operand	Instruction Group	
IRAM	None	Register file space access override prefix instrution	
DS2	None	Extended segment override prefix instruction	
DS3	None		
	DS2, reg16, mem32		
	DS3, reg16, mem32		
MOV	xsreg, reg16		
MOV	xsreg, mem16	Data transfer instruction	
	reg16, xsreg		
	mem16, xsreg		
	DS2		
PUSH	DS3/VPC		
	DS2	Stack manipulation instruction	
POP	DS3/VPC		
RSTWDT	imm8, imm8'	Watchdog timer manipulation instruction	
BTCLRL	sfri, imm3, short-label	Conditional branch instruction	
	reg8		
	mem8		
BSCH	reg16	 Bit manipulation instruction 	
_	mem16		
QHOUT	imm16		
QOUT	imm16	Queue manipulation instruction	
QTIN	imm16		
ALBIT	None		
COLTRP	None		
MHENC,	None		
MRENC	None		
SCHEOL	None	Dedicated FAX instruction	
GETBIT	None		
MHDEC	None		
MRDEC	None	1	
CNVTRP	None	1	

Remark VPC: Vector PC

17.2 INSTRUCTION SET OPERATIONS

Table 17	-1. 0	perand 1	Туре	Legend
----------	-------	----------	------	--------

Identifier	Description
reg,	8/16-bit general register
	(Destination register in an instruction using two 8/16-bit general registers)
reg'	Source register in an instruction using two 8/16-bit general registers
reg8,	8-bit general register
	(Destination register in an instruction using two 8-bit general registers)
reg8'	Source register in an instruction using two 8-bit general registers
reg16,	16-bit general register
	(Destination register in an instruction using two 16-bit general registers)
reg16'	Source register in an instruction using two 16-bit general registers
mem	8/16-bit memory address
mem8	8-bit memory address
mem16	16-bit memory address
mem32	32-bit memory address
sfr	Special function register location: FFF00H to FFFEFH
sfri	Special function register location: FFE00H to FFEFFH
dmem	16-bit direct memory address
imm	8/16-bit immediate data
imm3	3-bit immediate data
imm4	4-bit immediate data
imm8	8-bit immediate data
imm8'	8-bit immediate date (1's compliment of imm8)
imm16	16-bit immediate data
acc	Accumulator AW or AL
sreg	Segment register
xsreg	Extended segment register
src-table	Name of 256-byte conversion table
src-block	Name of source block addressed by register IX
dst-block	Name of destination block addressed by register IY
src-string	Name of source string addressed by register IX
dst-string	Name of destination string addressed by register IY
near-proc	Procedure start address in current program segment
far-proc	Procedure start address in a different program segment
near-label	Absolute address in current program segment
short-label	Relative address of memory in range -128 to +127 bytes from end of instruction
far-label	Absolute address in a different program segment
regptr16	16-bit general register holding call address offset in current program segment
memptr16	16-bit memory address holding call address offset in current program segment
memptr32	32-bit memory address holding call address offset and segment data in a different program segment
pop-value	Number of bytes removed from stack (0 to 64K, normally an even number)
fp-op	Immediate value which identifies external floating point operation coprocessor operation code
repeat	Repeat prefix instruction
IRAM :	Register file space access override prefix instruction
RAM :	Register set (AW, BW, CW, DW, SP, BP, IX, IY)
n ()	Omissible
or. /	Or
01,7	

■ 6427525 0063186 070 ■

Identifier	Description
w	Word/byte specification bit (1: word, 0: byte). However, when s = 1, sign extension byte data is
	specified as 16-bit operand even if W = 1.
reg, reg'	8/16-bit general register specification bits (000 to 111)
mod, mem	Memory addressing specification bits (mod: 00 to 10, mem: 000 to 111)
(disp-low)	Optional 16-bit displacement low byte
(disp-high)	Optional 16-bit displacement high byte
disp-low	16-bit displacement low byte for PC relative addition
disp-high	16-bit displacement high byte for PC relative addition
imm3	3-bit immediate data
imm4	4-bit immediate data
imm8	8-bit Immediate data
imm8'	8-bit immediate data (1's complement of imm8)
imm16-low	16-bit immediate data low byte
imm16-high	16-bit immediate data high byte
addr-low	16-bit direct address low byte
addr-high	16-bit direct address high byte
sreg	Segment register specification bits (00 to 11)
xsreg	Extended segment register specification bits (10 to 11)
8	Sign extension specification bit (1: sign extension, 0: no sign extension)
offset-low	Low byte of 16-bit offset data to be loaded in PC
offset-high	High byte of 16-bit offset data to be loaded in PC
seg-low	Low byte of 16-bit segment data to be loaded in PS
seg-high	High byte of 16-bit segment data to be loaded in PS
pop-value-low	Low byte of 16-bit data which specifies number of bytes to be removed from stack
pop-value-high	High byte of 16-bit data which specifies number of bytes to be removed from stack
disp8	8-bit displacement for relative addition to PC
х	
XXX	
YYY	Operation code of an external floating point operation coprocessor
ZZZ	

Table 17-2. Operation Code Legend

🔲 6427525 0063187 TO7 🖿

Identifier	Description
AW	Accumulator (16 bits)
AH	Accumulator (high byte)
AL	Accumulator (low byte)
8W	Register BW (16 bits)
CW	Register CW (16 bits)
CL	Register CL (low byte)
DW	Register DW
SP	Stack pointer (16 bits)
8P	Base Pointer (16 bits)
PC	Program counter (16 bits)
PSW	Program status word (16 bits)
IX	Index register (source) (16 bits)
IY	Index register (destination) (16 bits)
PS	Program segment register (16 bits)
DS3	Extended data segment 3 register (16 bits)
DS2	Extended data segment 2 register (16 bits)
DS1	Data segment 1 register (16 bits)
DS0	Data segment 0 register (16 bits)
SS	Stack segment register (16 bits)
AC	Auxiliary carry flag
CY	Carry flag
P	Parity flag
S	Sign flag
z	Zero flag
DIR	Direction flag
IE	Interrupt enable flag
V	Overflow flag
IBRK	I/O break flag
BRK	-
RBO	Break flag Resister back Office
RB1	Register bank 0 flag
RB2	Register bank 1 flag
RB3	Register bank 2 flag
VPC	Register bank 3 flag
()	Vector PC
	Contents of memory indicated by contents of in parenthesis
disp	Displacement (8/16-bit)
temp	Temporary register (8/16/32 bits)
ext-disp8	16 bits with 8-bit displacement sign-extended
seg	Immediate segment data (16 bits)
offset	immediate offset data (16 bits)
←	Transfer direction
+	Addition
-	Subtraction
×	Multiplication
+	Division
%	Modulo
^	Logical product (AND)
×	Logical sum (OR)
¥	Exclusive logical sum (exclusive OR)
ххH	2-digit hexadecimal number
xxxxH	4-digit hexadecimal number
1	Alternate function, or

Table 17-3	. Operation	Description	Legend
------------	-------------	-------------	--------

82

🔲 6427525 OO63188 943 🖿

Identifier	Description	
(Blank)	No change	
0	Cleared to 0	
1	Set to 1	
×	Set or cleared depending on result	
U	Undefined	
R	Previously saved value is restored	

Table 17-4. Flag Operation Legend

Table 17-5.	Memory	Addressing
-------------	--------	------------

mem mod	00	01	10
000	BW + IX	BW + IX + disp8	BW + IX + disp16
001	BW + IY	BW + IY + disp8	BW + IY + disp16
010	BP + IX	BP + IX + disp8	BP + IX + disp16
011	BP + IY	BP + IY + disp8	BP + IY + disp16
100	IX	IX + disp8	IX + disp16
101	IY	IY + disp8	IY + disp16
110	Direct address	BP + disp8	BP + disp16
111	BW	BW + disp8	BW + disp16

Note When BP is used in memory addressing other than in a primitive instruction, the default segment register is SS. When BP is not used, the default segment register is DS0.

In primitive instruction memory addressing, the destination block default segment register is DS1. In memory addressing, the source block default segment register is DS0.

Table 17-6.	8/16-Bit	General	Register	Selection
-------------	----------	---------	----------	-----------

reg	W = 0	W = 1
000	AL	AW
001	CL	cw
010	DL	DW
011	BL	BW
100	АН	SP
101	СН	BP
110	DH	IX
111	вн	IY

Table 1	17-7.	Segment	Register	Selection
---------	-------	---------	----------	-----------

sreg	
00	DS1
01	PS
10	SS
11	DS0

Table 17-8. Extended Segment Register Selection

xsreg	
10	DS3/VPC
11	DS2

🖬 6427525 OO63189 88T 🖿

Number of Clock Cycles

In the case of a memory operand the number of clock cycles depends on the addressing mode. The following numbers should be used for "EA" in Table 17-9 "Number of Clock Cycles"

mod mem	00	Clock Cycles	01	Clock Cycles	10	Clock Cycles
000	BW + IX	3	BW + IX + disp8	3	BW + IX + disp16	3
001	BW + IY	3	BW + IY + disp8	3	BW + IY + disp16	3
010	BP + IX	3	BP + IX + disp8	3	BP + IX + disp16	3
011	BP + IY	3	BP + IY + disp8	3	BP + IY + disp16	3
100	IX	2	IX + disp8	2	IX + disp16	2
101	IY	2	IY + disp8	2	IY + disp16	2
110	Direct address	2	BP + disp8	2	BP + disp16	2
111	BW	2	BW + disp8	2	BW + disp16	2

"T" indicates the number of wait states. Any number of wait states from "0" (no wait) up can be used.

r tion			idth•	Byte Pro	ocessing	Word Pr	ocessing
Instruction Group	Mnémonic	Operands	Bus Width [•]	On-Chip RAM Access	Other Access	On-Chip RAM Access	Other Access
		reg, reg'	-	2	2	2	2
		mem, reg		EA + 2	EA + 3	EA + 2	EA + 3
			8	54 0			EA + 8 + 2T
		reg, mem	16	EA + 2	EA + 5 + T	EA + 2	EA + 5 + T
		mem, imm	-	EA + 2	EA + 3	EA + 2	EA + 3
		reg, imm	-	2	2	2	2
		acc, dmem	8	4			10 + 2T
		acc, dmem	16	4	7 + T	4	7 + T
		dmem, acc	—	4	5	4	5
		sreg, reg16	_		—	2	2
SUC		xsreg, reg16 VPC, reg16	8	8		2	2
ructio			16			2	2
Data transfer instructions	MOV	sreg, mem16	8	_	_	EA + 2	EA + 8 + 2T
ansfe			16				EA + 5 + T
ita tra		xsreg,mem16/ VPC, mem16		EA + 2	EA + 8 + 2T		
õ			16	16			EA + 5 + T
		reg16, sreg	—	-	-	2	2
		reg16, xsreg/ reg16, VPC	8 16		_	2	2
		mem16, sreg	_	_	_	EA + 2	EA + 3
		mem16, xsreg/ mem16, VPC	8 16	_	_	EA + 2	EA + 3
		DS0,	8	<u></u>			EA + 17 + 4T
		reg16, mem32	16		-	EA + 5	EA + 11 + 2T
		DS2,	8				EA + 17 + 4T
		reg16, mem32	16		-	EA + 5	EA + 11 + 2T
		DS1,	8				EA + 17 + 4T
		reg16, mem32	16		-	EA + 5	EA + 11 + 2T
		DS3,	8	<u></u>			EA + 17 + 4T
		reg16, mem32	16	-	_	EA + 5	EA + 11 + 2T

Table 17-9. Number of Clock Cycles (1/20)

* 8 : 8-bit width 16 : 16-bit width — : Both 8-bit and 16-bit bus width

6427525 0063191 438 🛲

500			-#	Byte Pro	cessing	Word Processing		
Group	Mnemonic	Operands	Bus Width*	On-Chip RAM Access	Other Access	On-Chip RAM Access	Other Access	
		AH, PSW	-	2	2	—	—	
Data transfer instructions	MOV		8	3	3			
		PSW, AH	16	2	2	1 -	-	
	LDEA	reg16, mem16	-			EA + 2	EA + 2	
struc	TRANS/ TRANSB	src-table	—	6	9 + T	-	-	
fer in		reg, reg'		4	4	4	4	
trans	NOU	mem, reg/				FA . 4	EA + 10 + 2T	
Data	хсн	reg, mem		EA + 4	EA + 7 + T	EA + 4	EA + 7 + T	
		AW, reg16/ reg16, AW			—	4	4	
	MOVSPA					8	8	
	MOVSPB	reg16	-			9	9	
	REPC		-	0 to 1	0 to 1	0 to 1	0 to 1	
fixes	REPNC			0 to 1	0 to 1	0 to 1	0 to 1	
Repeat prefixes	REP/ REPE/ REPZ		-	0 to 1	0 to 1	0 to 1	0 to 1	
æ	REPNE/ REPNZ			0 to 1	0 to 1	0 to 1	0 to 1	
	MOVBK	dst-block,	- 8	18 + T 19 + T		21 + 2T	22 + 2T	
		src-block			19 + 1	9 + (14 + 2T)n	9 + (18 + 4T)n	
ŝ				(rep)		5	5	
uctio	MOVBKB/			9 + (11 + T)n	9 + (12 + 2T)n	- 18 + T	19 + T	
instr	MOVBKW		16	(rep CW = 0)		9 + (11 + T)n	9 + (12 + 2T)n	
nsfer				5	5	5	5	
Primitive block transfer instructions	СМРВК	src-block,			00 07	23 + 2T	28 + 4T	
bloc		dst-block	8	20 + T	22 + 2T	9 + (16 + 2T)n	9 + (21 + 4T)n	
nitive				(rep)	0 . (15 . 27)-	5	5	
Prin				9 + (13 + T)n	9 + (15 + 2T)n	- 20 + T	22 + 2T	
	СМРВКЖ		16	(rep CW = 0)		9 + (13 + T)n	9 + (15 + 2T)n	
		1		5	5	5	5	

Table 17-9	Number o	f Clock (Cycles	(2/20)
------------	----------	-----------	--------	--------

.

8 : 8-bit width 16 : 16-bit width

- : Both 8-bit and 16-bit bus width

Remark n: Number of repetitions

86

64275250063192374 📖

tion t	to to the tot to to the tot to to to the tot to		dth-	Byte Pro	cessing	Word Processing		
Instruction Group	Mnemonic	Operands	Bus Width*	On-Chip RAM Access	Other Access	On-Chip RAM Access	Other Access	
	СМРМ	dst-block			4.4		20 + T	
			8	15	17 + T	15	10 + (12 + 2T)n	
	0140147			(rep) 10 + 7n	10 + (9 + T)n	10 + 7n	5	
	CMPMB/ CMPMW				10 + (5 + 1)/1		- 17 + T	
			16	(rep CW = 0)			10 + (9 + T)n	
s				5	5	5	5	
Lotior	LDM	src-block		10	40 T		16 + T	
instr			8	10	13 + T	10	9 + (9 + 2T)n	
Primitive block transfer instructions				(rep) 9 + 3n	9 + (6 + T)n	9 + 3n	5	
k tra	LDMB/ LDMW				9 + (0 + 1)/1	9 + 3n	- 13 + T	
plo		16	16	(rep CW = 0)			9 + (6 + T)n	
nitive				5	5	5	5	
Prir	STM	dst-block	8	12	13	12	13	
				12	13	12	9 + (9 + 2T)n	
			(rep) 9 + 5n	9 + (6 + T)n	9 + 5n	5		
	STMB/ STMW				5 + (6 + 1/11	9 + 5h	- 13	
		16	16	(rep CW = 0)			9 + (6 + T)n	
				5	5	5	5	
Su		reg8, reg8'	8			22 to 63	31 to 72	
uctio	INS	1890, 1890	16		_	22 10 03	23 to 64	
instr		reg8, imm4	8			22 to 63	31 to 72	
ation		.ego, mmi4	16			22 10 03	23 to 64	
nipul		reg8, reg8'	8			19 to 41	19 + 2T to 48 + 4T	
Bit field manipulation instructions	EXT		16		_	13 (0.41	19 to 42 + 2T	
it fiel		reg8, imm4	8			19 +0 41	19 + 2T to 48 + 4T	
			16	_	_	— 19 to 41		

Table 17-9	I. Number	of Clock	Cycles	(3/20)
------------	-----------	----------	--------	--------

16:16-bit width

- : Both 8-bit and 16-bit bus width

Remark n: Number of repetitions

🖬 6427525 OO63193 2OO 🖿

n o			h •1	Byte Pro	cessing	Word Pr	ocessing
Instruction Group	Mnemonic	Operands	Bus Width	On-Chip RAM Access	Other Access	On-Chip RAM Access	Other Access
			8				10 + 2T
su		acc8, imm8	16		7 + T	_	7 + T
uctio	IN*2		8		7 . 7		10 + 2T
Input/output instructions		acc, DW	16	1	7+Ť		7 + T
utput			8		5	_	5
put/o	OUT-2	imm8, acc	16	-	5		
5		DW, acc	8		5	_	5
			16		5		
		dst-block, DW		17 + T	18 + T	20 + 2T	21 + 2T
			8			9 + (13 + 2T)n	9 + (17 + 4T)n
ŝ	INM*2			(rep) 9 + (10 + T)n	9 + (11 + 2T)n	5	5
uctio	1010-2		16	9+(10+1/1		17 + T	18 + T
instr				(rep CW = 0)		9 + (10 + T)n	9 + (11 + 2T)n
utput				5	5	5	5
put/o				14 + T	17 + 2T	17 + 2T	23 + 4T
ve in			8	14 + 1	17 + 21	9 + (10 + 2T)n	9 + (16 + 4T)n
Primitive input/output instructions				(rep)	9 + (10 + 2T)n	5	5
Ĩ.	OUTM*2	DW, src-block		9 + (7 + T)n	9 + (10 + 21)n	- 14 + T	17 + 2T
			16	(rep CW = 0)		9 + (7 + T)n	9 + (10 + 2T)n
				5	5	5	5

Table 17-9.	Number	of Clock	Cycles	(4/20)
-------------	--------	----------	--------	--------

16:16-bit width

2. When $\overline{IBRK} = 1$. As shown in the next page when $\overline{IBRK} = 0$.

Remark n: Number of repetitions

p n				€ Bγte Pro		cessing	Word Pro	cessing
Instruction Group	Mnemonic	Operands	Bus Wi	On-Chip RAM Access	Other Access	On-Chip RAM Access	Other Access	
		acc8, imm8	8		60 + 10T		60 + 10T	
su	IN	acco, immo	16	<u> </u>	40 + 5T		40 + 5T	
uctio		acc, DW	8		60 + 10T		60 + 10T	
inst			16	—	40 + 5T	_	40 + 5T	
Input/output instructions		imm8, acc	8		60 + 10T		60 + 10T	
put/o	ουτ		16	-	40 + 5T	-	40 + 5T	
5	001		8		60 + 10T		60 + 10T	
		DVV, acc	16	—	40 + 5T		40 + 5T	
out/ tions	INM	dst-block, DW	8	60 + 10T		60 + 10T		
/e inp		dst-block, Day	16		40 + 5T		40 + 5T	
Primitive input/ output instructions	ουτΜ	DW, src-block	8		60 + 10T		60 + 10T	
Pr outp		DITY, BIG*DIOCK	16	_	40 + 5T		40 + 5T	

16:16-bit width

🖿 6427525 OO63195 O&3 🛲

Б о			÷	Byte Pro	cessing	Word Pr	ocessing
Instruction Group	Mnemonic	Operands	Bus Width [•]	On-Chip RAM Access	Other Access	On-Chip RAM Access	Other Access
		reg, reg'	-	3	3	3	3
			8			54.4	EA + 10 + 2T
		mem, reg	16	EA + 4	EA + 7 + T	EA + 4	EA + 7 + T
			8	E4 . 0	EA + 6 + T	EA + 2	EA + 9 + 2T
	ADD	reg, mem	16	EA + 2	EA + 0 + 1	EA + 2	EA + 6 + T
		reg, imm	-	2	2	2	2
		mem, imm	8	 E A + 4	EA + 7 + T	EA + 4	EA + 10 + 2T
		mem, imm	i, imm EA + 4 EA + 7 + T E/		EA + 7 + T		
		acc, imm	-	2	2	2	2
s		reg, reg'	-	3	3	3	3
ction		mem, reg	8	EA + 4	EA + 7 + T EA	EA + 4	EA + 10 + 2T
nstru			16	EA + +			EA + 7 + T
Addition/subtraction instructions		reg, mem	8 EA + 2 EA + 6 + T	EA + 2	EA + 9 + 2T		
btrac	ADDC		16				EA + 6 + T
on/su		reg, imm	-	2	2	2	2
vdditi		mem, imm	8	EA + 4	EA + 7 + T	EA + 4	EA + 10 + 2T
٩			16				EA + 7 + T
		acc, imm	-	2	2	2	2
		reg, regʻ	-	. 3	3	3	3
		mem, reg	8	EA + 4	EA + 7 + T	EA + 4	EA + 10 + 2T
		menn, reg	16				EA + 7 + T
		reg, mem	8	EA + 2	EA + 6 + T	EA + 2	EA + 9 + 2T
	SUB	iog, mem	16				EA + 6 + T
		reg, imm	-	2	2	2	2
		mem, imm	8	EA + 4	EA + 7 + T	EA + 4	EA + 10 + 2T
			16				EA + 7 + T
		acc, imm	-	2	2	2	2

Table 17-	9. Number	of Clock	Cycles	(6/20)
-----------	-----------	----------	--------	--------

16:16-bit width

- : Both 8-bit and 16-bit bus width

90

🔲 6427525 0063196 T1T 📰

P tion			÷ t	Byte Pro	ocessing	Word Pr	ocessing
Instruction Group	Mnemonic	Operands	Bus Width*	On-Chip RAM Access	Other Access	On-Chip RAM Access	Other Access
		reg, regʻ	-	3	3	3	3
ions			8	EA + 4	EA + 7 + T	FA . 4	EA + 10 + 2T
tructi		mem, reg	16	EA + •	EA + / + 1	EA + 4	EA + 7 + T
Addition/subtraction instructions			8	EA + 2	EA . C. T	FA . 0	EA + 9 + 2T
ractio	SUBC	reg, mem	16	EA + 2	EA + 6 + T	EA + 2	EA + 6 + T
/subt		reg, imm		2	2	2	2
lition		mem, imm	8	EA + 4	EA + 7 + T	EA + 4	EA + 10 + 2T
Ăd			16	EA + 4	EA + 7 + 1	EA + 4	EA + 7 + T
		acc, imm	-	2	2	2	2
	ADD4S	dst-string, src-string	8	6 + (15 + T)n	6 + (19 + 3T)n		
			16			_	_
ions	SUB4S	dst-string, src-string	8	— 6 + (16 + T)n 6 + (20 + 3T)n —	6 + (20 + 3T)n		_
struct			16				
BCD operation instructions	CMP4S	dst-string,	8	6 + (15 + T)ri	6 + (18 + 2T)n		
eratic		src-string	16				
d D	ROL4	reg8	8	5	5	_	_
B		mem8	16	EA + 5	EA + 8 + T		_
	ROR4	reg8	8	5	5	_	-
<u> </u>		mem8	16	EA + 5	EA + 8 + T		
su		reg8		2	2		
ructio	INC	mem	8	EA + 3	EA + 7 + T	EA + 3	EA + 10 + 2T
t inst			16				EA + 7 + T
ment		reg16	-	_	_	2	2
Increment/decrement instructions		reg8		2	2		
nent/	DEC	mem	8	EA + 3	EA + 7 + T	EA + 3	EA + 10 + 2T
ncrer			16				EA + 7 + T
<u> </u>		reg16	-	_	-	2	2

Table	17-9.	Number	of Clock	Cycles	(7/20)
-------	-------	--------	----------	--------	--------

16:16-bit width

- : Both 8-bit and 16-bit bus width

Remark n: Half of number of BCD digits

6427525 0063197 956 📟

u o			÷	Byte Pro	cessing	Word Pr	ocessing
Group	Mnemonic	Operands	Bus Width ^e	On-Chip RAM Access	Other Access	On-Chip RAM Access	Other Access
		reg8	—	11	11	15	15
		_	8		EA + 14 + T	EA + 16	EA + 21 + 2T
	MULU	mem8	16	EA + 12	EA + 14 + 1	EA + 10	EA + 18 + T
	MOLO	reg16	-	11	11	15	15
			8		E A A A A	EA + 16	EA + 21 + 2T
		mem16	16	EA + 12	EA + 14 + T	EA + 16	EA + 18 + T
s		reg8	_	10	10	14	14
ction		mem8	8	EA + 11	EA + 13 + T	EA + 15	EA + 20 + 2T
instru			16	EA + 11	EA + 13 + 1	EA + 15	EA + 17 + T
Multiplication instructions		reg16		10	1.0	14	14
iplica			8		FA 40 T	EA + 15	EA + 20 + 2T
Mult		mem16	16	EA + 11	EA + 13 + T	EA + 15	EA + 17 + T
	MUL	reg16, reg16', imm8/reg16, imm8	-		-	14	14
		reg16,	8			EA + 15	EA + 20 + 2T
		mem16, imm8	16	_		EAT 15	EA + 17 + T
		reg16, reg16', imm16/reg16, imm16	-	_	_	14	14
		reg16,	8		· - · · ·	EA + 15	EA + 20 + 2T
		mem16, imm16	16	_		EA + 15	EA + 17 + T

Table 17-9.	Number	of Clock	Cycles	(8/20)
-------------	--------	----------	--------	--------

16:16-bit width

- : Both 8-bit and 16-bit bus width

🖿 6427525 0063198 892 페

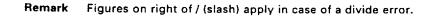
.

tion T			tt t	Byte Pro	cessing	Word Pr	ocessing
Instruction Group	Mnemonic	Operands	Bus Width*	On-Chip RAM Access	Other Access	On-Chip RAM Access	Other Access
			8	15/62 + 10T	15/62 + 10T	23/57 + 10T	23/57 + 10T
		reg8	16	15/42 + 5T	15/42 + 5T	23/42 + 5T	23/42 + 5T
			8	EA + 16/63 + 10T	EA + 18 + T/63 + 10T	EA + 24/58 + 10T	EA + 30 + 2T/58 + 10T
	DIVU	mem8	16	EA + 16/43 + 5T	EA + 18 + T/63 + 5T	EA + 24/43 + 5T	EA + 26 + T/43 + 5T
			8	15/62 + 10T	15/62 + 10T	23/57 + 10T	23/57 + 10T
		reg16	16	15/42 + 5T	15/42 + 5T	23/42 + 5T	23/42 + 5T
Division instructions			8	EA + 16/63 + 10T	EA + 18 + T/63 + 10T	EA + 24/58 + 10T	EA + 30 + 2T/58 + 10T
nstru		mem16	16	EA + 16/43 + 5T	EA + 18 + T/43 + 5T	EA + 24/43 + 5T	EA + 26 + T/43 + 5T
sion i			8	17/64 + 10T	17/64 + 10T	25/59 + 10T	25/59 + 10T
Divi		reg8	16	17/44 + 5T	17/44 + 5T	25/44 + 5T	25/44 + 5T
	DIV	mem8	8	EA + 18/65 + 10T	EA + 20 + T/65 + 10T	EA + 26/60 + 10T	EA + 31 + 2T/60 + 10T
			16	EA + 18/45 + 5T	EA + 20 + T/45 + 5T	EA + 26/45 + 5T	EA + 28 + T/45 + 5T
		reg16	8	17/64 + 10T	17/64 + 10T	25/59 + 10T	25/59 + 10T
			16	17/44 + 5T	17/44 + 5T	25/44 + 5T	25/44 + 5T
		mem16	8	EA + 18/65 + 10T	EA + 20 + T/65 + 10T	EA + 26/60 + 10T	EA + 31 + 2T/60 + 10T
			16	EA + 18/45 + 5T	EA + 20 + T/45 + 5T	EA + 26/45 + 5T	EA + 28 + T/45 + 5T
tions	ADJBA		8	6	9		
struc			16	9	5		_
BCD adjustment instructions	ADJ4A			3	3	_	_
Istme	ADJBS		8	6	6		
) adju	-0303		16	9	9		
BCC	ADJ4S		-	3	3		-
10	CVTBD			18	18		
Data conversion instructions	CVTDB		-	8	8	_	-
ata co	сутву			3	3		
	CVTWL			_		3	3

Table 17-9.	Number	of Clock	Cycles	(9/20)
-------------	--------	----------	--------	--------

16:16-bit width

- : Both 8-bit and 16-bit bus width



🎟 6427525 OO63199 729 🎟

ç e				Byte Processing		Word Processing			
Group	Mnemonic	Operands	Bus Width*	On-Chip RAM Access	Other Access	On-Chip RAM Access	Other Access		
		reg, regʻ		3	3	3	3		
ion s					8				EA + 9 + 2T
		mem, reg	16	EA + 4	EA + 6 + T	EA + 4	EA + 6 + T		
truct			8	FA - 2	EA + 6 + T	EA + 2	EA + 9 + 2T		
in in	СМР	reg, mem	16	EA + 2	EA + 6 + 1	EA + 2	EA + 6 + T		
arisc		reg, imm	1-1	2	2	2	2		
Comparison instructions			8				EA + 9 + 2T		
-		mem, imm	16	EA + 4	EA + 6 + T	EA + 4	EA + 6 + T		
		acc, imm	1-1	2	2	2	2		
		reg	_	2	2	2	2		
	NOT	mem 8 EA + 3 EA + 7 + T EA	54.5	EA + 10 + 2T					
Complement operation instructions			mem	16	EA + 3	EA + / + I	EA + 3	EA + 7 + T	
	NEG	reg	1-1	2	2	2	2		
		mem	8	EA . A	FA	FA . 0	EA + 10 + 2T		
3		mem	16	EA + 3	EA + 7 + T	EA + 3	EA + 7 + T		
		reg, reg'	1-1	3	3	3	3		
		mem, reg/ reg, mem	mem, reg/ 8	8			5 A	EA + 9 + 2T	
			16 EA + 4	EA + 4	EA + 6 + T	EA + 4	EA + 6 + T		
	TEST	reg, imm		2	2	2	2		
		mem, imm	mem, imm 8	8				EA + 9 + 2T	
uctions				16	EA + 4	EA + 6 + T	EA + 4	EA + 6 + T	
struct		acc, imm	1-1	2	2	2	2		
on in		reg, reg'	1-1	3	3	3	3		
erati			8	EA - 4	EA . 7 . 7	EALT	EA + 10 + 2T		
Logical operation instr		mem, reg	16	EA + 4	EA + 7 + T	EA + 4	EA + 7 + T		
Logic			8	EA : 0	EALCIT	EA : 2	EA + 9 + 2T		
	AND	reg, mem	16	EA + 2	EA + 6 + T	EA + 2	EA + 6 + T		
		reg, imm		2	2	2	2		
			8		FA	FA	EA + 10 + 2T		
		mem, imm	16	EA + 4	EA + 7 + T	EA + 4	EA + 7 + T		
,		acc, imm	1_1	2	2	2	2		

Table 17-9.	Number	of Clock	Cycles	(10/20)
12040 17-3.	140111DBI	OF GIUCK	CACIG2	(10/20/

94

🗰 6427525 0063200 270 📟

p p			ţ;	Byte Pro	cessing	· · · · · · · · · · · · · · · · · · ·	ocessing
Instruction Group	Mnemonic	Operands	Bus Width*	On-Chip RAM Access	Other Access	On-Chip RAM Access	Other Access
		reg, regʻ	-	3	3	3	3
			8				EA + 10 + 2T
		mem, reg	16	EA + 4	EA + 7 + T	EA + 4	EA + 7 + T
		reg, mem	8	EA + 2	EA + 6 + T	EA + 2	EA + 9 + 2T
	OR	reg, mem	16				EA + 6 + T
ļ		reg, imm	-	2	2	2	2
tions		mem, imm	8	EA + 4	EA + 7 + T	EA + 4	EA + 10 + 2T
struc			16		201771		EA + 7 + T
Logical operation instructions		acc, imm	I	2	2	2	2
oerati		reg, regʻ	-	3	3	3	3
cal of	XOR	mem, reg	8	EA + 4	EA + 7 + T	EA + 4	EA + 10 + 2T
Logi			16				EA + 7 + T
		reg, mem	8	EA + 2	EA + 6 + T	EA + 2	EA + 9 + 2T
			16				EA + 6 + T
		reg, imm	-	2	2	2	2
		mem, imm	em, imm	EA + 4	EA + 7 + T	EA + 4	EA + 10 + 2T
			16				EA + 7 + T
L		acc, imm	-	2	2	2	2
		reg8, CL	_	3	3	3	3
		mem8, CL	8	EA + 4	EA + 6 + T	EA + 4	EA + 9 + 2T
			16				EA + 6 + T
ions		reg16, CL	-	3	3	3	3
struct		mem16, CL	8	EA + 4	EA + 6 + T	EA + 4	EA + 9 + 2T
n ins	TEST1		16				EA + 6 + T
ulatic		reg8, imm3		2	2	2	2
Bit manipulation instructions		mem8, imm3	8	EA + 4	EA + 6 + T	EA + 4	EA + 9 + 2T
Bit n			16				EA + 6 + T
		reg16, imm4	-	2	2	2	2
		mem16, imm4	8	EA + 4	EA + 6 + T	EA + 4	EA + 9 + 2T
			16				EA + 6 + T

Table 17-9. Number of Clock Cycles (11/20)

* 8:8-bit width 16:16-bit width —: Both 8-bit and 16-bit bus width

🔳 6427525 006320l l07 📟

P ü				Byte Pro	ocessing	T	rocessing
Instruction Group	Mnemonic	Operands	Bus Width•	On-Chip RAM Access	Other Access	On-Chip RAM Access	Other Access
		reg8, CL	-	3	3	3	3
		mem8, CL	8 16	EA + 4	EA + 7 + T	EA + 4	EA + 10 + 2T EA + 7 + T
		reg16, CL	-	3	3	3	3
			8				EA + 10 + 2T
		mem16, CL	16	EA + 4	EA + 7 + T	EA + 4	EA + 7 + T
	NOT1	reg8, imm3	-	2	2	2	2
			8	EA . 4	EA . 7 . T	FA . 4	EA + 10 + 2T
		mem8, imm3	16	EA + 4	EA + 7 + T	EA + 4	EA + 7 + T
		reg16, imm4	-	2	2	2	2
tions		mem16, imm4	8	EA + 4	EA + 7 + T		EA + 10 + 2T
Bit manipulation instructions			16		EA + 7 + T	EA + 4	EA + 7 + T
on in		CY	-	2	2	2	2
pulati		reg8, CL		3	3	3	3
manil		mem8, CL	8	EA + 4	EA + 7 + T	EA + 4	EA + 10 + 2T
Bit			16				EA + 7 + T
		reg16, CL	-	3	3	3	3
		mem16, CL	8	EA + 4	EA + 7 + T	A + 7 + T EA + 4	EA + 10 + 2T
		men 10, CL	16	EA + +	EA + 7 + 1	EA + 4	EA + 7 + T
	CLR1	reg8, imm3	—	2	2	2	2
			8	5	FA . 7 . 7		EA + 10 + 2T
		m em 8, imm3	16	EA + 4	EA + 7 + T	EA + 4	EA + 7 + T
		reg16, imm4	-	2	2	2	2
		mem16, imm4 -	8	EA : 4	EA . 7 . 7	EA . 4	EA + 10 + 2T
			16	EA + 4	EA + 7 + T	EA + 4	EA + 7 + T
		CY	—	2	2	2	2
		DIR		2	2	2	2

Teble 17-9.	Number o	of Clock	Cycles	(12/20)
				(

16:16-bit width

- : Both 8-bit and 16-bit bus width

96

■ 6427525 OO632O2 O43 ■

p tion			dth.	Byte Pro	cessing	Word Pro	ocessing
Instruction Group			Bus Width*	On-Chip RAM Access	Other Access	On-Chip RAM Access	Other Access
		reg8, CL		3	3	3	3
		mem8, CL	8	EA + 4	EA + 7 + T	EA A	EA + 10 + 2T
			16			EA + 4	EA + 7 + T
ł		reg16, CL	-	3	3	3	3
		mem16, CL	8	EA + 4	EA + 7 + T	EA + 4	EA + 10 + 2T
ctions		mentio, CL	16			EA + +	EA + 7 + T
nstru	SET1	reg8, imm3	—	2	2	2	2
Bit manipulation instructions		mem8, imm3	8	EA + 4	EA + 7 + T	EA + 4	EA + 10 + 2T
ipula			16			EA + 4	EA + 7 + T
man		reg16, imm4		2	2	2	2
Bi		mem16, imm4	8	EA + 4	EA + 7 + T	EA + 4	EA + 10 + 2T
			16				EA + 7 + T
		CY	-	2	2	2	2
		DIR	—	2	2	2	2
		mem	8	EA + 8 + 3n + T	EA + 8 + 3n + T	EA + 11 + 3n + 2T	EA + 11 + 3n + 2T
	BSCH		16	EA + 8 + 3n + T	EA + 8 + 3n + T	EA + 8 + 3n + T	EA + 8 + 3n + T
<u> </u>		reg	—	4 + 3n	4 + 3n	4 + 3n	4 + 3n
		reg, 1	-	3	3	3	3
		mem, 1	8	EA + 3	EA + 7 + T	EA + 3	EA + 10 + 2T
suc			16				EA + 7 + T
instructions	;	reg, CL	-	5 + n	5 + n	5 + n	5 + n
	SHL	mem, CL	8	EA + 5 + n	EA + 8 + T + n	EA + 6 + n	EA + 11 + 2T + n
Shift			16				EA + 8 + T + n
		reg, imm8		5 + n	5 + n	5 + n	5 + n
		mem, imm8	8	EA + 6 + n	EA + 8 + T + n	EA + 6 + n	EA + 11 + 2T + n
			16				EA + 8 + T + n

16:16-bit width

- : Both 8-bit and 16-bit bus width



📕 6427525 0063203 TAT 📕 -

p ö			dth +	Byte Pro	ocessing	Word Processing			
Instruction Group	ຊີວິ Mnemonic Operands ເຮັບ 		Bus Width*	On-Chip RAM Access	Other Access	On-Chip RAM Access	Other Access		
		reg, 1	-	3	3	3	3		
						8			540
		mem, 1	16	EA + 3	EA + 7 + T	EA + 3	EA + 7 + T		
		reg, CL	-	5 + n	5 + n	5 + n	5 + n		
	SHR		8	F A F	54 . Q . T	ĖA + 6 + n	EA + 11 + 2T + n		
		mem, CL	16	EA + 5 + n	EA + 8 + T + n	EA + 6 + 1	EA + 8 + T + n		
		reg, imm8	-	5 + n	5 + n	5 + n	5 + n		
SE SE		·	8		54 . 0 . T	EA + 6 + n	EA + 11 + 2T + n		
Shift instructions		mem, imm8	16	EA + 0 + 1	EA+6+n .EA+8+T+n EA+6		EA + 8 + T + n		
t inst	SHRA	reg, 1	-	3	3	3	3		
Shir		mem, 1	8	EA + 3	EA + 7 + T	EA + 3	EA + 10 + 2T		
			16	1	EA + 7 + 1	EATS	EA + 7 + T		
		reg, CL	-	5 + n	5 + n	5 + n	5 + n		
		mem, CL	8	EA + 5 + n	EA + 8 + T + n	EA + 6 + n	EA + 11 + 2T + n		
		mem, CL	16				EA + 8 + T + n		
		reg, imm8	-	5 + n	5 + n	5 + n	5 + n		
		8	8	EA + 6 + n	EA + 8 + T + n	EA + 6 + n	EA + 11 + 2T + n		
		mem, imm8	16	EA + 0 + 11	EATOTITI		EA + 8 + T + n		
		reg, 1	-	3	3	З	3		
		mem, 1	8	EA + 3	EA + 7 + Τ	EA + 3	EA + 10 + 2T		
S			16	Ento			EA + 7 + T		
ructions		reg, CL	-	5 + n	5 + n	5 + n	5 + n		
Rotate instru	ROL	mem Cl	8	EA + 5 + n	EA + 8 + T + n	EA + 6 + n	EA + 11 + 2T + n		
otate		mem, CL	16				EA + 8 + T + n		
<u>۳</u>		reg, imm8	-	5 + n	5 + n	5 + n	5 + n		
		mem, imm8	8	EA + 6 + n	EA + 8 + T + n	EA + 6 + n	EA + 11 + 2T + n		
		mant, numb	16				EA + 8 + T + n		

Table	17-9.	Number	of	Clock	Cvcles	(14/20)
					-,	

16 : 16-bit width

--- : Both 8-bit and 16-bit bus width

Remark Number of shifts (n in a bit manipulation instruction indicates the bit number searched for)

98

🔳 6427525 0063204 916 📖

tion T			dth•	Byte Pro	cessing	Word P	rocessing		
Instruction Group	Mnemonic	Operands	Bus Width ^e	On-Chip RAM Access	Other Access	On-Chip RAM Access	Other Access		
		reg, 1	-	3	3	3	3		
			8	EA + 3	EA + 7 + T	54	EA + 10 + 2T		
		mem, 1	16	EA + 3	EA + / + 1	EA + 3	EA + 7 + T		
		reg, CL	-	5 + n	5 + n	5 + n	5 + n		
	ROR	Cl	8	EALELE	FA	F 4 0	EA + 11 + 2T + n		
		mem, CL	16	EA + 5 + n	EA + 8 + T + n	EA + 6 + n	EA + 8 + T + n		
		reg, imm8	-	5 + n	5 + n	5 + n	5 + n		
		mem, imm8	8	EA + 6 + n	EA + 8 + T + n	EA + 6 + n	EA + 11 + 2T + n		
			16			EA + 6 + n	EA + 8 + T + n		
	ROLC	reg, 1	-	3	3	3	3		
		mem, 1	8	8 EA + 3	EA + 7 + T	EA + 3	EA + 10 + 2T		
SU			16			EATS	EA + 7 + T		
Rotate instructions		reg, CL	-	5 + n	5 + n	5 + n	5 + n		
insti		mern, CL –	B EA	EA + 5 + n	EA + 8 + T + n	EA + 6 + n	EA + 11 + 2T + n		
lotate			16				EA + 8 + T + n		
E		reg, imm8	-	5 + n	5 + n	5 + n	5 + n		
		mem, imm8	8	EA + 6 + n	EA + 8 + T + n	EA + 6 + n	EA + 11 + 2T + n		
			16				EA + 8 + T + n		
		reg, 1	-	3	3	3	3		
		mem, 1	8	EA + 3	EA + 7 + T	EA + 3	EA + 10 + 2T		
			16				EA + 7 + T		
		reg, CL	_	5 + n	5 + n	5 + n	5 + n		
	RORC	mem, CL	8	EA + 5 + n	EA + 8 + T + n	EA + 6 + n	EA + 11 + 2T + n		
			16				EA + 8 + T + n		
		reg, imm8		5 + n	5 + n	5 + n	5 + n		
		mem, imm8	8	EA + 6 + n	EA + 8 + T + n	EA + 6 + n	EA + 11 + 2T + n		
	me	mem, imm8	mem, imm8	mem, imm8	16				EA + 8 + T + n

Table 17-9.	Number	of Clock	Cycles	(15/20)
-------------	--------	----------	--------	---------

16:16-bit width

- : Both 8-bit and 16-bit bus width

Remark Number of shifts

🖬 6427525 0063205 852 🔳

tion P			Syte Pro		cessing	Word Pr	Word Processing	
Instruction Group	Mnemonic	Operands	Bus Width *1	On-Chip RAM Access	Other Access	On-Chip RAM Access	Other Access	
		near-proc	8				19 + 2T	
			16	_	—	_	16 + T	
			8				18 + 2T	
		regptr16	16			-	15 + T	
			8			EA + 19 + 2T	EA + 24 + 4T	
SUG	CALL	memptr16	16	_	·	EA + 16 + T	EA + 18 + 2T	
ructic			8				29 + 4T	
l inst		far-proc	16			-	23 + 2T	
Subroutine control instructions			8			EA + 32 + 4T	EA + 44 + 8T	
ine c		memptr32	16		—	EA + 26 + 2T	EA + 32 + 4T	
brout	RET		8				18 + 2T	
Su			16] _	_	_	15 + T	
			8				19 + 2T	
		pop-value	16		_	_	16 + T	
		•2	8				26 + 4T	
			16		_	_	20 + 2T	
]	pop-value*2	8		_	_	27 + 4T	
			16				21 + 2T	
			8			54.3	EA + 13 + 2T	
		mem16	16		_	EA + 7	EA + 10 + T	
Stack manipulation instructions		reg16	-		_	_	7	
astrue		sreg	-		_		7	
tion ii	BUCU	xsreg/VPC	-	_	_	_	7	
ipulat	PUSH	PSW	-				6	
man		0	8				57 + 14T	
Stack		R	16	_	_	_	36 + 7T	
		imm8	-				6	
		imm16	-	_		_	6	

Table 17-9. Number of Clock Cycles (16
--

16:16-bit width

- : Both 8-bit and 16-bit bus width

2. Segment-external

Remark n: Number of shifts

100

🔳 6427525 0063206 799 📟

tion			÷	Byte Pro	cessing	Word Pr	ocessing
Instruction Group	Mnemonic	Operands	Bue Width +1	On-Chip RAM Access	Other Access	On-Chip RAM Access	Other Access
			8			EA + 13 + 2T	EA + 14 + 2T
		mem16	16	-	—	EA + 10 + T	EA + 11 + T
		16	8				10 + 2T
		reg16	16	—		_	7 + T
suo		erec	8				10 + 2T
Stack manipulation instructions	POP	sreg	16			_	7 + T
n ins	101	xsreg/VPC	8				10 + 2T
ulatio		Xalog/vi C	16				7 + T
anip		PSW	8	_	_	_	11 + 2T
ack n			16			_	8 + T
St		R	8			_	76 + 16T
			16				52 + 8T
	PREPARE*2	imm16, imm8		<u></u>		_	9
	DISPOSE		8	_	_	_	10 + 2T
			16	· · · · · · · · · · · · · · · · · · ·			7 + T
		near-label	-			-	9
		short-label	—				9
tions	1	regptr16				_	8
Branch instructions	BR	memptr16	8		_	EA + 9	EA + 14 + 2T
ch in			16				EA + 11 + T
Bran	7	far-label	_		_		9
		memptr32	8	_		EA + 12	EA + 24 + 4T
			16				EA + 18 + 2T

Table 17-9. Number of Clock Cycles (17/20)

16:16-bit width

- : Both 8-bit and 16-bit bus width

2. When imm8 = 0. As shown below when imm8 \ge 1.

PREPARE	imm16, imm8	8	 _	 15+2T+(16+4T)n
		16		14 + (12 + T)n

n : imm8

🖿 6427525 OO63207 625 🎟

p d			÷.	Byte Pro	cessing	Word Pr	ocessing
Instruction Group	Mnemonic	Operands	Bus Width [•]	On-Chip RAM Access	Other Access	On-Chip RAM Access	Other Access
	8 ∨	short-label	1-		—	9/3	9/3
	BNV	short-label	-	_		9/3	9/3
	BC/BL	short-label	-	_		9/3	9/3
	BNC/BNL	short-label	-	_		9/3	9/3
	BE/BZ	short-label	-	_		9/3	9/3
	BNE/BNZ	short-label	-		_	9/3	9/3
	BNH	short-label	-			9/3	9/3
	вн	short-label	-			9/3	9/3
ctions	BN	short-label	-	<u> </u>		9/3	9/3
nstru	ВР	short-label	-	_	_	9/3	9/3
Conditional branch instructions	BPE	short-label	-		_	9/3	9/3
l brar	вро	short-label	-		-	9/3	9/3
tional	BLT	short-label	1-		-	9/3	9/3
ondi	BGE	short-label	-	_	_	9/3	9/3
0	BLE	short-label	-	· · · · · · · · · · · · · · · · · · ·	_	9/3	9/3
	BGT	short-label	-		_	9/3	9/3
	DBNZNE	short-label	-	-	_	10/5	10/5
	DBNZE	short-label		_	<u> </u>	10/5	10/5
	DBNZ	short-label		_		10/5	10/5
	BCWZ	short-label	1-	-	_	10/5	10/5
		sfr, imm3	8				· · · · · · · · · · · · · · · · · · ·
	BTCLR	short-label	16	1 -	21/14		-
		sfrl, imm3	8				
	BTCLRL	short-label	16	1 -	20/13	_	

Table 17-9.	Number	of Clock	Cycles	(18/20)
-------------	--------	----------	--------	---------

16: 16-bit width

- : Both 8-bit and 16-bit bus width

102

🖬 6427525 0063208 561 🛤

tion.				Byte Pro	cessing	Word Pro	ocessing
Instruction Group	Mnemonic	Operands	Bus Width +1	On-Chip RAM Access	Other Access	On-Chip RAM Access	Other Access
		3	8				50 + 10T
	BRK*2	3	16	_		—	36 + 4T + t
	DRK"Z	imm8 (≠3)	8				52 + 10T
\$		immo (≠3)	16			_	38 + 4T + t
ction	BRKV*2		8				51 + 10 T
nstru	BULA -2		16	-		_	37 + 4T + t
Interrupt instructions	RETI		8				28 + 4T
Inter	NE II		16	-		_	22 + 2T
	RETRBI		-	_		-	9
	FINT		-	3	3	3	3
	CHKIND*3		8			EA + 11	EA + 21 + 4T
	on and o		16	_			EA + 15 + 2T
+4	BRKCS	reg16	-	·	-	12	12
	TSKSW	reg16	-	—	_	13	13
	HALT				_	_	—
ions	STOP		-	_	_	-	-
struct	IDLE				_	_	-
ol in	POLL		-	_	_	_	—
CPU control instructions	DI		-	3	3	3	3
CPU	EI		-	3	3	3	3
	BUSLOCK		-	0 to 1	0 to 1	0 to 1	0 to 1

Table	17-9.	Number	of	Clock	Cycles	s (19/20)
		149111941	•••	GIGOR	~,	

- 16:16-bit width
- : Both 8-bit and 16-bit bus width
- 2. When BRK = 1, add 50 + 10T in case of 8-bit bus width, and 34 + 4T in case of 16-bit bus.
- When (mem32) > reg16 or (mem32 + 2) < reg16, add 50 + 10T in case of 8-bit bus width, and 34 + 4T + t in case of 16-bit bus width.
- 4. Register bank switching instructions

Remarks When $T \ge 2$, t = T - 1

🖬 6427525 0063209 4T8 📟

p n			÷.	Byte Pro	pcessing	Word Pi	rocessing
Instruction Group	Mnemonic	Operands	Bus Width •1	On-Chip RAM Access	Other Access	On-Chip RAM Access	Other Access
			8				50 + 10T
	5004	fp-op	16		_		36 + 4T + t
CPU control instructions	FPO1	4	8				EA + 50 + 10 T
struc		fp-op, mem	16	-	_	_	EA + 36 + 4T + t
rol ir		f=	8				50 + 10T
cont	FPO2	fp-op	16	_	_	_	36 + 4T + t
D D D	FF 02	fa	8				EA + 50 + 10 T
		fp-op, mem	16	. –		_	EA + 36 + 4T + t
	NOP		-	4	4	4	4
•2	RSTWDT	imm8, imm8'	8		9/54 + 10T*3		
			16		9/40 + 4T + t*3		
•4			-	0 to 1	0 to 1	0 to 1	0 to 1
ipula- tions	αμουτ	imm16	-	—	-	—	-
e man nstruc	ΩΟυτ	imm16	-		_	_	
Queue manipula- tion instructions		imm16	-		_	-	-
	ALBIT		-		_	_	_
	COLTRP		-		_	_	—
tions	MHENC		_	-			
Istruc	MRENC		-			-	—
Dedicated fax instructions	SCHEOL		-		_	_	·
ated	GETBIT		_	_	_	_	
Dedic	MHDEC		_	_			
	MRDEC		-		_		—
	CNVTRP		-	_	_		—

Table 17-9. Number of Clock Cycles (20/2)	Table	17-9.	Number	of Clock	Cycles	(20/20
---	-------	-------	--------	----------	--------	--------

16:16-bit width

- : Both 8-bit and 16-bit bus width

- 2. Watchdog timer manipulation instruction
- 3. Figure after / (slash) applies when word processing is performed during data error. When T \geq 2, t = T 1
- Segment override prefix instructions (DS0:, DS1:, PS:, SS:) Extended segment override prefix instructions (DS2: DS3:) Register file space access override prefix instruction (IRAM)

104

🖿 6427525 0063210 11T 📟

	И																					
	S																					
Flags	۹.																					
"	> X																	-+				
	AC CY																					
	<																					—
	Operation	reg←reg'	(mem)←reg	reg←(mem)	(mem)←imm	reg←imm	If W = 0, AL←(dmem) If W = 1, AH←(dmem + 1), AL←(dmem)	If W = 0, (dmem)← AL If W = 1, (dmem + 1)←AH, (dmem)←AL	sreg←reg16 sreg : SS, DS0, DS1	xsreg←reg16 xsreg : DS2, DS3	sreg←(mem16) sreg : SS, DS0, DS1	xsreg←(mem16)	reg16←sreg	reg16←xsreg	(mem16)←sreg	(mem16)←xsreg	reg16← (mem32) DS0←(mem32 + 2)	reg16←(mem32) DS1←(mem32 + 2)	reg16←(mem32)	DS2←(mem32 + 2)	reg16←(mem32)	DS3←(mem32 + 2)
	Bytes	2	2 to 4	2 to 4	3 to 6	2 to 3	m	е.	8	2	2 to 4	2 to 4	7	2	2 to 4	2 to 4	2 to 4	2 to 4	3 to F		2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2	
n Code	76543210	11 reg reg'	mod reg mem	mod reg mem	mod 0 0 mem				1 1 0 sreg reg	1 1 1 xsreg reg	mod 0 sreg mem	mod 1 xsreg mem	110 sreg reg	1 1 1 xsreg reg	mod 0 sreg mem	mod 1 xsreg mem	mod reg mem	mod reg mem	00111110		00110110	
Operation Code	76543210	1000101W	1000100W	1000101W	1100011W	1011W reg	101000W	101001W	10001100	1000110	10001100	10001110	1001100	1001100	10001100	1001100	11000101	11000100	00001111	mod reg mem	0001111	
	Operand(s)	reg, reg*	mem, reg	reg, mem	mem, imm	reg, imm	acc, dmem	dmem, acc	sreg, reg16	xsreg, reg16*	sreg, mem16	xsreg, mem16*	reg16, sreg	reg16, xsreg*	mem16, sreg	mem16, xsreg*	DS0, reg16, mem33	DS1, reg16, mem32	DS2 real6*	mem32	DS3 rea16*	Uos, leg lo',
1	Mnemonic Mnemonic	1							ŞI	noitoi		sfer ⊗ V	nert e	c)e()								

📟 6427525 OO63211 O56 📖

17.3 INSTRUCTION SET TABLE

NEC



105

dr			Operation Code	n Code				Œ	Flags		
ותפנרעכ Grou	Mnemonic	Operand(s)	76543210	76543210	Bytes	Operation	AC CY	>	٩	S	N
<u> </u>		AH, PSW	10011111		-	AH←S, Z, F1, AC, F0, P, IBRK, CY					
	200W	PSW, AH	10011110		-	S, Z, F1, AC, F0, P, IBRK, CY←AH	×		×	×	×
SU.	LDEA	reg16, mem16	10001101	mod reg mem	2 to 4	reg16←mem16					
	TRANS TRANSB*1	src-table	1101011		-	AL←(BW + AL)					
neni		reg, reg'	100001 W	1 1 reg reg'	2	reg↔reg'					
refer	хсн	mem, reg reg, mem	10001 W	mod reg mem	2 to 4	(mem)↔reg					
isit et	*	AW, reg16 reg16, AW	10010 reg		-	AW↔reg16					
۵	MOVSPA*2		00001111	00100101	7	New register bank SS, SP \leftarrow Old register bank SS, SP					
•		1001	00001111	10010101	~	SS, SP of register bank indicated by reg16 \leftarrow					
		61631	11111 reg		,	current register bank SS, SP					
	REPC		01100101		-	While CW \neq 0, the following byte primitive block transfer instruction is executed and CW is decremented (-1). If there is a pending interrupt, it is serviced. If CY \neq 1, the loop is exited.					
səxite	REPNC		01100100		-	Same as above. If CY ≠ 0, the loop is exited.					
Repeat pre	REP REPE REPZ		11110011			While CW \neq 0, the following byte primitive block transfer instruction is executed and CW is decremented (-1). If there is a pending interupt, it is serviced. If the primitive block transfer instruction is CMPBK or CMPM, and Z \neq 1, the loop is exited.					
•	REPNE REPNZ		11110010		-	Same as above. If Z ≠ 0, the loop is exited.					

106

6427525 OO63212 T92 🖿

Powered by ICminer.com Electronic-Library Service CopyRight 2003

	Operation Code	n Code	Rutes	Operation	Flags	
	76543210	76543210	oyles		AC CY V P S	N
				If W = 0, (IY)←(IX) DIR = 0: IX←IX + 1, IY←IY + 1 DIR = 1: IX←IX - 1, IY←IY - 1		····
	10100100 W0100			If W = 1, (IY + 1, IY)←(IX + 1, IX) DIR = 0: IX←IX + 2, IY←IY + 2 DIR = 1: IX←IX - 2, IY←IY - 2		1
				If W = 0, (IX) - (IY) DIR = 0: IX←IX + 1, IY←IY + 1 DIR = 1: IX←IX - 1, IY←IY - 1	>	>
	W1 1 0 0 1 0 1 W1 1 0 0 1 0 1		-	If W = 1, (IX + 1, IX) – (IY + 1, IY) DIR = 0: IX←IX + 2, IY←IY + 2 DIR = 1: IX←IX – 2, IY←IY – 2	(<
				If W = 0, AL - (IY) DIR = 0: IY←IY + 1 ; DIR = 1: IY←IY - 1	>	>
	W1 1 1 0 1 0 1 W1 1 1 0 1 0 1			If W = 1, AW − (IY + 1, IY) DIR = 0: IY←IY + 2 ; DIR = 1: IY←IY − 2	((((
				If W = 0, AL←(IX) DIR = 0: IX←IX + 1 ; DIR = 1: IX←IX - 1		
	1010110W		-	If W = 1, AW + (IX + 1, IX) DIR = 0: IX + 2 ; DIR = 1: IX←IX - 2		
1				ff W = 0, (IY)←AL DIR = 0: IY←IY + 1 ; DIR = 1: IY←IY - 1		· · · · · ·
	W1010101		_	If W = 1, AW − (IY + 1, IY)←AW DIR = 0: IY←IY + 2 ; DIR = 1: IY←IY − 2		

🗯 6427525 0063213 929 🗰

Oct Matematicana 7 6 5 4 3 2 10 7 6 5 16 10 10 7 6 5 16 10 10 7 6 5 16 10 10 7 6 5 10 10 10 11 1 1 1 10 11 10 1 1 1 10 11 10 1 1 1 10 11 10 1 1 1 10 11 10 1 1 1 10 11 10 1 1 1 10 11 10 1 1 1 10 11 10 1 1 1 10 11 10 1 1 1 10 11 10 1 1 1 10 10 1 1 1 10 11 10 1 1 1 10 11 10 1 1 1 10 11 10 1 1 1 10 11 10 1 1 1 10 11 10 1 1 1 10 11 10 1 1 1 1 10 1 1 1 10 1 1 1 1 10	nction UUD			Operation Code	n Code			Flags	
INS 0 0 0 0 0 11 11 0 0 11 0 0 1 3 reg8. reg8 11 reg° reg 11 reg° 1 INS 0 0 0 0 11 11 0 0 11 10 0 1 4 reg8. imm4 1 1 0 0 0 reg 0 0 0 11 11 0 0 11 10 0 1 INS reg8. imm4 1 1 0 0 0 reg 0 0 11 10 0 1 4 reg8. imm4 1 1 0 0 0 reg 1 1 reg° reg 2 INS reg8. imm4 1 1 0 0 0 reg 1 1 reg° 2 Instructions 0 0 0 0 11 11 0 0 11 10 11 3 reg8. imm4 1 1 0 0 0 reg 1 1 1 0 0 1 reg 2 Imm8. acc. imm8 1 1 1 0 1 1 0 W 2 2 Imm8. acc. imm8 1 1 1 0 1 1 W 0 1 1 0 1 1 2 Imm8. acc 1 1 1 0 1 1 0 W 2 2 2 Imm8. acc 1 1 1 0 1 1 0 W 2 2 2 2 Imm8. acc. imm8 1 1 1 0 1 1 1 W 2 2 2 2 Imm8. acc. imm8 1 1 1 0 1 1 1 W 2 2 <td< td=""><td>untent Dið</td><td>Mnemonic</td><td>Uperand(s)</td><td>6543</td><td>65432</td><td>Dyles</td><td>Cperatori</td><td>V P S</td><td></td></td<>	untent Dið	Mnemonic	Uperand(s)	6543	65432	Dyles	Cperatori	V P S	
rego. rego 11 reg. reg reg <threg< th=""> reg <threg< th=""></threg<></threg<>				000	011000	ſ			
Instructions 0 0 0 0 11 1 1 0 0 11 1 0 0 1 4 reg8. imm4 1 1 0 0 0 reg 0 0 1 1 0 0 1 1 4 reg8. imm4 0 0 0 0 1 1 1 1 0 0 1 1 0 0 1 1 4 reg8. imm4 0 0 0 0 1 1 1 1 0 0 1 1 0 0 1 1 3 reg8. imm4 1 1 0 0 0 reg 0 0 0 1 1 1 0 1 1 4 reg8. imm4 1 1 0 0 1 reg 0 0 1 1 1 0 1 1 4 reg8. imm8 1 1 1 0 0 1 reg 2 2 lmstructions acc. imm8 1 1 1 0 0 1 row 2 2 Jum9. acc. Dw 1 1 1 0 1 1 w 1 1 2 2 Jum8. acc 1 1 1 0 1 1 w 2 2 2 Jum8. acc 1 1 0 1 1 w 2 2 2 Jum8. acc 1 1 0 1 1 w 1 1 1 2 2 Jum8. acc 1 1 0 1 1 w 2 2 2 2 2 Jum8. acc 1 1 0 1 1 1 w 2 2 2 2 2 2 2 2 2 2		9	regă, regă	1 regʻ		n			
regg. imma 11000 reg 0011011 3 regg. imma 11 reg' reg 00111011 3 regg. imma 11 reg' reg 00111011 3 regg. imma 11000 reg 11 reg' reg 3 regg. imma 11000 reg 11000 reg 3 regg. imma 11000 reg 11000 reg 3 regg. imma 111000 reg 11000 reg 3 regg. imma 111000 reg 3 3 output instructions 3	noite			000	011100				
Instructions 0011011 00110011 3 FXT reg8, reg8 11 reg' eg 11 reg' 3 FXT reg8, reg8 11 reg' 00111011 3 Instructions acc, imm8 111000 reg 2 2 IN* acc, imm8 111001 w 2 2 Inm0. buv, acc 111011 w 2 2 Imm8. acc DVV acc 111011 w 2 2 Imm8. acc DVV acc 111011 w 2 2 Imm8. acc DVV 0 1 2 2	luqin enoit	611.01	reg8, imm4	000		4			
FXT rege. reg 11 reg reg	em b struc		ā	000	011001	· · · ·			
FXI 00001111 00111011 reg8, imm4 11000 reg 11000 reg 4 acc. imm8 1110010W 2 2 imstructions imm8, acc. DW 1110011W 2 2 DUT+ bw, acc. DW 1110011W 2 2 1 DUT+ bw, acc 1110011W 2 2 1 DW. acc 111011W 111011W 2 2 1 DW. acc 111011W 111011W 1 1 1 1 Src-block 011011W 0110W 1 1 1 1 1	leit ti ni		rega, rega	-		n			
regg. imma 11000 reg acc. imm8 1110010W acc. imm8 1110010W acc. imm8 1110110W acc. imm8, acc 11100110W imm8, acc 1110110W bW, acc 1110111W	8	EXI		000111	011101	•			
Acc. imm8 1 1 1 0 0 1 0 W 2 acc. imm8 1 1 1 0 0 1 0 W 2 IN* acc. DW 1 1 1 0 1 1 0 W 2 Imm8, acc. DW 1 1 1 0 1 1 1 W 1 1 OUT* DW, acc 1 1 1 0 1 1 1 W 1 1 Imm8, acc. DW 1 1 1 0 1 1 1 W 1 1 2 OUT* DW, acc 1 1 1 0 1 1 1 W 1 1 1 DW DW, acc 1 1 1 0 1 1 1 W 1 1 1 1 DW DW, acc 1 1 1 0 1 1 1 W 1			regs, imm4	1000		4			
acc. DW 1 1 1 0 1 1 0 W acc. DW 1 1 1 0 1 1 0 W imm8, acc 1 1 1 0 1 1 W DW, acc 1 1 1 0 1 1 W DW, acc 1 1 1 0 1 1 W DW, acc 1 1 1 0 1 1 W DW, acc 1 1 1 0 1 1 W DW, acc 1 1 1 0 1 1 W DW, acc 1 1 1 0 1 1 W DW, acc 1 1 1 0 1 1 W DW 0 1 1 0 1 1 W DW 0 1 1 0 1 1 W			acc, imm8	0 7		7	If W = 0, AL←(imm8) If W = 1, AH←(imm8 + 1), AL←(imm8)		
Imm8, acc 1 1 1 0 0 1 1 W 2 OUT * Imm8, acc 1 1 1 0 0 1 1 W 2 OUT * DW, acc 1 1 1 0 1 1 1 W 1 Ast-block 0 1 1 0 1 1 0 W 1 1 DW acc 1 1 1 0 1 1 1 W 1 1 DW acc 1 1 1 0 1 1 1 W 1 1 1 DW DW 0 1 1 0 1 1 0 W 1 1 1 1 DW DW 0 1 1 0 1 1 0 W 1	tuqtu enoit:		acc, DW	10		-	If W = 0, AL←(DW) If W = 1, AH←(DW + 1), AL←(DW)		
DW, acc 1 1 1 0 1 1 1 W DW, acc 1 1 1 0 1 1 1 W dst-block, 0 1 1 0 1 1 0 W DW, acc 1 1 1 0 1 1 W DW, acc 1 1 1 0 1 1 1 W 1 1 2 1 0 1 1 0 W 1 2 1 0 1 1 0 W 1 1 0 1 1 1 W 1 2 1 1 1 1 1 W 1 2 1 1 1 1 1 1 W 1 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	o\tuqnl wrteni	1	imm8, acc	110		5	If W = 0, (imm8)←AL If W = 1, (imm8 + 1)←AH, (imm8)←AL		
dst-block, 0 1 1 0 1 1 0 W DW DW DW, 0 1 1 0 W src-block 0 1 1 0 1 1 1 W src-block 0 1 1 0 1 1 1 W		•	DW, acc			-	If W = 0, (DW)←AL If W = 1, (DW + 1)←AH, (DW)←AL		
DV DV DV DV DV DV DV DV DV DV DV DV DV D			dst-block,			•	If W = 0, (IY)←(DW) DIR = 0: IY←IY + 1 ; DIR = 1: IY←IY - 1		
output in DW, src-block 0 1 1 0 1 1 1 W	tuqni e voitointe		MQ			-	If W = 1, (IY + 1, IY)←(DW + 1, DW) DIR = 0: IY←IY + 2 ; DIR = 1: IY←IY - 2		
src-block of the starbulack	Vitimitiv utotu		DW,			•	If W = 0, (DW)←(IX) DIR = 0: IX←IX + 1 ; DIR = 1: IX←IX - 1		
			src-block	-		-	If W = 1, (DW + 1, DW)←(IX + 1, IX) DIR = 0: IX←IX + 2 ; DIR = 1: IX←IX - 2		

108

🔲 6427525 0063214 865 📟

Powered by ICminer.com Electronic-Library Service CopyRight 2003

_t	N	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×
- H	s	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×
Flags	۵.	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×
Ē	>	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×
	2	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×
	AC	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×
	Cheratical	reg←reg + reg'	(mem)←(mem) + reg	reg←reg + (mem)	reg←reg + imm	(mem)←(mem) + imm	If W = 0, AL←AL + imm If W = 1, AW←AW + imm	reg←reg + reg' + CY	(mem)←(mem) + reg + CY	reg←reg + (mem) + CY	reg←reg + imm + CY	(mem)←(mem) + imm + CY	If W = 0, AL←AL + imm + CY If W = 1, AW←AW + imm + CY	reg←reg - reg'	(mem)←(mem) – reg	regt⊷reg – (mem)	reg←reg – imm	(mem)←(mem) ~ imm	If W = 0, AL←AL – imm If W = 1, AW←AW – imm	reg←reg – reg' – CY	(mem)←(mem) – reg – CY	reg←reg – (mem) – CY	reg←reg – imm – CY	(mem)←(mem) – imm – CY	If W = 0, AL←AL – imm – CY If W = 1, AW←AW – imm – CY
	bytes	2	2 to 4	2 to 4	3 to 4	3 to 6	2 to 3	2	2 to 4	2 to 4	3 to 4	3 to 6	2 to 3	2	2 to 4	2 to 4	3 to 4	3 to 6	2 to 3	2	2 to 4	2 to 4	3 to 4	3 to 6	2 to 3
Code	76543210	11 reg reg'	mod reg mem	mod reg mem	11000 reg	mod 0 0 0 mem		1 reg reg'	mod reg mem	mod reg mem	11010 reg	mod 0 1 0 mem		1 reg reg'	mod reg mem	mod reg mem	11101 reg	mod 1 0 1 mem		11 reg reg'	mod reg mem	mod reg mem	11011 reg	mod 0 1 1 mem	
				-	-	Ĕ		-		-	-	Ĕ		-				-							
Operation Code	76543210	000001W	000000M	000001W	10000sW 1	100000 SW m	00001000	0001001W 1	000100W r	0 0 1 0 0 1 W	100000 SW 1	10000sW m	0001010M	0010101W 1	0010100M	0010101W	10000 s W	100005W	0010100	0001101W	0001100W	0 0 0 1 1 0 1 W	100000 SW	100000 s W	0001110W
	654321	00000	00000	00001W	0 0 0 s W	0 0 0 s W	0001	001001W	0 0 1 0 0 W	001001W	10000 s W	10000sW	00101	0 0 1 0 1 0 1 W	0010100 E	010101W	10000sW	10000sW	001011	0 0 1 1 0 1 W	0 0 1 1 0 0 W	0 0 1 1 0 1 W	10000 s W	n 100000sW	001
	7654321		000000	0 0 0 0 0 0 1 W	000001W 0000000W 000000W	0000001W 0000000W 0000001W 1000001W	0000001W 0000000W 0000001W 100000sW	0 0 0 0 0 0 0 1 W 0 0 0 0 0 0 0 0 W 1 0 0 0 0 0 0 1 W 1 0 0 0 0 0 8 W 0 0 0 0 0 1 0 W	0000001W 00000000W 1000001W 1000003W 1000003W 0000010W	0000001W 0000000W 1000003W 1000003W 000003W 00010010W	0000001W 0000000W 1000003W 1000003W 000003W 00010010W 0001001W	0000001W 0000000W 0000001W 1000003W 1000003W 0000003W 0001001W 0001001W 1000100W	0 0 0 0 0 0 0 1 W 0 0 0 0 0 0 0 0 W 1 0 0 0 0 0 0 1 W 1 0 0 0 0 0 0 8 W 0 0 0 0 0 0 0 1 0 W 0 0 0 1 0 0 1 W 1 0 0 0 1 0 0 W 1 0 0 0 1 0 0 1 W 1 0 0 0 0 8 W	0000001W 0000000W 1000000W 1000000% 1000000% 0001001W 0001001W 0001001W 1000000%	00000000000000000000000000000000000000	0000001W 0000000W 1000000W 1000000W 1000000W 0001001W 0001001W 1000000W 1000000W 0001001W 0001001W 0001001W 0001001W	0000001W 0000001W 1000008W 1000008W 0000008W 0001001W 0001001W 1000008W 1000008W 1000008W 1000008W 0001001W 0001001W 00010101W	0000001W 0000000W 1000000W 1000000W 1000000W 0001001W 0001001W 1000000W 1000000W 0001001W 0001001W 00010101W 00010101W	0000001W 0000001W 1000008W 1000008W 1000008W 0001001W 0001001W 10000100W 0001001W 0001001W 0001001W 0001001W 1000008W 1000008W 1000008W	0000001W 0000000W 1000000W 1000000W 1000000W 0001001W 0001001W 1000000W 1000000W 1000000W 00010100W 0001001W 1000000W 0010101W 0010101W 00010100W	0000001W 0000001W 1000000W 10000008W 1000008W 0001001W 0001001W 0001001W 0001001W 0001001W 0001001W 1000008W 1000008W 1000008W 1000008W 1000008W	0000001W 0000000W 1000000W 1000000W 1000000W 0001001W 0001001W 1000000W 1000000W 0001001W 0001001W 0001000W 1000000W 1000000W 1000000W 0010101W 00010100W	0000001W 0000001W 1000000W 10000008W 1000008W 1000008W 0001001W 0001001W 1000008W 1000008W 1000008W 1000008W 1000008W 1000008W 1000008W 1000008W 1000008W 1000008W 00110100W	0 0	0000001W 0000000W 1000000W 1000000W 1000000W 0001001W 0001001W 1000000W 1000000W 1000000W 00010101W 00010101W 0001000W 0001000W 0001000W 0001101W 1000000W 1000000W 1000000W 1000000W

🖬 6427525 0063215 7T1 📰

L												Г
noita			Operation Code	n Code					Flags			
urteni Gro	Mnemonic	Operand(s)	76543210	76543210	Bytes	Operation	ک ۲	र ठ	<u>م</u> >	s s		N
	ADD4S+1	(dst-string, src-string)	00001111	0010000	2	dst BCD string←dst BCD string + src BCD string*2	5	×	2 2			×
	SUB4S*1	(dst-string, src-string)	00001111	0010010	2	dst BCD string←dst BCD string – src BCD string•2	þ	×	<u>ר</u>	n n		×
su	CMP4S*1	(dst-string, src-string)	00001111	00100110	2	dst BCD string – src BCD string*2	2	×	ר ה	n n		×
ructio		reg8	00001111	00101000	3							
tsni			11000 reg									
noiter	ROL4	mem8	00001111	0010100	3 to 5	mem						
ədo			mod 0 0 0 mem			1 1000						
BCD		reg8	00001111	00101010	m							
			11000 reg									
	ROH4	mem8	00001111	00101010	3 to 5							
			mod 0 0 mem									
n		reg8	1111110	11000 reg	2	reg8←reg8 + 1	×		×	×	×	×
າອເມຊ	NC	mem	W111111W	mod 0 0 mem	2 to 4	(mem)←(mem) + 1	×		Ŷ	×	×	×
aroat	UOIX	reg16	01000 reg		1	reg16←reg16 + 1	×		×	×	×	×
o/tua	2truc	reg8	1111110	11001 reg	2	reg8←reg8 – 1	×		$\hat{\mathbf{x}}$	÷	×	×
ເພື່ອມ	E DEC	mem	1111111W	mod 0 0 1 mem	2 to 4	(mem)←(mem) – 1	×		×	×	×	×
oul		reg16	01001 reg		1	reg16←reg16 – 1	×		÷	Ŷ	×	×
*	1. The open 2. The num	The operand can be omitted. The number of BCD digits is	The operand can be omitted. The number of BCD digits is given by the CL register: a value between 1 and 254 can be set.	∋gister: a value betwe	sen 1 a	nd 254 can be set.						

110

6427525 0063216 638 🔳

	Operation Code	n Code				Ē	Flags	ł
Operand(s)	76543210	76543210	Bytes	Operation	AC	> ሪ	۵.	2 S
reg8	11110110	11100 reg	2	AW←AL × reg8 AH = 0: CY←0, V←0 AH ≠ 0: CY←1, V←1	D	× ×	Þ	> >
mem8	1110110	mod 1 0 0 mem	2 to 4	AW←AL × (mem8) AH = 0: CY←0, V←0 AH ≠ 0: CY←1, V←1	5	× ×	Э	ר ר
reg16	1110111	1 1 1 0 0 reg	2	DW, AW←AW × reg16 DW = 0: CY←0, V←0 DW ≠ 0: CY←1, V←1	5	×	Þ	ר ר
mem16	1110111	mod 1 0 0 mem	2 to 4	DW, AW←AW × (mem16) DW = 0: CY←0, V←0 DW ≠ 0: CY←1, V←1	2	× ×	Э	ר ר
reg8	1110110	11101 reg	2	AW←AL × reg8 AH = AL sign extension: CY←0, V←0 AH ≠ AL sign extension: CY←1, V←1	<u> </u>	×	Э	> >
тет8	11110110	mod 1 0 1 mem	2 to 4	AW←AL × (mem8) AH = AL sign extension: CY←0, V←0 AH ≠ AL sign extension: CY←1, V←1		×	Э	כ כ
reg16	1110111	11101 reg	2	DW, AW←AW × reg16 DW = AW sign extension: CY←0, V←0 DW ≠ AW sign extension: CY←1, V←1	>	× ×	<u> </u>	> >
mem16	1110111	mod101 mem	2 to 4	DW, AW←AW × (mem16) DW = AW sign extension: CY←0, V←0 DW ≠ AW sign extension: CY←1, V←1		×	2	כ כ
reg16, (reg16',)* imm8	01101011	11 reg regʻ	3	reg16←reg16' × imm8 Product ≤ 16 bits: CY←0, V←0 Product > 16 bits: CY←1, V←1	5	×	5	
reg16, mem16, imm8	01101011	mod reg mem	3 to 5	reg16←(mem16) × imm8 Product ≤ 16 bits: CY←0, V←0 Product > 16 bits: CY←1, V←1		×	5	> >
reg16, (reg16',)* imm16	0110101	11 reg reg'	4	reg16←reg16' x imm16 Product ≤ 16 bits: CY←0, V←0 Product > 16 bits: CY←1, V←1	5	×	- >	ר ר ר
reg16, mem16, imm16	01101001	mod reg mem	4 to 6	reg16←(mem16) × imm16 Product ≤ 16 bits: CY←0, V←0 Product > 16 bits: CY←1, V←1	n n	×	- 0	n n

🖬 6427525 OO63217 574 🔳

μ**PD70433**

	N	>	5	Ð	Э
	S			n	ר ה
5	٩.	ב ר ר ר ר ר	> > > > > > >	ר ר ר ר ר ר ר ר ר ר	ר ר ר ר ר ר
Flags			 	2	
	AC CY V		 	 	
}	2				
	<				
	Operation	temp←AW If temp + reg8 ≤ FFH AH←temp%reg8, AL←temp + reg8 If temp + reg8 > FFH (SP - 1, SP - 2)←PSW, (SP - 3, SP - 4)←PS (SP - 5, SP - 6)←PC, SP←SP - 6 IE←0, BRK←0, PS←(3, 2), PC←(1, 0)	temp←AW If temp + (mem8) ≤ FFH AH←temp%(mem8), AL←temp + (mem8) If temp + (mem8) > FFH (SP - 1, SP - 2)←PSW, (SP - 3, SP - 4)←PS (SP - 5, SP - 6)←PC, SP←SP - 6 IE←0, BRK←0, PS←(3, 2), PC←(1, 0)	temp←DW, AW If temp + reg16 ≤ FFFFH DW←temp%reg16, AW←temp + reg16 If temp + reg16 > FFFFH (SP - 1, SP - 2)←PSW, (SP - 3, SP - 4)←PS (SP - 5, SP - 6)←PC, SP←SP - 6 IE ←0, BRK←0, PS←(3, 2), PC←(1, 0)	temp←DW, AW If temp + (mem16) ≤ FFFFH DW←temp%(mem16), AW←temp + (mem16) If temp + (mem16) > FFFFH (SP - 1, SP - 2)←PSW, (SP - 3, SP - 4)←PS (SP - 5, SP - 6)←PC, SP←SP - 6 IE←0, BRK←0, PS←0, PS←1, 0)
	bytes	2	2 to 4	2	2 to 4
n Code	76543210	11110 reg	mod 1 1 0 mem	11110 reg	mod 1 1 0 mem
Operation Code	76543210	1 1 1 0 1 1 0	1110110	1110111	11101111
	Uperand(s)	80 92	8 Hereway	reg16	mem 16
dno	E Mnemonic		anoitanterions		

🖬 6427525 00632**1**8 400 🔜

Powered by ICminer.com Electronic-Library Service CopyRight 2003

	Ν	D	Þ	5	>
	s	D C	2	5	5
Flags	۵.		5	5	2
E	>	5	D	5	>
	5	5	5	5	5
	AC AC	5	D	⊃	⊃
	Operation	tempt-AW If temp + reg8 > 0 and temp + reg8 ≤ 7FH; or if temp + reg8 < 0 and temp + reg8 > 0 - 7FH - 1 AH-temp%reg8, AL-temp + reg8 > 7FH; or If temp + reg8 < 0 and temp + reg8 > 0 - 7FH - 1 (SP + 1, SP - 2)(-PSW, (SP - 3, SP - 4)(-PS (SP - 5, SP - 6)(-PC, SP(-SP - 6) IE (-0, BRK+0, PS(-(3, 2), PC(-(1, 0)))	$\begin{split} temp \leftarrow AW \\ If temp + (mem8) > 0 and temp + (mem8) \leq 7FH; or \\ if temp + (mem8) < 0 and temp + (mem8) > 0 - 7FH - 1 \\ AH \leftarrow temp % (mem8), AL \leftarrow temp + (mem8) > 7FH; or \\ If temp + (mem8) > 0 and temp + (mem8) > 7FH; or \\ if temp + (mem8) < 0 and temp + (mem8) > 7FH; or \\ SP - 1, SP - 2) \leftarrow PSW, (SP - 3, SP - 4) \leftarrow PS \\ (SP - 5, SP - 6) \leftarrow PC, SP \leftarrow SP - 6 \\ IE \leftarrow 0, BRK \leftarrow 0, PS \leftarrow (3, 2), PC \leftarrow (1, 0) \end{split}$	tempt-DW, AW If temp + reg16 > 0 and temp + reg16 \leq 7FFFH; or if temp + reg16 > 0 and temp + reg16 > 0 - 7FFH - 1 DW(-temp%reg16, AW(-temp + reg16 > 7FFFH; or If temp + reg16 > 0 and temp + reg16 \leq 0 - 7FFH - 1 (SP - 1, SP - 2)(-PSW, (SP - 3, SP - 4)(-PS (SP - 5, SP - 6)(-PC, SP(-SP - 6)(-1, 0)) E(-0, BRK(-0), PS(-(3, 2), PC(-(1, 0)))	$\begin{split} temp\leftarrowDW, AW \\ If \; temp + (mem16) > 0 \; and \; temp + (mem16) \leq 7FFFH, or \\ if \; temp + (mem16) < 0 \; and \; temp + (mem16) > 0 - 7FFH - 1 \\ AH\leftarrowtemp8(nem16) < 0 \; and \; temp + (mem16) > 0 - 7FFH - 1 \\ AH\leftarrowtemp8(nem16) < 0 \; and \; temp + (mem16) > 0 - 7FFH - 1 \\ If \; temp + (mem16) > 0 \; and \; temp + (mem16) > 7FFH : or \\ If \; temp + (mem16) < 0 \; and \; temp + (mem16) > 7FFH : or \\ SP - 1, \; SP - 2)\leftarrowPSW, (SP - 3, \; SP - 4)\leftarrowPS \\ SP - 5, \; SP - 6)\leftarrowPC, \; SP\leftarrowSP - 6 \\ IE\leftarrow0, \; BRK\leftarrow0, \; PS\leftarrow(3, 2), \; PC\leftarrow(1, 0) \end{split}$
	Bytes	7	2 to 4	м	2 to 4
n Code	76543210	1 1 1 1 reg	mod 1 1 mem	1 1 1 1 reg	mod 1 1 mem
Operation Code	76543210			1 1 1 0 1 1 1	1 1 0 1 1 1
	Operand(s)	reg 8	ж Э	reg 16	лет 16
dn	Gro Mnemonic			oisivib bangi2	
Laoita	Instruc	I			

Memoric Metalitying ADJBA 7 6 5 4 3 2 1 0 7 8 × 1 U U U U U U U U U U U U U U U U U U U U V × × × V U V × × × V U V × × × V V × × × V × × × V × × × V × × × × × U U U V × × × U V V × × × V × × × × V × × × × V × × × × × × V × × × × ×	up tion			Operation Code	on Code					Flags	2		_
ADJBA D 0 0 11 0 11 1 1 I.A.L. OH+ S OF ACC = 1: ALI-AL, OH+ X × U V X V X V X V V X V V X V V X V V X V V X V V X V V X V V X V V X V V X V V X V V X V V X V V X V V X V V X V V X V V X X V V X X V V X X U U U V X X U U V X X U U U X X U U U U U X X U U U U U U U	Instru Gro		Operand(s)	654321	654321	Bytes	Operation	AC		1			
ADIA 00100111 Int. Service - 1: Int. Service	suc	ADJBA		011011		٢	If AL ∧ 0FH > 9 or AC = 1: AL←AL + 6 AH←AH + 1, AC←1, CY←AC, AL←AL ∧ 0FH	×		·		+	.
ADIBS Definition Introduct Intrut Intrut Intrut<	oitounteni tr	ADJ4A		0100		ŀ		×	····				
ADJ4S 0 0 10 11 11 1 If AL, OFH > 9 or AC=1; AL=AL-6, CY-VAC, AC+1 × × V × × V × × V × × V × × V × × × V × × × V × × V × × × V ×<	iəmteu įbi	ADJBS		011111		-	If AL ∧ 0FH > 9 or AC = 1: AL←AL - 6, AC←1 CY←AC, AL←AL ∧ 0FH	×	<u> </u>			<u></u>	
CVTBD 1101010 0001010 2 AH-AH+0AH, AL-AL%0AH U U U V × CVTBD i 11010101 00001010 2 AL-AH*0AH AL.AH+0 U U U V × CVTBW i 10011001 2 AL-AH*0AH AL.AH+0 U U U V ×	6 O 38	AD14S		010111		-	If AL ∧ 0FH > 9 or AC = 1: AL←AL - 6, CY←CY ∨ AC, AC←1 If AL > 9FH or CY = 1: AL←AL - 60H, CY←1	×			f		
CVTDB 11010101 00001010 2 AL-AH × 0AH + AL, AH-O U U U V X CVTDB 10011000 10011000 1 I f AL < 80H: AH-O, otherwise: AH-FFH		CVTBD		101010	000101	2	AH←AH + 0AH, AL←AL%0AH	Э			<u> </u>		
ECVTBW 1001100 11 14 L 80Hi AH-0, otherwise: AH-FH 1 <th1< td="" th<=""><td></td><td>CVTDB</td><td></td><td>1010</td><td>000101</td><td>2</td><td>AL←AH × 0AH + AL, AH←0</td><td>Э</td><td><u>+</u></td><td></td><td><u> </u></td><td>-</td><td></td></th1<>		CVTDB		1010	000101	2	AL←AH × 0AH + AL, AH←0	Э	<u>+</u>		<u> </u>	-	
CVTWL 10011001 11 reg reg' 1 If AW < 8000H: DW-G, otherwise: DW-FFFFH 1 I I I <thi i<="" th=""> I I</thi>		CVTBW		01100		-	If AL < 80H: AH←0, otherwise: AH←FFH		<u> </u>				
reg, reg, reg, reg, reg, reg, reg, reg,		CVTWL		001100		-	If AW < 8000H: DW←0, otherwise: DW←FFFH						
mem.reg 0011100W mod reg mem 2104 (mem)-reg x			reg, reg'	0111	reg	2	reg - reg'	×	ł —	<u> </u>		<u> </u>	
reg. mem 0011101W mod reg mem 2 to 4 reg. (mem) x x <th< td=""><td></td><td></td><td>mem, reg</td><td>01110</td><td>reg</td><td>2 to 4</td><td>(mem)- reg</td><td>×</td><td><u> </u></td><td><u>{</u></td><td></td><td><u> </u></td><td></td></th<>			mem, reg	01110	reg	2 to 4	(mem)- reg	×	<u> </u>	<u>{</u>		<u> </u>	
CMP reg. imm 100000sW 11111 reg 3to 4 reg. imm x	rosit noit:		reg, mem	01110	reg	2 to 4	reg - (mem)	×	<u> </u>		ļ		r
Image: mem,imm 100000sW mod111mem 3to6 (mem)-imm x	եզm 20112	CMP	reg, imm	00000	-	5	reg - imm	×			<u> </u>		1
acc.imm 0 0 1 1 1 1 0 W 2 to 3 If W = 0, AL - imm x	o) ni		mem, imm	0000	mod 1 1 1 mem	2	(mem) – imm	×					
NOT reg 1111011W 11010 reg 2 reg-reg No mem 1111011W mod 010 mem 2 to 4 (mem) - (mem) r </td <td></td> <td></td> <td>acc, imm</td> <td>01111</td> <td></td> <td>9</td> <td>If W = 0, AL - imm If W = 1, AW - imm</td> <td>×</td> <td></td> <td></td> <td></td> <td></td> <td></td>			acc, imm	01111		9	If W = 0, AL - imm If W = 1, AW - imm	×					
mem 1111011W mod010mem 2 to 4 (mem) - (mem) mem N		NOT	Gəı	11101	1010	2	<u>5a</u> →5aı			<u> </u>			
reg 1111011W 11011 reg 2 reg←reg + 1 ×	*		mem	1111011W	-	to	(mem) - (<u>mem</u>)						
mem 1111011W mod 011mem 2 to 4 (mem)←(mem) + 1 x x x x x x x x x x x x x x x x x x		NFG	reg	111	011	2	reg (~reg + 1	×					
		2	mem	111		2 to 4	(mem)←(mem) + 1	×		<u> </u>	Į		

🖬 6427525 OO63220 O69 🔳

Powered by ICminer.com Electronic-Library Service CopyRight 2003

Flags	AC CY V P S	U 0 0 × ×	N 0 0 X	× × 0	0 × ×	× ×	× ×	×	×	×	×	×	×	T		T	×	×	×	×	×	×	×	×
Flag	cy ۷	0 0	0	0		×	i x l						\rightarrow	×	×	×		<u> </u>						
	Σ	0	0		0			_×	<u>×</u>	×	×	×	×	×	×	×	×	×	×	×	×	×	×	
	ACC					•	0	-	•	•	•	•	-	•	•	<u> </u>	-	-	-	-	•	0	•	•
	A	-		0 0	u o	0)	о Э	。)	0 2	0 0	о Л	• >	<u> </u>	0 0	0 つ	0 0	• >	о Л	。 万	~	-	0	<u> </u>	<u> </u>
							_	_		-	-		_	_	_		_		_	2	2	<u> </u>	-	<u> </u>
	Operation	reg ∧ reg'	(mem) ∧ reg	reg ~ imm	(mem) ^ imm	lf W = 0, AL ~ imm8 lf W = 1, AW ~ imm16	reg←reg ∧ reg'	(mem)←{mem} ∧ reg	reg←reg ∧ (mem)	reg←reg ∧ imm	(mem)←(mem) ∧ imm	If W = 0, AL←AL ^ imm8 If W = 1, AW←AW ^ imm16	reg←reg v reg'	(mem)←(mem) ∨ reg	reg←reg ∨ (mem)	reg←reg ∨ imm	(mem)←(mem) ~ imm	lf W = 0, AL←AL ∨ imm8 lf W = 1, AW←AW ∨ imm16	reg←reg v reg'	(mem)←(mem) v reg'	reg←reg v (mem)	reg←reg ヤ imm	(mem)←(mem)	If W = 0, AL←AL v imm8 If W = 1, AW←AW v imm16
	Bytes	5	2 to 4	3 to 4	3 to 6	2 to 3	2	2 to 4	2 to 4	3 to 4	3 to 6	2 to 3	2	2 to 4	2 to 4	3 to 4	3 to 6	2 to 3	2	2 to 4	2 to 4	3 to 4	3 to 6	2 to 3
n Code	76543210	11 reg' reg	mod reg mem	11000 reg	mod 0 0 0 mem		11 reg reg'	mod reg mem	mod reg mem	11100 reg	mod 1 0 0 mem		11 reg reg'	mod reg mem	mod reg mem	11001 reg	mod 0 0 1 mem		11 reg reg'	mod reg mem	mod reg mem	11110 reg	mod 1 1 0 mem	
Operation Code	76543210	100010W	100010W	W1101111	W1101111	101010W	001001W	001000W	001001W	100000W	100000W	00100100	0000101W	00001000	0000101W	100000W	100000W	0000110M	0011001W	001100W	0011001W	100000W	100000W	0 0 1 1 0 1 0 W
	Operand(s)	reg, reg	mem, reg reg, mem	reg, imm	mem, imm	acc, imm	reg, reg'	mem, reg	reg, mem	reg, imm	mem, imm	acc, imm	reg, reg'	mem, reg	reg, mem	reg, imm	mem, imm	acc, imm	reg, reg'	mem, reg	reg, mem	reg, imm	теп, ітп	acc, imm
dr	Mnemonic		- 4	TEST	<u> </u>				(AND		instruc				е О	·	L		i	1	HOX	•	± <u></u>

🔲 6427525 0063221 TT5 🔜

	N	×	×	×	×	×	×	×	×	r—		<u> </u>						l	
	s	<u> </u>	5	5	Э	2	5	5	D										
Flags	٩.	2	5	5	D	5	D	5	5										-
Fla	>	0	0	0	0	0	•	0	0					-					
	ζ	0	0	0	. 0	0	•	0	0										×
	AC (D	Э	>	2	Þ	>	2	5						<u> </u>	ļ,			
	Uperation	reg8 bit NO.CL = 0 : Z←1 reg8 bit NO.CL = 1 : Z←0	(mem)8 bit NO.CL = 0 : Z←1 (mem)8 bit NO.CL = 1 : Z←0	reg16 bit NO.CL = 0 : Z←1 reg16 bit NO.CL = 1 : Z←0	(mem16) bit NO.CL = 0 : Z←1 (mem16) bit NO.CL = 1 : Z←0	reg8 bit NO.imm3 = 0 : Z←1 reg8 bit NO.imm3 = 1 : Z←0	(mem8) bit NO.imm3 = 0 : Z←1 (mem8) bit NO.imm3 = 1 : Z←0	reg16 bit NO.imm4 = 0 : Z←1 reg16 bit NO.imm4 = 1 : Z←0	(mem16) bit NO.imm4 = 0 : Z←1 (mem16) bit NO.imm4 = 1 : Z←0	reg8 bit NO.CL←reg8 bit NO.CL	(mem8) bit NO.CL←(mem8) bit NO.CL	reg16 bit NO.CL←reg16 bit NO.CL	(mem16) bit NO.CL←(mem16) bit NO.CL	reg8 bit NO.imm3←reg8 bit NO.imm3	(mem8) bit NO.imm3←(mem8) bit NO.imm3	reg16 bit NO.imm4←reg16 bit NO.imm4	(mem16) bit NO.imm4←(mem16) bit NO.imm4	1st byte = 0FH	cv← <u>cv</u>
	bytes	3	3 to 5	3	3 to 5	4	4 to 6	4	4 to 6	m	3 to 5	e	3 to 5	4	4 to 6	4	4 to 6	*	-
n Code	76543210	1 1 0 0 0 reg	mod 0 0 0 mem	11000 reg	mod 0 0 0 mem	11000 reg	mod 0 0 0 mem	1 1 0 0 0 reg	mod 0 0 0 mem	11000 reg	mod 0 0 0 mem	11000 reg	mod 0 0 0 mem	11000 reg	mod 0 0 0 mem	11000 reg	mod 0 0 0 mem	3rd byte •	
Operation Code	76543210	0001000	0000	0001	0001	1000	1000	1001	1000	0110	0110	0111	0111	1110	1110	1111	1111	2nd byte *	11110101
	Operand(s)	reg8, CL	mem8, CL	reg16, CL	mem16, CL	reg8, imm3	mem8, imm3	reg16, imm4	mem16, imm4	reg8, CL	mem8, CL	reg16, CL	mem16, CL	reg8, imm3	mem8, imm3	reg16, imm4	mem16, imm4		c۲
duo	Logi Mnemonic						ieni no	iseluqi	nem jiš	3			MOT-						NOTI

NEC

6427525 0063222 931 🗰

Flags	-																		0			
	AC CY																					
	Operation	reg8 bit NO.CL←0	(mem8) bit NO.CL←0	reg16 bit NO.CL←0	(mem16) bit NO.CL←0	reg8 bit NO.imm3←0	(mem8) bit NO.imm3←0	reg16 bi NO.imm4←0	(mem16) bit NO.imm4←0	reg8 bit NO.CL←1	(mem8) bit NO.CL←1	reg16 bit NO.CL←1	(mem16) bit NO.CL←1	reg8 bit NO.imm3←1	(memB) bit NO.imm3←1	reg16 bi NO.imm4←1	(mem16) bit NO.imm4←1	1st byte = 0FH	CY←0	DIR←0	CY←1	DIR←0
	Bytes		3 to 5	e	3 to 5	4	4 to 6	4	4 to 6	e	3 to 5	3	3 to 5	4	4 to 6	4	4 to 6	*	-	٦	1	-
n Code	76543210	11000 reg	mod 0 0 mem	11000 reg	mod 0 0 0 mem	11000 reg	mod 0 0 mem	11000 reg	mod 0 0 mem	11000 reg	mod 0 0 0 mem	1 1 0 0 0 reg	mod 0 0 0 mem	11000 reg	mod 0 0 0 mem	11000 reg	mod 0 0 0 mem	3rd byte *				
Operation Code	76543210	00010010	0010	0011	0011	1010	1010	101	1011	0100	0100	0101	0101	1100	1100	1101	1101	2nd byte *	1111000	1111100	1111001	1111101
	Operand(s)	reg8, CL	mem8, CL	reg16, CL	mem16, CL	reg8, imm3	mem8, imm3	reg16, imm4	mem16, imm4	reg8, CL	mem8, CL	reg16, CL	mem16, CL	reg8, imm3	mem8, imm3	reg16, imm4	mem16, imm4		C۷ .	DIR	ζ	aid
di	Mnemonic					Sr Sr	ctioi	การเ	ni no	itelu	qine	m ti		SETI		·	· · · · · ·			CLR1		SET1

6427525 0063223 878 🖿

	N		×		×		×		x	×	×	
	S		>		<u>></u>		<u>></u>		>	n n n n	7 7 7 7 7 7 7 7	
Flags	Р		>		>		>		2	<u> </u>	2	
"	AC CY V		>		2		>		>	<u> </u>	>	
	<u>ठ</u>	:	2		>		>		2	2	2	
	¥_		<u> </u>		<u> </u>		>		>	>	2	
	Operation	 © CL← 0 © [When regB bit No.CL=0] if CL<7, re-executed from CL← CL+1, © 	if CL=7 Z← 1 [When reg8 bit No.CL=1] Z← 0	 Ø CL← 0 Ø [When mem8 bit No.CL=0] if CL<7, re-executed from CL← CL+1, Ø 	if CL=7 2←1 [When mem8 bit No.CL=1] 2←0	 0 CL← 0 0 When real5 bit No.CL=01 if CL<15. re-executed from CL← CL+1. Ø 	if CL=15 Z←1 U U U U U U U U U U U	0 CL← 0 © [When mem]§ bit No.CL=0] if CL<15. re-exercised from Cl← Cl+1 ©	if CL=15 Z← 1 [When mem16 bit No.CL=1] Z← 0	Removes block queued at head of queue and stores its segment address in P2.	Removes queue block indicated by P2.	Queues block indicated by P2 at end of queue.
ć	bytes		m	L L	3 10 5		m		c 01 £	4	4	4
n Code	76543210	00111100		00111100		00111101		10111101		01110000	01110001	01110010
Operation Code	76543210	00001111	11000 reg	00001111	mod 0 0 0 mem	00001111	11000 reg	00001111	mod 0 0 0 mem	00001111	00001111	00001111
(a)passar	operanois	- 0 ·	rego				regio		01 (12)	imm 16	imm 16	imm16
More			1	<u></u>						аноит•	00UT•	atin•
nction UC	nsol 10		·			ักธฑ มาระก				-uqinar enoitout		

This instruction is newly added to the V25 or V35.

Remarks P2: Parameter table (in register file)

6427525 0063224 704 🖿

	И	×	×	×	×	×	×	×	x	×	×	×	×
	s	×	×	×	×	×	×	×	×	×	×	×	×
Flags	٩	×	×	×	×	×	×	×	×	×	×	×	×
	>	×	×	D	n	2	D	×	×	n	2	n	D
	2	×	×	×	×	×	×	×	×	×	×	×	. ×
	AC	D	n	D D	5	5	2	. ⊃	<u></u>	2	>	<u> </u>	2
	Operation	CY←reg MSB, reg←reg × 2 If reg MSB ≠ CY, V←1 If reg MSB = CY, V←0	CY←(mem) MSB, (mem)←(mem) × 2 If (mem) MSB ≠ CY, V←1 If (mem) MSB = CY, V←0	temp←CL, while temp ≠ 0, the following operations are repeated: CY←reg MSB, reg←reg × 2 temp←temp − 1	temp←CL, while temp ≠ 0, the following operations are repeated: CY←(mem) MSB, (mem)←(mem) × 2 temp←temp − 1	temp←imm8, while temp ≠ 0, the following operations are repeated: CY←reg MSB, reg←reg × 2 temp←temp − 1	temp←imm8, while temp ≠ 0, the following operations are repeated: CY←(mem) MSB, (mem)←(mem) × 2 temp←temp - 1	CV←reg LSB, reg←reg + 2 If reg MSB ≠ bit after reg MSB: V←1 If reg MSB = bit after reg MSB: V←0	CY←(mem) LSB, (mem)←(mem) + 2 If (mem) MSB ≠ bit after (mem) MSB: V←1 If (mem) MSB = bit after (mem) MSB: V←0	temp←CL, while temp ≠ 0, the following operations are repeated: CY←reg LSB, reg←reg + 2 temp←temp - 1	temp←CL, while temp ≠ 0, the following operations are repeated: CY←(mem) LSB, (mem)←(mem) + 2 temp←temp - 1	tempt–imm8, while temp ≠ 0, the following operations are repeated: CY←reg LSB, reg←reg + 2 temp←temp – 1	temp←imm8, while temp ≠ 0, the following operations are repeated: CY←(mem) LSB, (mem)←(mem) + 2 temp←temp - 1
	bytes	2	2 to 4	2	2 to 4	m	3 to 5	7	2 to 4	7	2 to 4	e	3 to 5
n Code	76543210	11100 reg	mod1 0 0 mem	11100 reg	mod1 0 0 mem	11100 reg	mod1 0 0 mem	11101 reg	mod1 0 1 mem	11101 reg	mod1 0 1 mem	11101 reg	mod1 0 1 mem
Operation Code	76543210	1101000	1 1 0 1 0 0 0 W	110101W	110101W	1100000	100000	10100W	1101000	W1001011	1101001W	1 1 0 0 0 0 W	110000W
	Operand(s)	reg, 1	mem, 1	reg, CL	mem, CL	reg, imm8	mem, imm8	reg, 1	mem, 1	reg, CL	mem, CL	reg, imm8	mem, imm8
unto notion			·		л Л Л		enoitous	teni tiid2			SHR	.	

📟 6427525 OO63225 640 📟

	Ν	×	×	×	×	×	×						
	S	×	×	×	×	×	×						
Flags	ď	×	×	×	×	×	· ×						
Ξ	۷	0	0	n	n	Э	5	×	×	D	5	n	n
	ζ	×	×	×	×	×	×	×	×	×	×	×	×
	AC	Þ	Э	n	2	D	⊃						
		CY←reg LSB, reg←reg + 2, V←0 MSB of operand is unchanged.	CY←(mem) LSB, (mem)←(mem) + 2, V←0 MSB of operand is unchanged.	temp←CL, while temp ≠ 0, the following operations are repeated: CY←reg LSB, reg←reg + 2 temp←temp - 1, MSB of operand is unchanged.	temp←CL, while temp ≠ 0, the following operations are repeated: CY←(mem) LSB, (mem)←(mem) + 2 temp←temp - 1, MSB of operand is unchanged.	termp←imm8, while termp ≠ 0, the following operations are repeated: CY←reg LSB, reg←reg + 2 temp←termp - 1, MSB of operand is unchanged.	temp←imm8, while temp ≠ 0, the following operations are repeated: CY←(mem) LSB, (mem)←(mem) + 2 temp←temp – 1, MSB of operand is unchanged.	CY←reg MSB, reg←reg × 2 + CY reg MSB ≠ CY: V←1 reg MSB = CY: V←0	CY←(mem) MSB, (mem)←(mem) × 2 + CY (mem) MSB ≠ CY: V←1 (mem) MSB = CY: V←0	temp←CL, while temp ≠ 0, the following instructions are repeated: CY←reg MSB, reg←reg × 2 + CY temp←temp – 1	temp←CL, while temp ≠ 0, the following instructions are repeated: CY←(mem) MSB, (mem)←(mem) × 2 + CY temp←temp - 1	temp←imm8, while temp ≠ 0, the following instructions are repeated: CY←reg MSB, reg←reg × 2 + CY temp←temp – 1	tempt-imm8, while temp \neq 0, the following instructions are repeated: CY \leftarrow (mem) MSB, (mem) \leftarrow (mem) × 2 + CY tempt-temp - 1
Butae	calfa	2	2 to 4	2	2 to 4	e	3 to 5	2	2 to 4	2	2 to 4	3	3 to 5
ר Code	76543210	11111 reg	mod1 1 mem	1111 reg	mod1 1 1 mem	1111 reg	mod1 1 mem	11000 reg	mod0 0 0 mem	11000 reg	mod0 0 0 mem	11000 reg	mod0 0 0 mem
Operation Code	76543210	110100W	1101000W	1101001W	W1001011	110000W	110000W	1101000W	1101000W	1101001W	110101W	110000W	110000W
Onersodiel		reg, 1	mem, 1	reg, CL	mem, CL	reg, imm8	mem, imm8	reg, 1	mem, 1	reg, CL	mem, CL	reg, imm8	mem, imm8
uction oup	_ 6۲	· · · · · · · · ·	<i>-</i>	noitou	nteni thir N HR A A	IS				truction	eni etsto	 ਮ	

6427525 0063226 587 🖿

μ**PD7043**3

Powered by ICminer.com Electronic-Library Service CopyRight 2003

	2										
3	S										
Flags	⊿ >	. ×	×	<u> </u>				×			
	- ک	· ^		 ×	 ×		ر ×		×	< ×	 ×
	AC 0								~		
	Operation	CY←reg LSB, reg←reg + 2 reg MSB←CY reg MSB ≄ bit after reg MSB: V←1 reg MSB = bit after reg MSB: V←0	CY←(mem) LSB, (mem)←(mem) + 2 (mem) MSB←CY (mem) MSB ≠ bit after (mem) MSB: V←1 (mem) MSB = bit after (mem) MSB: V←0	temp←CL, while temp ≠ 0, the following operations are repeated: CY←reg LSB, reg←reg + 2 reg MSB←CY temp←temp – 1	temp←CL, while temp ≠ 0, the following operations are repeated: CY←(mem) LSB, (mem)←(mem) + 2 (mem) MSB←CY temp←temp - 1	temp←imm8, while temp ≠ 0, the following operations are repeated: CY←reg LSB, reg←reg + 2 reg MSB←CY temp←temp - 1	temp←imm8, while temp ≠ 0, the following operations are repeated: CY←(mem) LSB, (mem)←(mem) + 2 (mem) MSB←CY temp←temp - 1	tmpcy←CY, CY←reg MSB reg←reg × 2 + tmpcy reg MSB ≠ CY: V←1 reg MSB = CY: V←0	tmpcy←CY, CY←(mem) MSB (mem)←(mem) × 2 + tmpcy (mem) MSB ≠ CY: V←1 (mem) MSB = CY: V←0	temp←CL, while temp ≠ 0, the following operations are repeated: tmpcy←CY, CY←reg MSB reg←reg × 2 + tmpcy temp←temp − 1	temp \leftarrow CL, while temp \neq 0, the following operations are repeated: tmpcy \leftarrow CY, CY \leftarrow (mem) MSB (mem) \leftarrow (mem) × 2 + tmpcy temp \leftarrow temp – 1
	bytes	5	2 to 4	3	2 to 4	e	3 to 5	2	2 to 4	2	2 to 4
n Code	76543210	11001 reg	mod 0 0 1 mem	11001 reg	mod 0 0 1 mem	11001 reg	mod 0 0 1 mem	11010 reg	mod 0 1 0 mem	11010 reg	mod 0 1 0 mem
Operation Code	76543210	1 1 0 1 0 0 W	1 1 0 1 0 0 0 W	1 1 0 1 0 0 1 W	1 1 0 1 0 0 1 W	1 1 0 0 0 0 W	110000W	110100W	1 1 0 1 0 0 W	110101W	1101001W
	Operand(s)	reg, 1	mem, 1	reg, CL	mem, CL	reg, imm8	тет, ітт8	reg, 1	mem, 1	reg, CL	mem, CL
dn	Mnemonic				С. Ж	enoi):	ourteni etet	о <u>н</u>	1	ROLC	

	N								
	S								
Flags	٥.								
Ē	>	С	Э	×	×	n	D	<u> </u>	C
	CΥ	×	×	×	×	×	×	×	×
	AC								
	Operation	temp←imm8, while temp ≠ 0, the following instructions are repeated: tmpcy←CY, CY←reg MSB reg←reg × 2 + tmpcy temp←temp − 1	temp←imm8, while temp ≠ 0, the following instructions are repeated: tmpcy←CY, CY←(mem) MSB {mem}←(mem) × 2 + tmpcy temp⊄-temp – 1	tmpcy←CY, CY←reg LSB reg←reg + 2 reg MSB←tmpcy reg MSB ≠ bit after reg MSB: V←1 reg MSB = bit after reg MSB: V←0	tmpcy←CY, CY←(mem) LSB (mem)←(mem) + 2 (mem) MSB←tmpcy (mem) MSB ≠ bit after (mem) MSB: V←1 (mem) MSB = bit after (mem) MSB: V←0	temp←CL, while temp ≠ 0, the following operations are repeated: tmpcy←CY, CY←reg LSB reg←reg + 2 reg MSB←tmpcy temp←temp - 1	temp←CL, while temp ≠ 0, the following operations are repeated: tmpcy←CY, CY←(mem) LSB (mem)←(mem) + 2 (mem) MSB←tmpcy temp←temp - 1	temp←imm8, while temp ≠ 0, the following operations are repeated: tmpcy←CY, CY←reg LSB reg←reg + 2 reg MSB←tmpcy temp←temp ~ 1	temp←imm8, while temp ≠ 0, the following operations are repeated: tmpcy←CY, CY←(mem) LSB (mem)←(mem) + 2 (mom) MSB←tmpcy temp←temp - 1
	Bytes	3	3 to 5	3	2 to 4	7	2 to 4	ю	3 to 5
n Code	7.6543210	11010 reg	mod 0 1 0 mem	11011 reg	mod 0 1 1 mem	11011 reg	mod 0 1 1 mem	11011 reg	mod 0 1 1 mem
Operation Code	76543210	110000W	110000W	110100W	1101000W	1101001W	110101W	110000W	1 1 0 0 0 0 W
	Operand(s)	reg, imm8	mem, imm8	reg, 1	тет, 1	reg, CL	mem, CL	reg, imm8	mem, imm8
dna	Mnemonic								
noito	untenl				ructions	teni ətstoA			

🖿 6427525 0063228 35T 🖿

noira qu			Operation Code	n Code				Flags	
unten Gro	Mnemonic	Uperand(s)	76543210	76543210	bytes	Operation	AC CY V	٩	s z
		near-proc	11101000		e	(SP - 1, SP - 2)←PC, SP←SP - 2 PC←PC + disp			
		regptr16	1111111	11010 reg	2	(SP - 1, SP - 2)←PC, SP←regptr16 SP←SP - 2			
SL	CALL	memptr16	1111111	mod 0 1 0 mem	2 to 4	(SP – 1, SP – 2)←PC, SP←SP – 2 PC←(memptr16)			
Intruction		far-proc	10011010		ß	(SP - 1, SP - 2)←PS, (SP - 3, SP - 4)←PC SP←SP - 4 PC←seg, PC←offset			
ni lontrol ir		memptr32	111111	mod 0 1 1 mem	2 to 4	(SP - 1, SP - 2)←PS, (SP - 3, SP - 4)←PC SP←SP - 4 PC←(memptr32 + 2), PC←(memptr32)			
enituo			1100011		-	PC←(SP + 1, SP) SP←SP + 2	 		
Subro		pop-value	1 1 0 0 0 1 0		e	PC←(SP + 1, SP) SP←SP + 2, SP←SP + pop-value			
	RET		1 1 0 0 1 0 1 1		1	PC←(SP + 1, SP) PS←(SP + 3, SP + 2) SP←SP + 4			
	•	pop-value	1 1 0 0 1 0 1 0		3	PC←(SP + 1, SP) PS←(SP + 3, SP + 2) SP←SP + 4, SP←SP + pop-value			
		mem16	1111111	mod 1 1 0 mem	2 to 4	(SP - 1, SP - 2)←(mem16) SP←SP - 2			
uc		reg16	01010 reg		-	(SP – 1, SP – 2)←reg16 SP←SP – 2			
oitounta	•	sreg	0 0 0 sreg 1 1 0		1	(SP – 1, SP – 2)←sreg SP←SP – 2			
ni noit	PUSH	PSW	10011100		l	(SP - 1, SP - 2)←PSW SP←SP - 2			
einq	L <u></u>	æ	01100000		-	Push registers on the stack			
inem X	L	imm8	01101010		2	(SP - 1, SP - 2)←imm8 sign extension SP←SP - 2			
Stac	L	imm16	01101000		e	(SP - 1, SP - 2)←imm16 SP←SP - 2			
		DS2*	00001111	00111110	2	(SP - 1, SP - 2)←DS2 SP←SP - 2			
F •	his instructi	This instruction is newly added to the	led to the V25 or V35.						

■ 6427525 0063229 296 **■**

noita			Operation Code	n Code				–	Flags		
untenl Dið	Mnemonic	Operand(s)	76543210	76543210	Bytes	Operation	ACC	<u>د <</u>	٩	s	N
	PUSH	DS3/VPC+	00001111	00110110	5	(SP - 1, SP - 2)←DS3/VPC SP←SP - 2				1	1
su		тет16	10001111	mod 0 0 0 mem	2 to 4	SP←SP + 2 (mem16)←(SP - 1, SP - 2)		<u> </u>			T
ruction		reg16	01011 reg		1	SP←SP + 2 reg16←(SP – 1, SP – 2)					<u> </u>
teni no		sreg	0 0 0 sreg 1 1 1	-	1	SP←SP + 2 sreg16←(SP - 1, SP - 2) sreg: SS, DS0, DS1					
iteluqi	POP	PSW	10011101		-	SP←SP + 2 PSW←(SP - 1, SP - 2)	æ	عد حد	٣	œ	æ
nem		œ	01100001		-	Pop registers from the stack		-		1	1
Stack		DS2*	00001111	00111111	2	SP←SP + 2 DS2←(SP - 1, SP - 2)					<u> </u>
		DS3/VPC+	00001111	00110111	2	SP←SP + 2 DS3/VPC←(SP - 1, SP - 2)				<u> </u>	<u> </u>
	PREPARE	imm16, imm8	11001000		4	Prepare New Stack Frame		<u> </u>			
	DISPOSE		11001001		+	Dispose of Stack Frame					
		near-labet	1110101		e	PC←PC + disp					
uoi		short-label	11101011		2	PC←PC + ext-disp8					
touri		regptr16	1111111	11100 reg	2	PC←regptr16					
ຣບເ ເ	BR	memptr16	1111111	mod 1 0 0 mem	2 to 4	PC←(memptr16)					
Branch		far-label	11101010		ß	PS ← seg PC ← offset					
		memptr32	1111111	mod 1 0 1 mem	2 to 4	PS←(memptr32 + 2) PC←(memptr32)					
•	This instruct	This instruction is newly added to the	led to the V25 or V35.		- 						

🖿 6427525 0063230 TO& 📰

Mathematication Deparinded J 6 6 4 3 2 10 J 6 5 4 3 2 10 J 6 5 4 3 2 10 J 6 5 4 3 2 10 J 6 6 4 7 4 J 6 7 4 J 6 7 4 J 7 8 1 W short-label 0 11 10 0 0 1 1 10 0 0 2 1 1 4 - 0 P C - PC + extrding 8 1 1 1 0 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 1 0 1 1 1 0 1 1 1 1 0					Operation Code	n Code					Flags	
BVshort-label0<11<0002it V-1BVVshort-label0<010		monic	Uperand(s)		54321	654321	bytes	Uperatio		AC CY	>	
BWVshort-label0001 10010 2 $11V = 0$ BCshort-label00110 2 $11V = 0$ 10010 2 $11V = 0$ BVCshort-label01010 2 $11C = 1$ 2 $11C = 1$ BVLshort-label01010 2 $11C = 1$ 2 $11C = 1$ BVLshort-label01010 2 $11C = 1$ 2 $11C = 1$ BVLshort-label0110 2 $11C = 1$ 2 $11C = 1$ BVLshort-label 1 1001 2 $11C = 1$ BVshort-label 1 1001 2 $11C = 1$ BVshort-label 1 1001 2 $11C = 2$ BVshort-label 1 1001 2 $11C = 2$ BVshort-label 1 1100 2 $11C = 2$ BVshort-label 1 1001 2 $11C = 2$ BVshort-label 1 1000 2 $11C = 2$ BVshort-label 1 1000 2 $11C = 2$	BV		short-label	0	1000		5		-PC + ext-disp8			
BC BC BUCshort-label001002If CV = 1BUC BUC BUC BUC BUC BUC BUC BUCshort-label001012If CV = 0BUC B	BNV		short-label		0 0		2		-PC + ext-disp8			
BNC short-label 00111 2011 2 if CY=0 BE short-label 0100 2 if CY=0 BNE short-label 0101 2 if CY=0 BNE short-label 0110 2 if CY=0 BNE short-label 0110 2 if CY<2=0	BC BL		short-label		0 1		2	CY = 1	-PC + ext-disp8			
BE BDX BDX BDX 	BNC BNL		short-label		0		3	CY = 0	-PC + ext-disp8			
BNE short-label D 1 0 1 Z it Z = 0 BNH short-label D 1 1 0 Z if C V Z = 1 BH short-label D 1 1 0 Z if C V Z = 0 BH short-label D 1 1 0 Z if C V Z = 0 BH short-label D 1 0 0 0 Z if S = 0 BP short-label D 1 0 0 1 Z if S = 0 BP short-label D 1 0 1 1 Z if S = 0 BP short-label D 1 0 1 1 Z if S = 0 BP short-label D 1 0 1 1 Z if S = 0 BP short-label D 1 1 0 0 Z if S = 0 BE short-label T 1 1 1 0 Z if S = 0 BE short-label T 1 1 1 0 Z if S = 0 BE short-label T 1 1 1 0 Z if S = 0 BE short-label T 1 1 0 Z if S = 0 BE short-label T 1 1 0 </td <td>BE BZ</td> <td></td> <td>short-label</td> <td></td> <td>1 0</td> <td></td> <td>2</td> <td>Z = 1</td> <td>-PC + ext-disp8</td> <td></td> <td></td> <td></td>	BE BZ		short-label		1 0		2	Z = 1	-PC + ext-disp8			
BNH short-label 0 1 1 1 2 if CY v Z = 0 BH short-label 0 1 1 1 2 if CY v Z = 0 BN short-label 1 0 0 0 2 if S = 1 BN short-label 1 0 0 0 2 if S = 0 BP short-label 1 0 0 1 2 if S = 0 BP short-label 1 0 1 0 2 if S = 0 BP short-label 1 1 0 0 2 if S = 0 BPO short-label 1 1 1 0 2 if S = 0 BIL short-label 1 1 1 0 2 if S = 0 BIL short-label 1 1 1 0 2 if S = V = 0 BIL short-label 1 1 1 0 2 if S = V = 0 BIL short-label 1 1 1 0 2 if S = V = 0 BIL short-label 1 1 1 0 2 if S = V = 0 DBNZNE short-label 1 1 1 0 2 if S = V = 0 DBNZNE short-label	BNE BNZ		short-label		1 0		7	Z == 0	-PC + ext-disp8			
BH short-label 0111 2 if CV v Z = 0 BN short-label 1000 2 if S = 1 BP short-label 1001 2 if S = 1 BP short-label 1011 2 if S = 0 BP short-label 1011 2 if S = 0 BPO short-label 11010 2 if S = 0 BC short-label 11100 2 if S = V = 1 BC short-label 1110 2 if S = V = 0 BC short-label 1110 2 if S = V = 1 BC short-label 1110 2 if S = V = 1 BC short-label 1110 2 if S = V = 0 BC short-label 11100 2 if S = V = 0 BC short-label 11100 2 if S = V = 0 BC short-label 11100 2 if S = V = 0 BRZNE short-label 111000 2	BNH		short-label		-		2	CY v Z = 1	-PC + ext-disp8			
BN short-label 1000 2 if S=1 BP short-label 1001 2 if S=0 BP short-label 1001 2 if S=0 BPO short-label 1011 2 if S=0 BPO short-label 1011 2 if S=0 BPO short-label 1110 2 if S=V=1 BC short-label 1110 2 if S=V=0 BL short-label 1110 2 if S=V=0 BGT short-label 1110 2 if S=V=0 BGT short-label 1 1100 2 if S=V) DBNZE short-label 1 2 if S=V) 2 DBNZE short-label 1 0 2	L		short-label		-		2		-PC + ext-disp8			
BP short-label 1001 2 if $F=0$ BPC short-label 1010 2 if $F=0$ BPO short-label 1011 2 if $F=0$ BPO short-label 1100 2 if $F=0$ BPO short-label 1100 2 if $F=0$ BC short-label 1100 2 if $S=V=0$ BC short-label 1110 2 if $S=V=0$ BG short-label 11100000 2 if $S=V=0$ DBNZE short-label 111000000 2 if $S=V=0$ DBNZ short-label 10001 2 if $S=V=0$ DBNZ short-label 0001 2 if $S=V=0$ DBNZ short-label 0001 2 <	I		short-label		0		2	S = 1	-PC + ext-disp8			
BPE short-label 1010 2 if P=1 BPO short-label 1011 2 if P=0 BLT short-label 1100 2 if SvV=1 BLT short-label 11101 2 if SvV=1 BL short-label 11101 2 if SvV=1 BL short-label 11101 2 if SvV=0 BGE short-label 11101 2 if SvV=0 BGT short-label 1110000 2 if SvV=0 BGT short-label 11100000 2 if (SvV)vZ=0 DBNZK short-label 1100000 2 if (SvV)vZ=0 DBNZ short-label 100010 2 if (SvV)vZ=0 DBNZ short-label 0	<u> </u>		short-label		0		2	S = 0	-PC + ext-disp8			
BPO short-label 1011 2 if $P = 0$ BLT short-label 1100 2 if $S = V = 1$ BGE short-label 1100 2 if $S = V = 0$ BGE short-label 1110 2 if $S = V = 0$ BGE short-label 1110 2 if $S = V = 0$ BGT short-label 1110 2 if $S = V = 0$ BGT short-label 1110 2 if $(S = V) = 0$ BGT short-label 1110 2 if $(S = V) = 0$ DBNZNE short-label 11100000 2 if $(S = V) = 0$ DBNZ short-label 11100000 2 if $(S = V) = 0$ DBNZ short-label 10001 2 if $(S = V) = 0$ BNZ short-label 10001 2 if $(S = V) = 0$ BNZ short-label 10001 2 if $(S = V) = 0$ BCWZ short-label 10001 2 if $(S = V) = 0$ BCWZ	L		short-label		0 1		2	P = 1	-PC + ext-disp8			
BLT short-label 1100 2 If S $\vee V = 1$ BGE short-label 1101 2 If S $\vee V = 0$ BLE short-label 1110 2 If S $\vee V = 0$ BLE short-label 1110 2 If S $\vee V = 0$ BLE short-label 1110 2 If S $\vee V = 0$ BLE short-label 1110 2 If S $\vee V = 0$ BGT short-label 1110 2 If (S $\vee V) \vee Z = 0$ DBNZNE short-label 1110000 2 If (S $\vee V) \vee Z = 0$ DBNZNE short-label 1<100000 2 If (S $\vee V) \vee Z = 0$ DBNZNE short-label 0<001 2 If (S $\vee V) \vee Z = 0$ DBNZ short-label 0<001 2 If (S $\vee V) \vee Z = 0$ DBNZ short-label 0<001 2 If (S $\vee V) \vee Z = 0$ DBNZ short-label 0<001 2 If (S $\vee V) \vee Z = 0$ DBNZ short-label 0<010 2 If (S $\vee V) \vee Z = 0$ DBNZ short-label 0<011 2 If (S \vee	1		short-label		0		2	P = 0	-PC + ext-disp8	·		
BGE short-label 1101 2 If S \vee 10 BLE short-label 1110 2 if (S \vee 2) \vee 2=1 BLE short-label 1110 2 if (S \vee 2) \vee 2=0 BGT short-label 11100000 2 if (S \vee 1) \vee 2=0 DBNZNE short-label 11100000 2 if (S \vee 0) \vee 2=0 DBNZNE short-label 11100000 2 if (S \vee 0 DBNZ short-label 0001 2 if (S \vee 0 1 DBNZ short-label 0001 2 if (S \vee 0 1 DBNZ short-label 00010 2 if (S \vee 0 1 BCVZ short-label 00010 2 if (S \wedge 0 1 BCVZ short-label 00011 10011100 5 if (S \wedge 0 BCLR+2 Sfr/imm3, 00001111 10011101 5 if (S \wedge 0 BTCLR+2 Short-label 0001111 10011101 5 if (S \wedge 0	L		short-label		0		2	L	-PC + ext-disp8			
BLEshort-label11102if (S × Z) × Z = 1BGTshort-label \intercal 11112if (S × V) × Z = 0BGTshort-label \intercal 11112if (S × V) × Z = 0DBNZKshort-label \intercal 111000002if (S × V) × Z = 0DBNZEshort-label \intercal 00012if (S × V) × Z = 0DBNZEshort-label \intercal 00012if (S × V) × Z = 0DBNZshort-label \intercal 00012if (S × V) × Z = 0BNZshort-label \intercal 00102if (S × V) × Z = 0BNZshort-label \intercal 00102if (S × V) × Z = 0BTCLR1short-label \intercal 0001111100111005if (S × V) × Z = 0BTCLR12short-label \intercal 0001111100111005if (S × V) × Z = 0BTCLR12short-label 0 0001111100111015if (S × V) × Z = 0BTCLR12short-label 0 0001111100111015if (S × V) × Z = 0	1	 -	short-label		10		2	S + V = 0	-PC + ext-disp8			
short-label 1 1 1 1 2 if (S \vee V) \vee Z = 0 E short-label 1 1 1 0 0 0 0 0 2 if Z = 0 and CW \neq 0 short-label 1 1 1 0 0 0 0 0 2 if Z = 1 and CW \neq 0 short-label 0 0 0 1 2 if Z = 1 and CW \neq 0 short-label 0 0 0 1 2 if Z = 1 and CW \neq 0 short-label 0 0 1 0 2 if Z = 0 and CW \neq 0 short-label 0 0 1 0 2 if Z = 0 and CW \neq 0 short-label 0 0 1 1 1 0 0 1 1 1 0 2 if C = 0 short-label 0 0 0 1 1 1 1 1 0 0 1 1 1 0 5 if (sf1) bit No. imm3 = 1: short-label short-label 0 0 0 0 1 1 1 1 1 0 0 1 1 1 0 1 5 if (sf1) bit No. imm3 = 1: short-label 1 0 0 1 1 1 0 1 5 if (sf1) bit No. imm3 = 1: short-label			short-label		1		7	-	-PC + ext-disp8			
E short-label 1 1 0 0 0 0 0 2 $CW = CW - 1$ short-label 1 1 0 0 0 0 1 2 $CW = CW - 1$ short-label 0 0 0 1 2 $CW = CW - 1$ short-label 0 0 0 1 2 $CW = CW - 1$ short-label 0 0 1 0 2 $CW = CW - 1$ short-label 0 0 1 0 2 $CW = CW - 1$ short-label 0 0 1 1 2 $CW = CW - 1$ short-label 1 0 0 1 1 2 $If CW \neq 0$ short-label 0 0 0 0 1 1 1 1 1 0 0 1 1 1 0 0 5 $If (sfr) bit No. imm3 = 1$: short-label 0 0 0 0 1 1 1 1 1 0 0 1 1 1 0 1 5 $PC-PC + ext-disp8, (sfr) bit No. imm3 = 1$:	BGT		short-label		11		2	0 -	-PC + ext-disp8			
short-label 0 0 0 1 2 CW = CW - 1 short-label 0 0 1 0 2 if Z = 1 and CW ≠ 0 short-label 0 0 1 0 2 if CW ≠ 0 short-label 0 0 1 1 1 2 if CW ≠ 0 short-label 0 0 1 1 1 2 if CW ≠ 0 short-label 0 0 0 1 1 1 1 0 0 1 1 1 0 5 if CW ≠ 0 short-label 0 0 0 0 1 1 1 1 1 0 0 1 1 1 0 0 5 if Str1 bit No. imm3 = 1: *2 short-label 0 0 0 0 1 1 1 1 1 0 0 1 1 1 0 1 5 if Str1 bit No. imm3 = 1:	DBNZ	ZNE	short-label		0000		2		-PC + ext-disp8			
short-label 0 0 1 0 2 $CW = CW - 1$ short-label 0 0 1 1 2 if $CW \neq 0$ short-label 0 0 0 1 1 1 1 0 0 1 1 1 0 2 if $CW = 0$ short-label 0 0 0 0 1 1 1 1 1 0 0 1 1 1 0 0 5 if (sfr) bit No. imm3 = 1: t sfr, imm3, 0 0 0 0 1 1 1 1 1 0 0 1 1 1 0 0 5 if (sfr) bit No. imm3 = 1: t str, imm3, 0 0 0 0 1 1 1 1 1 0 0 1 1 1 0 1 5 if (sfr) bit No. imm3 = 1: t str, imm3, 0 0 0 0 1 1 1 1 1 0 0 1 1 1 0 1 5 if (sfr) bit No. imm3 = 1:	DBNZ	ZE	short-label		0 0		2		-PC + ext-disp8			
short-label 0 0 1 1 1 0 0 1 1 1 2 if CW = 0 •1 sfr, imm3, short-label 0 0 0 0 1 1 1 1 1 0 0 1 1 1 0 0 5 If (sfr) bit No. imm3 = 1; PCC-PC + ext-disp8, (sfr) bit No. imm3 = 1; short-label •2 sfrl, imm3, short-label 0 0 0 0 1 1 1 1 1 0 0 1 1 1 0 1 5 If (sfrl) bit No. imm3 = 1; PCC-PC + ext-disp8, (sfrl) bit No. imm3 = 1; short-label	DBNZ	2	short-label		01		2		-PC + ext-disp8			
sfr. imm3, short-label 0 0 0 0 1 1 1 1 1 0 0 1 1 1 0 0 5 short-label 0 0 0 0 1 1 1 1 1 0 0 1 1 1 0 1 5 short-label 0 0 0 0 1 1 1 1 1 0 0 1 1 1 0 1 5	BCW	2	short-label		1		2		-PC + ext-disp8			
sfrl, imm3, 00001111 10011101 5 short-label	BTCL	.R*1	sfr, imm3, short-label	0	0111	01110	2	If (sfr) bit No. imm3 = 1: PC←PC + ext-disp8, (sfr) bit No.i	imm3←0			
	BTCL	.RL*2	sfrl, imm3, short-label	0	00111	001110	ß	lf (sfrl) bit No. imm3 = 1: PC←PC + ext-disp8, (sfrl) bit No.	.imm3←0			

📖 6427525 DD63231 944 🔜

125

2. This instruction is newly added to the V25 or V35.

Bit Operation Constitution 7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0 7 6 7 3 2 0 7 6 5 4 3 2 0 7 6 5 4 3 2 0 7 6 5 4 3 2 0 7 6 5 6 5 3 0 7 6 5 6 5 3 0 7 6 5 6 5 3 0 7 6 6 7 1 3 1 0 7 6 7 1 3 1 0 7 6 7 1 3 1 0 7 6 7 1 3 1 0 7 6 7 1 3 1 0 7 6 7 1 3 1 0 7 6 7 1 3 1 0 7 6 7 1 3 1 0 7 6 7 1 3 1 0 7 6 7 1 3 1 0 7 6 7 1 3 1 0 7 6 7 1 3 1 0 7 6 7 1 3 1 0 7 6 7 1 3 1 0 7 6 7 1 3 1 0 7 6 7 1 3 1 0 7 6 7 1 3 1 0 7 6 7 1 1 0 7 6 7 1 1 1 1 7 6 7 1 1 1 1 1 7 6 7 1 1 1 1 1 1 7 6 7 1 1 1 1 1 1 1 7 7 1 1 1 1 1 1 1 7 7 1 1 1 1 1 1 1 7 7 1 1 1 1 1 1 1 7 1 1 1 1 1 1 1 7 1 1 1 1 1 1 1 7 1 1 1 1 1 1 1 1 7 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1		<u> </u>		Operation Code	n Code					Flace	_	
BIX 3 111001100 1 159-1.57-21.57-21.57-41-55 1 BIX imm8 11001101 1 159-1.57-31.57-31.57 1 1 BIX imm8 110011101 2 159-1.57-31.74.7 1		Mnemonic	Operand(s)	654321	654321	Bytes	Operation	ACIC	Ľ			
Imm Intoning Intoning <thintoning< th=""> <thintoning< th=""> <thint< td=""><td>R</td><td>0 2 2</td><td>m</td><td>00110</td><td></td><td></td><td>- 4)(+PS</td><td>1</td><td></td><td></td><td></td><td></td></thint<></thintoning<></thintoning<>	R	0 2 2	m	00110			- 4)(+PS	1				
Mix International			imm8 (≭ 3)	0110		2	$(SP - 1, SP - 2) \leftarrow PSW$, $(SP - 3, SP - 4) \leftarrow PS$ $(SP - 5, SP - 6) \leftarrow PC$, $SP \leftarrow SP - 6$ $IE \leftarrow 0$, $BRK \leftarrow 0$ $PS \leftarrow (n \times 4 + 3, n \times 4 + 2)$, $PC \leftarrow (n \times 4 + 1, n \times 4)$ $n = imm8$				ļ	l
RETIN 11001111 1001001 2 PSW-LISPE, SP, SP-SP + 6 R	t instructions	L.,		00111		-	If V = 1: (SP = 1, SP = 2)←PSW (SP = 3, SP = 4)←PS (SP = 5, SP = 6)←PC, SP←SP = 6 IE←0, BRK←0 PS←(19, 18), PC←(17, 16)		<u> </u>	l		1
FILTERIst 0 0 0 0 11 1 1 1 0 0 1 0 0 0 1 2 PCC-Value of PSW save area in currently selected bank register R	terrup			100111		-	PC+-(SP + 1, SP), PS+-(SP + 3, SP + 2) PSW+-(SP + 5, SP + 4), SP←-SP + 6			<u> </u>		<u> </u>
FINT* 0 0 0 0 1 1 1 1 1 0 0 0 0 0 1 1 1 1 1 0 0 0 0 1 1 1 1 1 0 0 0 0 0 1 1 1 1 1 0 0 0 0 0 1 1 1 1 1 0 0 0 0 1 1 1 1 1 0 0 0 0 0 1 1 1 1 1 0 0 0 0 0 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	ul 			000111	0100	2	PC←Value of PC save area in currently selected bank register, PSW←Value of PSW save area in currently selected bank register		I		· · · ·	<u> </u>
CHKIND reg. mem32 01100010 mod reg mem 210 (SP - 1, SP - 2), FPC, SP - 2) SP - 1, FPS		FINT*		000111	001001	2	Indicates to interrupt controller incorporated in CPU that interrupt servicing has ended.			ļ		
BRKCs* reg16 0 0 0 0 1 1 1 1 0 0 1 0 1 1 0 1 3 tempt-PSW BRCs* reg16 1 1 0 0 0 reg PSW save area in newly selected register bankt-PC PSW save area in unewly selected register bankt-PC PSW save area in newly selected register bankt-PC PSW save area in rewly selected register bankt-PC PSW save area in rewly selected register bankt-PC PSW save area in rewly se		снкімо	reg, mem32	1100	mod reg mem	2 to 4					<u> </u>	ļ
TSKSW* reg16 0 0 0 0 1 1 1 1 1 0 0 1 0 1 0 0 PSW save area in currently selected register bankt-PSW ×	bank structions		reg16	0 0 0 1 1 1 1 0 0 0 reg	01010	m	temp←PSW RB3 to 0←reg16 low-order 4 bits, IE←0, BRK←0 PSW save area in newly selected register bank←temp PC save area in newly selected register bank←PC			ļ		ļ
ALBIT 0 0 0 0 1 1 1 1 1 0 0 1 1 0 1 1 1 0 0 1 1 0 1 2 If CH + CL > 16: BW, DW→DS1:IY output to transmit buffer If CH + CL > 16: CH, BW, DW→BW U X U U U U U U U U V U U V U U V U	rətzigəA ari gaidətiwa		reg16	000111 1111 reg	001010	m				+		
End COLTRP 0 0 0 0 1 1 1 1 1 0 0 1 1 0 1 1 2 Stores 1 line pixel data change point information in change point table (start white run length). 1 0 0 1 1 1 1 1 0 0 1 0 1 1 1 0 0 1 0 1 1 1 0 0 1 0 1 1 1 0 0 1 0 1 1 2 Stores 1 line pixel data change point information in change point table (start white run length). 1 0 0 1 1 1 1 0 0 1 0 1 1 2 Generates MH code from change point table. U V U				000111	001101	7						
² MHENC 0 0 0 0 1 1 1 1 1 0 0 1 0 1 1 2 Generates MH code from change point table. U V V U				000111	001101	2	Stores 1 line pixel data change point information in change point table (start white run length).					
0 0 0 0 1 1 1 1 1 1 0 0 1 0 1 1 1 2 Generates MR code from change point table. U U U		ł		000111	001001	2					Э	
		MRENC		000	001011	7						

This instruction is newly added to the V20 or V30.

126

🖬 6427525 0063232 880 🔳

Powered by ICminer.com Electronic-Library Service CopyRight 2003

Mnemonic SCHEOL 76543210 76543210 PMes ECU: detection in MH SCHEOL 000011111 01111001 2 Fetches pixel data start GETBIT 000011111 01111101 2 Generates change poin MHDEC 000011111 01111101 2 Generates change poin MHDE 00001111 01111101 2 Generates change poin MHDE 00001111 01111101 2 Generates change poin MHDE 00001111 01111101 2 Generates change poin CNVTRP 000011111 01111101 2 CPU Halt CNVTRP 000011111 10011110 2 CPU Halt CNVTRP 10011011 1 CPU Halt 1 CNVTRP 10011011 1 CPU Halt 1 CNVTRP 10001111 1 CPU Halt 1 1 CNUTRP 10111010 2 CPU Halt 1 1 CNUTRP 11111011 1 CPU Halt 1	ction du			Operation Code	n Code			Fla	Flags		
	untenl Gro	Mnemonic	Operand(s)	654321	54321	Bytes			Ч		Z
GETBIT 00001111 01111001 2 Factore priori data start bit and sets it to CY flag. V	suo	SCHEOL		000111	111100	2		 ⊋	Э		×
	itoun	GETBIT		000111	111100	2		 Э	Э		×
$ \begin{array}{ c c c c c c c c c c c c c c c c c c c$	leni x	MHDEC		000111	11110	2		 Э			×
CNVTRP 00001111 01111010 2 Converts line chaole to pixel HALT 11110100 10001111 10011110 2 P0Hat FIDP1 00001111 10011011 2 P0Hat FIDP1 11001101 10011011 2 P0Hat POLL 11110010 2 P0Hat POLL 11111010 1 P0Hat POLL 11111010 2 P0Hat POLL 11111010 1 P0Hat POLL 11111010 1 P0Hat POLL 11111010 2 P0Hat POLL 11111010 2 P0Hat POLL 11111010 2 P0Hat POL 11111010 1 P0Hat POL 11111010 2 P0Hat POL 11111000 1 P0HAT POL 11111000 1 POL POL POL 1 POL POL POL<	eî be	MRDEC		000111	111110	2		 Э			×
HALT 11110100 1 CPU Halt STOP+1 00001111 10011110 2 CPU Stop STOP+1 10011011 10011110 2 CPU Stop POLL 11111010 1 I Poll and wait DI 11111010 1 E 2 CPU Stop DI 11111010 1 1 E 1 E DI 1111000 11111011 11 1 E 1 E 1 DI 11111010 1 11111011 1 E 1 E 1 E 1 E 1 E 1 E 1 E 1 E 1 E 1 1 E 1 E 1 E 1 E 1 E 1 E 1 1 E 1 E 1 E 1 E 1 E 1 E 1 1 1 <	teoibeO	CNVTRP		000111	111101	7	Converts 1 line change point information in change point table to pixel data.	 			
STOP*1 000001111 10011110 2 CPU Stop POLL 10011011 1001110 1 E<0		HALT		111010		۱	CPU Halt	 	_	_	
POLL 10011011 10011011 1 Poll and wait DI 11111010 1 E←0 1 1 Ef 11111011 1 E←0 1 E<0		STOP*1		0001	001111	2	CPU Stop				1
DI 11111010 1111010 1 Et-0 Ef 11111010 1111010 1 Et-1 BUSLOCK 111110000 11111010 1 BusLock Prefix BUSLOCK 111110000 111110000 1 BusLock Prefix BUSLOCK $fp-op$ 111011XXX $11YYYZZZ$ 2 SP-1,SP-2)+PS $fpO1$ $fp-op$ $11011XXX$ $modYYY$ mem 2 to 4 SP-5,SP-6)+PC $fpO1$ $fp-op$ $0110011X$ $11YYYZZZ$ 2 SP-6,SP-6)+PC $fPO2$ $fp-op$ $0110011X$ $11YYYZZZ$ 2 SP-6,SP-6)+PC $fPO2$ $fp-op$ $pO10011X$ $pO1YY$ $pO1001000$ $pO1001000000000000000000000000000000000$	8	POLL		01101		-	Poll and wait			_	
EI 11111011 1 IE-1 BUSLOCK 11110010 11110010 1 Bus Lock Prefix BUSLOCK 11011XXX 11YYYZZZ 2 SP-1, SP-2)+PS $FPO1$ $fp-op$ 11011XXX mod YY mem 2 to 4 SP-5, SP-6)+PC $FPO1$ $fp-op$ 0110011X $11YYYZZZ$ 2 SP-5, SP-6)+PC $FPO2$ $fp-op$ $0110011X$ $11YYYZZZ$ 2 SP-5, SP-6)+PC $FPO2$ $fp-op$ $0110011X$ $mod YYYmem$ 2 SP-6, SP-6)+PC $FPO2$ $fp-op$ $0110011X$ $mod YYYmem$ 2 SP-6, SP-6)+PC $FPO2$ $fp-op$ $mod YYYmem$ $2 SP-6, SP-6)+PC SP-6, SP-6) FPO2 fp-op $	noit	D		11101		۰,	IE←0				
BUSLOCK 11110000 11110000 1 Bus Lock Prefix BUSLOCK fp-op 11011XXX 11YYYZZZ 2 SP-1, SP-2)+PS $fPO1$ fp-op, mem 11011XXX mod YY mem 2 to 4 SP-5, SP-6)+PC $FPO1$ fp-op, mem 0110011X 11YYYZZZ 2 E<0, BRK-0 $FPO2$ fp-op, mem 0110011X mod YY mem 2 to 4 SP-5, SP-6)+PC $FPO3$ fp-op, mem 0110011X 11YYYZZZ 2 E<0, BRK-0 $FPO3$ fp-op, mem 0110011X mod YY mem 2 to 4 SP-5, SP-6)+PC $FPO3$ fp-op, mem 0110011X mod YY mem 2 to 4 SP-6, SP-6)+PC $FPO3$ fp-op, mem 01100110 mod YY mem 2 to 4 SP-6, SP-6)+PC $FPO3$ fp-op, mem 010001111 10010110 Men imm8 imm8 Men imm8 imm8 $RSTWDT*2 imm8, imm8 imm8 imm8 Men imm8 imm8 Men imm8 imm8 RSTWDT*2 imm8, imm8 imm8 imm8 $	struc	EI		11101		٦	lE←1				
FPO1 fp-op 11011XXX $11VYYZZZ$ 2 (SP-1, SP-2)+PS FPO1 fp-op, mem 11011XXX mod YY mem 2 to 4 (SP-5, SP-6)+PG fPO2 fp-op, mem 0110011X 11YYYZZZ 2 (SP-5, SP-6)+PG FPO2 fp-op, mem 0110011X 11YYYZZZ 2 (SP-5, SP-6)+PG FPO3 fp-op, mem 0110011X 11YYYZZZ 2 (SP-5, SP-6)+PG FPO3 fp-op, mem 0110011X nod YY mem 2 to 4 (SP-6, GNH), OICH), OICH), IN NOP mm8, imm8 imm8 imm8 Imm8 Imm8 NOP imm8, imm8 imm8 Imm8 Imm8 Imm8 RSTWDT*2 imm8, imm8 imm8 Imm8 Imm8 Imm8 SS2:	sui lo	BUSLOCK		11000		1	Bus Lock Prefix				
PPO1 fp-op, mem 11011XXX mod Y Y mem 2 to 4 (SP-5, SP-6)+PC $fp-op, mem$ 0110011X 11Y Y Z Z Z 2 E -0 , BRK -0 $fp-op, mem$ 0110011X mod Y Y mem 2 to 4 PC-(01DH; 01CH), NOP 10010000 10010110 10 No Operation 1 No Operation NOP mm8, imm8 00001111 10010110 1 Nen imm8 = imm8 RSTWDT*2 imm8, imm8 00001111 10010110 1 Nen imm8 = imm8 RSTWDT*2 imm8, imm8 00001111 10010110 1 Nen imm8 = imm8 RSTWDT*2 imm8, imm8 imm8 imm8 imm8 imm8 RSTWDT*2 imm8, imm8 1 10010110 1 Segment override f Science 01100011 1 1 1 Segment override f Science 0110001 1 1 1 Extended segment Science 1 1 1 1 1 1	ontro	-041	do-dj	0	ZZ	2	(SP – 1, SP – 2)←PSW, (SP – 3, SP – 4)←PS				
$ \begin{array}{ c c c c c c c } FPO2 & fp \mbox{pop} & fp \mbox{pop} & 0 \mbox{mem} & 0 \mbox{11} \mbox{mod} \mbox{Y} \mbox{mem} & 2 \mbox{mem} & 2 \mbox{pop} & PC \mbox{mem} \mbox{pol} \mbox{mem} & 1 \mbox{pol} \mbox{mem} & 2 \mbox{me} & 2 \mbox{me} & 2 \mbox{me} & 2 \mbox{mem} & 2 \mbox{me} & 2 \mbox{mem} & 2 \mbox{me} & 2 $	o Uq	Ind.	fp-op, mem	101		to 10	(SP – 5, SP – 6)←PC – x*6, SP←SP – 6				T
TPO2 fp-op, mem 0 1 1 0 0 1 1 X mod Y Y mem 2 to 4 PC+(01DH; 01CH). NOP 1 0 0 1 0 0 0 0 1 1 0 1 0 0 0 1 No Operation 1 No Operation NOP 1 0 0 1 0 1 0 1 0 1 1 0 0 1 0 1 0 1 0 1 Nen imm8 = imm8 RSTWDT*2 imm8, imm8, imm8 imm8 imm8 imm8 imm8 RSTWDT*2 imm8, imm8 0 0 0 1 1 1 1 1 0 0 1 0 1 1 0 1 Segment override r RSTWDT*2 imm8, imm8 imm8 imm8 imm8 imm8 imm8 RSTWDT*2 imm8, imm8 imm8 imm8 imm8 imm8 imm8 Stated of 1 0 0 0 1 1 0 0 1 1 1 Segment override r Segment override r DS2: *2 1 1 0 1 0 1 1 1 1 Extended segment IAAM: *2 1 1 1 1 0 0 0 1 1 1 Register file override	D	001	fp-op	1100	1 Y Y Y Z Z	2	IE←0, BRK←0				
NOP 100100 1001000 1000000 1000000 10000000 100000000 1000000000000000000000000000000000000		FFU2	fp-op, mem	110		2 to 4	PC←(01DH; 01CH), PS←(01FH, 01EH)			_	
RSTWDT*2 Imm8, Imm8 Mena imm8 = Imm8 RSTWDT*2 Imm8, Imm8, Imm8 4 When Imm8 = Imm8 4 00001111 10010110 1 Segment override f •4 01100011 01100011 1 Segment override f 053: *2 11010110 11 Extended segment DS3: *2 11110001 1 Retended segment		NOP		00100		ł	No Operation				
KSIWDI*2 Imm8, Imm8 Imm8 Imm8 A •4 0 0 1 sreg1 1 0 1 Segment override p •4 0 1 1 0 0 0 1 1 1 Extended segment override p 053: *2 1 1 0 1 0 1 1 0 1 Extended segment override p IRAM: *2 1 1 1 1 0 0 0 1 1 1 Register file override p				000111	001011	•					
•4 001sreg110 1 DS2: *2 01100011 1 DS3: *2 11010110 1 IRAM: *2 11110001 1	" *	RSTWD1*2	ітт8, ітт8	imm8	<u>imm8</u>	4	15r − 3, 5r − 5)←rC − x −5, 5r←5r − 6 16←0, BRK←0 PC←(20H, 21H), PS←(22H, 23H)				1
DS2: *2 01100011 1 DS3: *2 11010110 1 IRAM: *2 11110001 1		•4		0 1 sreg 1 1		1	Segment override prefix		_		
DS3: *2 11010110 10110 110 110 110 110 110 110		DS2: *2		110001		-	Extended segment override prefix				Ţ
11110001 1		DS3: *2		101011	-	1	Extended segment override prefix				
	*5 *	IRAM: *2		1000		-	Register file override prefix				

Register file space access override prefix instruction x: Number of instruction bytes + number of prefixes

This instruction is newly added to the V25 or V35.

Watchdog timer manipulation instruction

Four kinds: DS0:, DS1:, PS:, SS:

64275250063233717 📟

Powered by ICminer.com Electronic-Library Service CopyRight 2003

18. ELECTRICAL SPECIFICATIONS

This section shows the electrical specifications of the V55Pl[™] using the three categories below.

μPD70433GD: μPD70433 μPD70433GD/R/GJ-12: μPD70433-12 μPD70433GD/R/GJ-16: μPD70433-16

ABSOLUTE MAXIMUM RATINGS (TA = 25 °C)

PARAMETER	SYMBOL	TEST CONDITIONS	RATINGS	UNIT
	Voo		0.5 to +7.0	v
Supply voltage	AVDD		-0.5 to Voc + 0.5	V
	AVss		-0.5 to +0.5	V
	AVREF		-0.5 to AVoo + 0.3	V
Input voltage	Vi		-0.5 to Voo + 0.5	V
Output voltage	Vo		-0.5 to VDD + 0.5	V
Output current low	lor	One pin	4.0	mA
	IOL	Total of all pins	100	mA
Output current high	1	One pin	-1.0	mA
output current nign	юн	Total of all pins	-20	mA
Operating ambient temperature	TA		-40 to +85	°C
Storage temperature	Tstg		-65 to +150	°C

- Notes 1. The IC product output (or input/output) pins should not be directly connected between VDD, Vcc or GND. However, direct connection between the open-drain pins or betwen the open collector pins is possible. Direct connection is also possible for an external circuit via timing design that avoids collision of output at pins which become high impedance.
 - 2. Exceeding the absolute maximum ratings even in one of the parameters even for an instant may affect the product quality.

That is, the absolute maximum ratings are rated values at which the product is on the verge of suffering physical damage, and therefore avoid using the product close to the rated values. The specifications and conditions shown in the DC characteristics and AC characteristics comprise the normal operation and guaranteed quality range.

128

🔜 6427525 OO63234 653 📟

DC Characteristics (TA = -40 to +85 °C, VDD = +5.0 V \pm 10 %)

(1) µPD70433

PARAMETER	SYMBOL	TEST CONDITIONS	MIN.	TYP.	MAX.	UNIT
Input voltage low	VIL1	*1	0		0.8	v
	Vil.2	*2	0		0.2Vpp	v
Input voltage high	ViH1	*1	2.2		Voo	v
	ViHz	*2	0.8Voo		Vdd	V
Schmitt-triggered input threshold voltage	V1*	*3, rise		3.3		v
	V1 ⁻	*3, fall		1.6		v
Schmitt-triggered input hysteresis width	V1 ⁺ - V1 ⁻	•3	0.5			v
Output voltage low	Vol	lot = 2.0 mA			0.45	v
Output voltage high	Vон	loн = −0.4 mA	Voo -1.0			v
Input leakage current	lu	$0 V \leq V_1 \leq V_{00}$			±10	μA
Output leakage current	luo	0 V ≤ Vo ≤ Voo			±10	μA
Voo supply current*4	ÍDD1	Operating mode#5		6.4fx + 30	6.4fx + 50	mA
	looz	HALT mode#5		4.4 fx	4.4fx + 20	mA
	6001	STOP mode*5		10	50	μA
AVoo supply current	Alpos	Operating mode#5		2	10	mA
	Alboz	HALT mode*5		2	10	mA
	Aloos	STOP mode*5		10	50	μA

*1. Other than *2

2. RESET, P10/NMI, X1, P11/INTP0 to P16/INTP5, P30/TxD0/SO0/SB0, P31/RxD0/SB1/SI0, P32/TxC/SCK0, P33/CTS0, P35/RxD1/SI1, P36/SCK1/CTS1

3. RESET, P10/NMI, P11/INTP0 to P16/INTP5

4. The unit for the constants is mA/MHz.

5. $V_{DD} = AV_{DD}$

🖬 6427525 0063235 59T 🔳

(2) µPD70433-12, 70433-16

PARAMETER	SYMBOL	TEST CONDITIONS	MIN.	TYP.	MAX.	UNIT
Input voltage low	ViLi	•1	0		0.8	v
	VIL2	*2	0		0.2Voo	v
Input voltage high	ViH1	*1	2.2		Voo	v
	ViHz	*2	0.8Voo		Voo	v
Schmitt-triggered input threshold voltage	V⊤*	*3, rise		3.3		v
	Vī-	*3, fall		1.6		v
Schmitt-triggered input hysteresis width	V⊤ ⁺ – V⊤ [−]	*3	0.5			v
Output voltage low	Vol	loi = 2.0 mA			0.45	v
Output voltage high	Vон	lон = -0.4 mA	Voo -1.0			v
Input leakage current	lu	0 V ≤ Vi ≤ Voo			±10	μA
Output leakage current	lio	0 V ≤ Vo ≤ Vod			±10	μA
Voo supply current*4	1001	Operating mode		5.4fx + 30	5.4fx + 50	mA
	looz	HALT mode		3.7 fx	3.7fx + 20	mA
	εσσΙ	STOP mode		10	50	μA
AVeo supply current	Aloo1	Operating mode		1.5	2.5	mA
	Alooz	HALT mode		0.6	1	mA
	Aloos	STOP mode		10	50	μA

- *1. Other than *2
- 2. RESET, P10/NMI, X1, P11/INTP0 to P16/INTP5, P30/TxD0/SO0/SB0, P31/RxD0/SB1/SI0, P32/TxC/SCK0, P33/CTS0, P35/RxD1/SI1, P36/SCK1/CTS1
- 3. RESET, P10/NMI, P11/INTP0 to P16/INTP5
- 4. The unit for the constants 5.4 and 3.7 is mA/MHz.

CAPACITANCE (TA = 25 °C, VDD = 0 V)

PARAMETER	SYMBOL	TEST CONDITIONS	MIN.	TYP.	MAX.	UNIT
Input capacitance	Ci	fc = 1 MHz			10	pF
Output capacitance	Co	Unmeasured pins are returned to 0 V.			20	рF
I/O capacitance	Сю				20	pF

OPERATING CONDITIONS

PART NUMBER	INT. CLOCK FREQUENCY	OPERATING TEMPERATURE (TA)	SUPPLY VOLTAGE (VDD)
μΡD70433 μΡD70433-12	0.25 MHz ≤ fx ≤ 12.5 MHz	-40 to +85 °C	+5.0 V ± 10 %
μPD70433-16	0.25 MHz ≤ fx ≤ 12.5 MHz		

130

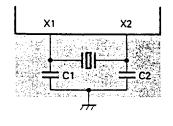
🔲 6427525 OO63236 426 📖

RECOMMENDED OSCILLATION CIRCUIT

The circuit shown below is recommended for a clock input.

(1) µPD70433

(a) Ceramic resonator connection (TA = -10 to +70 °C \pm 10 %, Vob = 5 V \pm 10 %)



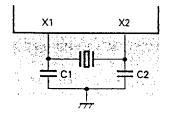
MANUEACTURER	MANUFACTURER FREQUENCY	PRODUCT NAME	RECOMMENDED CONSTANTS			
	fxx [MHZ]	THODOCT NAME	C1 (pF)	C2 [pF]		
Murata Mfg. Co., Ltd.	25	CSA25.00MXZ040	5	5		

Notes 1. The oscillator should be located as close to the X1 and X2 pins as possible.

- 2. Other signal lines should not cross the shaded area.
- 3. When matching the μ PD70433 with a resonator, careful evaluation is required.

(b) Crystal resonator connection

(i) Basic wave recommended condition (TA = -10 to +70 °C, VDD = 5 V \pm 10 %)



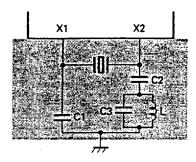
MANUFACTURER	OSCILLATOR NUFACTURER FREQUENCY PI	PRODUCT NAME	RECOMMENDED CONSTANTS			
	fxx [MHZ]	TRODUCT NAME	C1 (pF)	C2 (pF)		
Kinseki	25	HC-49/U-S	10	10 [.]		

Notes 1. The oscillator should be located as close to the X1 and X2 pins as possible.

2. Other signal lines should not cross the shaded area.

3. When matching the μ PD70433 with a resonator, careful evaluation is required.

(ii) 3rd-overtone recommended condition (TA = -20 to +70 °C, VDD = 5 V \pm 10 %)

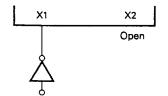


MANUFACTURER	OSCILLATOR FREQUENCY	PRODUCT NAME	RECOMMENDED CONSTANTS					
MANOFACTOREN	fxx [MHZ]	THODOCT NAME	C1 (pF)	C2 [µF]	C3 [pF]	L (µH)		
Kinseki	25	HC-49/U	20	0.01	18	3.3		

Notes 1. The oscillator should be located as close to the X1 and X2 pins as possible.

- 2. Other signal lines should not cross the shaded area.
- 3. When matching the μ PD70433 with a resonator, careful evaluation is required.

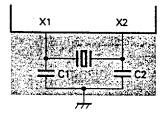
(c) External clock input



🖬 6427525 0063238 2T9 📰

(2) µPD70433-12, 70433-16.

(a) Ceramic resonator connection (TA = -40 to +85 °C \pm 10 %, VDD = 5 V \pm 10 %)



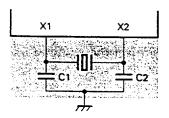
MANUFACTURER FREQUENC	OSCILLATOR		RECOMMENDE	D CONSTANTS
	fxx [MHZ]	PRODUCT NAME	C1 [pF]	C2 (pF)
Murata Mfg. Co., Ltd.	25	CSA25.00MXZ040	5	5
	32	CSA32.00MXZ040	3	3

Notes 1. The oscillator should be located as close to the X1 and X2 pins as possible.

- 2. Other signal lines should not cross the shaded area.
- 3. When matching the μ PD70433 with a resonator, careful evaluation is required.

(b) Crystal resonator connection

(i) Basic-wave recommended condition (TA = -10 to +70 °C, VDD = 5 V ± 10 %)



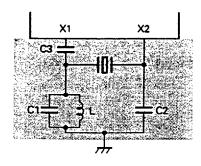
MANUFACTURER	OSCILLATOR FREQUENCY	PRODUCT NAME	RECOMMENDE	D CONSTANTS
	fxx [MHZ]	TRODUCT NAME	C1 (pF)	C2 [pF]
Kinseki	25	HC-49/U-S	5	5
			10	10

Notes 1. The oscillator should be located as close to the X1 and X2 pins as possible.

- 2. Other signal lines should not cross the shaded area.
- 3. When matching the μ PD70433 with a resonator, careful evaluation is required.

🖬 6427525 DD63239 135 📰

(ii) 3rd-overtone recommended condition (TA = -20 to +70 °C, Vpc = 5 V \pm 10 %)

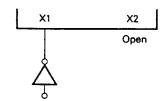


MANUFACTURER	OSCILLATOR FREQUENCY	PRODUCT NAME	RECOMMENDED CONSTANTS					
ManorAcronen	fxx [MHZ]	PRODUCT NAME	C1 (pF)	C2 [µF]	C3 (pF)	L [µH]		
Kinseki	25	HC-49/U	15	15	1000	3.3		
	32		10	5	1000	3.3		

Notes 1. The oscillator should be located as close to the X1 and X2 pins as possible.

- 2. Other signal lines should not cross the shaded area.
- 3. When matching the μ PD70433 with a resonator, careful evaluation is required.

(c) External clock input





AC CHARACTERISTICS (TA = -40 to +85 °C, V_{DD} = +5.0 V ± 10 %)

PARAMETER	SY	MBOL	TEST CONDITIONS	MIN.	MAX.	UNIT
X1 input cycle time	\bigcirc	tovx		40	250	ns
X1 input high-level width	2	twxn		15		ns
X1 input low-level width	3	twxL		15		ns
X1 input rise time	\bullet	txa	· · ·		10	ns
X1 input fall time	5	DXF			10	ns
CLKOUT output cycle time	6	tсук	· · · · · · · · · · · · · · · · · · ·	80	4000	ns
CLKOUT output high-level width	\bigcirc	twich		0.5T – 7		ns
CLKOUT output low-level width	8	twiki		0.5T – 7		ns
CLKOUT output rise time	9	tka			7	ns
CLKOUT output fail time	10	tkr			7	ns
	1	tin 1	*1		10	ns
Input rise time	12	tin:	*2		20	ns
Input fall time	13	t#F1	*1		10	ns
	14	t⊯2	*2		20	ns
Output rise time	15	ton			10	ns
Output fall time	16	tor	· · · · · · · · · · · · · · · · · · ·		10	กร
CLKOUT delay time from X11	118	toxk	X2: open		20	ns
Address delay time from CLKOUT1	$\overline{\mathbb{1}}$	toka		5	30	ns
Address hold time (from CLKOUT ¹)	18	thka1		5		กร
	19	thkaz		5	· · · · · · · · · · · · · · · · · · ·	ns
Address float delay time from CLKOUT	20	tfka		tнказ	40	ns
Address setup time (to ASTB↓)	2	tsast		(n + 0.5)T - 25		ns
Address hold time (from ASTB↓)	22	thsta		0.5T – 15		กร
ASTB↓ delaytime from CLKOUT↓	23	TOKSTL		0	25	ns
ASTB↑ delay time from CLKOUT↓	24	TOKSTH		0	25	ns
ASTB high-level width	25	twsth		(n + 1)T – 15		ns
RD↓ delay time from CLKOUT↑	26	TOKAL		0	25	ńs
RD1 delay time from CLKOUT1	27	токян		0	25	ns
RD low-level width	28	twri		(N + 1.5)T – 15		ns
RD↓ delaytime from address float	29	TFARL		0		ns
Address delay time from RD1	30	tona		0.5T		ns

n: Number of address wait states

N: Number of data wait states

Т : tсук

* 1. Other than *2

2. RESET, P10 NMI, X1, P11/INTP0 to P16/INTP5, P30/TxD0/SO0/SB0, P31/RxD0/SB1/SI0, P32/TxC/SKC0, P33/CTS0, P35/RxD1/SI1, P36/SCK1/CTS1

Remark Numbers in the Symbol column correspond to numbers in the timing charts.

🔲 6427525 0063241 893 🔳

PARAMETER	SY	MBOL	TEST CONDITIONS	MIN.	MAX.	UNIT
ASTBT delay time from RDT, IORDT	119	TORSTH		0		ns
RDT, TORDT delay time from WRLT, WRHT, TOWRT	120	towah		0		ns
DEX delay time from CLKOUT↓	31	tokox		0	30	ns
DEX hold time (from CLKOUT↓)	32	тнкох		0		ns
Dets input setup time (to CLKOUT↓)	33	tsok		15		ns
Data input hold time (from CLKOUT↓)	34	тнкоя		0		ns
WR↓ delay time from CLKOUT↓	35	tokwl		0	25	ns
WRT delay time from CLKOUT↓	36)	tokwh		0	25	ns
WR low-level width	37	twwL		(N + 1)T - 15		ns
Data output delay time from CLKOUT1	38	toko		3	30	ns
Data output hold time (from CLKOUT↓)	39	thkow		0		ns
ASTB1 delay time from WR1	•	towsth		0		ns
RAS↓ delay time from CLKOUT↑	41	TOKRAL		nT	nT + 25	ns
RAST delay time from CLKOUTT	42	TOKRAH		0	25	ns
RAS high-level width	43	twrah		(n + 1)T- 15		ns
RAS↑ delay time from WRH↓, WRL↓	(21)	towaah		(N + 0.5)T - 10		ns
Address setup time (to RAS↓)	(122)	t SARAL		nT – 15		ns
READY setup time (to CLKOUT↓)	••	tsayhk		twкн — 10		កទ
READY hald time (from CLKOUT↓)	45	THKRYL		15		ns
READY setup time (to CLKOUT↓)	46	tsrylk		twкн – 10		ns
READY hold time (from CLKOUT↓)	(47)	thkryh		15		ាន
RESET low-level width	48	twesL1	STOP release/power-on reset	30		ms
	49	twrsL2	System reset	1000 + 2T		ns
NMI high-level width	50	twnin		5		μs
NMI low-level width	51	twnil		5		μs
INTPm setup time (to CLKOUT↓)	(52)	tsiak	m = 0 to 3	30		ns
INTPm high-level width	(53)	twich	m = 0 to 3	10T		ns
INTPm low-level width	54	twiai	m = 0 to 3	10T		ns
POLL setup time (to CLKOUT↓)	55	tsplk		30		ns
HLDRQ setup time (to CLKOUT↓)	56	tshak	······································	30		ns
HLDAK↓ delay time from CLKOUT↑	5 7	ТОКНА	······································	0	30	ns
HLDAK↓ delay time from bus float	5 8	TECHA		0		ns
Bus output delay time from HLDAKT	<u>59</u>	TOHAC		T – 40		ns

n: Number of address wait states

N: Number of data wait states

Т : tсук

Remark Numbers in the Symbol column correspond to numbers in the timing charts.

136

🖬 6427525 0063242 72T 📰

PARAMETER	SY	MBOL	TEST CONDITIONS	MIN.	MAX.	UNIT
HLDAKT delay time from HLDRQ↓	60	tohoha			2.5T + 80	ns
Bus output delay time from HLDRQ↓	61	tonac		0.5T + 50		រាន
HLDRQ low-level width	62	twnoL		2T		ns
HLDAK low-level width	63	twhal	•	3T – 10		ns
BUSLOCK delay time from CLKOUTT	64	toksi		0	30	ns
DMARQm setup time (to CLKOUT↓)	65	tsdak	Except demand release mode; m = 0, 1	30		ns
DMARQm high-level width	66	twoan	Except demand release mode; m = 0, 1	2T		ns
DMARQm low-level width	67	twoal	Except demand release mode; m = 0, 1	2Т		ns
DMARQm setup time (to CLKOUTT)	68	tskog	Demand release mode; m = 0 or 1		5	ns
DMARQm low-level hold time (from CLKOUT↓)	69	<u>тнкро</u>	Demand release mode; m = 0 or 1	15		n s
DMAAKm↓ delay time from CLKOUT↑	70	tokda	m = 0 or 1	0	30	ns.
DMAAKm low-level width	\bigcirc	twoal	m = 0 or 1	(3 + n + N)T - 10		ns
TCEm↓ delay time from CLKOUT1	72	tokte	m = 0 or 1	0	30	ns
TCEm low-level width	73	twicl	m = 0 or 1	T – 10		ns
TOUT high-level width	74	twroн		8T – 10		ns
TOUT low-level width	75	twrol		8T – 10		ns
WDTOUT low-level width	76	twwr.		32T – 10		ns
SCK cycle time	(77)		Input	8T		ns
		tcysk	Output	8T – 10		កទ
SCK high-level width	(78)		Input	4T – 10		ns
		twskh	Output	4T - 10		. ns
SCK low-level width	(79)	twskl	Input	4T – 10		ns
		LWSKL	Output	4T - 10		ns
SI, SB setup time (to SCKT)	80	tsssk		50		ns
SI, SB hold time (from SCK1)	81	thsks		150		ns
SO, SB delay time from SCK1	82	toskse:	IOE mode (CMOS push-pull output)	0	90	ns
So, ob delay time from SCR4	83	tosks82	SBI mode (open-drain output, RL = 1 kΩ)	0	190	ns
SB high-level hold time (from SCK1)	84	tHSKSB		4T		ns
SB low-level setup time (to $\overline{SCK}\downarrow$)	85	tssesk	SBI mode	4T		ns
SB high-level width	86	twsen		4T		ns
SB low-level width	87	twsau		4T		ns

n: Number of address wait states

Т: tсук

Remark Numbers in the Symbol column correspond to numbers in the timing charts.

🔳 6427525 0063243 666 🔳

N: Number of data wait states

PARAMETER	SY	MBOL	TEST CONDITIONS	MIN.	MAX.	UNIT
CTS high-level width	88	tweth		2T		ns
CTS low-level width	89	TWCTL		2T		កទ
Transmit/receive data cycle	90	tcvo	UART	32T		ns
TxC output clock cycle	91	teve		32T		ПS
TxC output clock high-level width	92	twch		16T – 10		ns
TxC output clock low-level width	93	twci	UART	16T – 10		ns
TxD delay time from TxC↓	94	toтсто		0	90	ns
TxD delay time from CTS↓	95	тостто			2tcvc	กร
DATASTB setup time	96	tsosk	Input mode	30		ns
	97	twosui	Input mode	2T		ns
DATASTB low-level width	98	twosu;	Output mode	2T – 10	512T	ns
PD setup time (to DATASTB↓)	99	tspoos:	Input mode	50		ns
PD hold time (from DATASTB↓)	100	THOSPOT	(DATASTB↓ latch mode)	4T		ns
BUSY delay time from DATASTB↓	101	toosay1			4 T	ns
PD setup time (to DATASTB [↑])	102	tspoos2	Input mode	50		ns
PD hold time (from DATASTBT)	103	THDSP02	(DATASTB1 latch mode)	4T		ns
BUSY delay time from DATASTB1	104	toosev2			4T	ns
DATASTBJ delay time from PD	105	toposi		2T - 30	512T	ns
DATASTB setup time (to ACK↓)	(j)	tsdsak		0		ns
ACK input low-level width	107	tw aki,	Output mode	2T		ns
DATASTB setup time (to BUSY ¹)	108	tsdsby		0		ns
BUSY input high-level width	109	twayn		2T		ns
Port output delay time (from CLKOUT↓)	123	toke		10	55	ាន
Port input setup time (to CLKOUT↓)	124	tsex		30		ns
Port input hold time (from CLKOUT↓)	125	ther		20		ns
DMARQm high-level hold time (from ASTB↓)	26	thstdo	Demand release mode; m = 0 or 1	0		ns
REFRQ1 delay time from CLKOUT1	127	TOKREL		0	25	ns
REFRQT delay time from CLKOUTT	128	tokneh		0	25	កទ
RAS delay time from REFROL	(29)	TOMERA	······································	nT – 5		ns

n : Number of address wait states

N: Number of data wait states

Т: tсук

Remark Numbers in the Symbol column correspond to numbers in the timing charts.

🖬 6427525 0063244 ST2 🖬

(2) µPD70433-12

PARAMETER	SYN	MBOL	TEST CONDITIONS	MIN.	MAX.	UNIT
X1 input cycle time	\odot	tcvx		40	250	ns
X1 input high-level width	2	twxн		15		ns
X1 input low-level width	3	twxL		15		ns.
X1 input rise time		txn			10	ns
X1 input fall time	5	DXF.			• 10	ńs
CLKOUT output cycle time	6	tcyx		40	4000	ns
CLKOUT output high-level width	\odot	twich		0.5T – 5	······································	ns
CLKOUT output low-level width	8	twici		0.5T - 5		ns
CLKOUT output rise time	9	tka			7	ns
CLKOUT output fall time	10	tke			7	ns
	\odot	tins	*1		10	ns
Input rise time	12	tiaz	*2		20	ns
	13	tiF1	•1		10	ns
Input fall time	14	tiF2	*2		20	ns
Output rise time	15	ton			10	ns .
Output fall time	16	tor	· · · · · · · · · · · · · · · · · · ·		10	ns
CLKOUT delay time from X11	118	toxk	X2: open		18	ns
Address delay time from CLKOUT1	\bigcirc	t dka		5	27	ាន
	18	thkai		0		ns
Address hold time (from CLKOUT ¹)	19	thka2		0		ns
Address float delay time from CLKOUT?	20	t fka		thka1	36	ns
Address setup time (to ASTB↓)	21	tsast		(n + 0.5)T - 25		ns
Address hold time (from ASTB↓)	22	theta		0.5T – 15		ns
ASTB↓ delay time from CLKOUT↓	23	tokst L		0	22	ns
ASTB↑ delay time from CLKOUT↓	24	toksth		0	22	ns
ASTB high-level width	25	twsтн		(n + 1)T - 15		ns
RD↓ delay time from CLKOUT↑	26	TOKAL		0	22	ns
RD1 delay time from CLKOUT1	27	tokrh		0	22	ns
RD low-level width	28	twal		(N + 1.5)T - 15		ns
RD↓ delay time from address float	29	TFARL		0		ns
Address delay time from RD1	30	tora		0.5T		ns

n: Number of address wait states

N: Number of data wait states

Т: tсук

- * 1. Other than *2
 - 2. RESET, P10 NMI, X1, P11/INTP0 to P16/INTP5, P30/TxD0/S00/SB0, P31/RxD0/SB1/SI0, P32/TxC/SCK0, P33/CTS0, P35/RxD1/SI1, P36/SCK1/CTS1

Remark Numbers in the Symbol column correspond to numbers in the timing charts.

🖬 6427525 OO63245 439 🎟

PARAMETER	SY	MBOL	TEST CONDITIONS	MIN.	MAX.	UNIT
ASTBT delay time from RDT, IORDT	119	TORSTH		0		ns
RDT, IORDT delay time from WRLT, WRHT, IOWRT	120	town		0		ns
DEX delay time from CLKOUT↓	3	tokox		0	27	ns
DEX hold time (from CLKOUT1)	32	тнкох		0		ns
Data input setup time (to CLKOUT↓)	3	tsok		11		ns
Data input hold time (from CLKOUT↓)	34	тнков		0		កទ
WR↓ delay time from CLKOUT↓	35	t DKWL		0	22	nş
WR↑ delay time from CLKOUT↓	36	tokwh		0	22	ns
WR low-level width	37	tww		(N + 1)T - 12		ns
Data output delay time from CLKOUT1	38	toko		3	27	ns
Data output hold time (from CLKOUT↓)	39	THKDW		0		ńs
ASTB [↑] delay time from WR [↑]	40	towsth		0		ns
RAS↓ delay time from CLKOUT↑	(41)	TOKRAL		nT	nT + 22	កទ
RAST delay time from CLKOUTT	42	TOKRAH		0	22	ns
RAS high-level width	43	twrah		(n + 1)T – 15		ns
RAS↑ delay time from WRH↓, WRL↓	121	towsah		(N + 0.5)T - 10		ns
Address setup time (to RAS↓)	(122)	tsaral		nT - 12		ns
READY setup time (to CLKOUT↓)	44	t SRYHK		18		ns
READY hold time (from CLKOUT↓)	45	t hkryl		12		ns
READY setup time (to CLKOUT↓)	46	ts aylk		18		ns
READY hold time (from CLKOUT↓)	(47)	t HKRYH		12		ns
RESET low-level width	48	twest1	STOP release/power-on reset	30		ms
	49	twrsL2	System reset	1000 + 2T		ns.
NMI high-level width	50	twnih		5		μs
NMI low-level width	51	twnil		5		μs
INTPm setup time (to CLKOUT↓)	52	tsiax	m = 0 to 3	25		ns
INTPm high-level width	63	twian	m = 0 to 3	10T		ns
INTPm low-level width	54	twia	m = 0 to 3	10T		ns
POLL setup time (to CLKOUT)	55	tserk		25		ns
HLDRQ setup time (to CLKOUT↓)	66	tsнак		25		ns
HLDAK↓ delay time from CLKOUT1	57	T DKHA		0	27	ns
HLDAK↓ delay time from bus float	58	t FCHA		0		ns
Bus output delay time from HLDAK↑	69	tohac		T - 22.5		ns

n: Number of address wait states

N: Number of data wait states

Т: tсук

Remark Numbers in the Symbol column correspond to numbers in the timing charts.

140

6427525 0063246 375 **6**

PARAMETER	SY	MBOL	TEST CONDITIONS	MIN.	MAX.	UNIT
HLDAKT delay time from HLDRQ↓	60	tongha		0.5T - 15	3.5T + 35	ns
Bus output delay time from HLDRQ↓	61	toнас		0.5T + 45		ns -
HLDRQ low-level width	62	twhat		2T		ns.
HLDAK low-level width	63	twhal		3T – 10		ns
BUSLOCK delay time from CLKOUTT	64	toksi.		0	27	ns
DMARQm setup time (to CLKOUT↓)	65	tsdak	Except demand release mode; m = 0, 1	25		ns
DMARQm high-level width	66	twdah	Except demand release mode; m = 0, 1	2T		ns
DMARQm low-level width	67	twoqu	Except demand release mode; m = 0, 1	2Т		ns
DMARQm setup time (to CLKOUTT)	8	tskog	Demand release mode; m = 0 or 1		5	ns
DMARQm low-level hold time (from CLKOUT↓)	69	t нкро	Demand release mode; m = 0 or 1	12		ns
DMAAKmJ delay time from CLKOUT	70	TDKDA	m = 0 or 1	0	27	ns
DMAAKm low-level width	$\overline{(1)}$	TWDAL	m = 0 or 1	(3 + n + N)T - 10		ńs
TCEm↓ delay time from CLKOUT1	72	EDICTE	m = 0 or 1.	0	27	ns
TCEm low-level width	73	twrcl	m = 0 or 1	T – 10		ns
TOUT high-level width	74	twтон		8T 10		ńs.
TOUT low-level width	75	twtol		8T – 10		ns
WDTOUT low-level width	76	tww⊓∟		32T - 10		ns
SCK cycle time	Ð	tcysk	Input	8T		ns
			Output	8T - 10		ns
SCK bigh lavel width		twskh	Input	4T - 10		ns
SCK high-level width	(78)		Output	4T - 10		ns
SCK low-level width		twskl	Input	4T - 10		ns
	(79)		Output	4T - 10		ns
SI, SB setup time (to SCK1)	80	tsssk		50		ns
SI, SB hold time (from SCKT)	81	tusks		150		ns
SO, SB delay time from SCK↓	82	tosksai	IOE mode (CMOS push-pull output)	0	90	ns
	83	tosksøz	SBI mode (open-drain output, RL = 1 kΩ)	0	190	ns
SB high-level hold time (from \overline{SCKT})	84	LHSKSB	- SBI mode	4T	<u> </u>	ns
SB low-level setup time (to SCK1)	85	tssesx		4T		ns
SB high-level width	86	twsen	······································	4T		ns
SB low-level width	(87)	twsai	· · · · · · · · · · · · · · · · · · ·	4T		ns

n : Number of address wait states

Т: tсук

Remark Numbers in the Symbol column correspond to numbers in the timing charts.

🗰 6427525 0063247 201 🛤

N: Number of data wait states

PARAMETER	SY	MBOL	TEST CONDITIONS	MIN.	MAX.	UNIT
CTS high-level width	88	twoth		2T		ns
CTS low-level width	89	tworu		2T		Пз
Transmit/receive data cycle	90	tero	UART	32T		ns
TxC output clock cycle	91	teve		32T		ns
TxC output clock high-level width	92	twch		16T - 10	· .	ns
TxC output clock low-level width	93	twcL	UART	16T - 10		ns
TxD delay time from TxC↓	94	torcro		0	90	ns
TxD delay time from CTS↓	95	toctro			2tcvc	ns
DATASTB setup time	96	tsdsk	Input mode	25		ns
	97	twosL1	Input mode	2T		лs
DATASTB low-level width	98	twosuz	Output mode	2T 10	512T	ns
PD setup time (to DATASTB↓)	99	tspoosi	Input mode	45		ns
PD hold time (from DATASTB↓)	100	tHDSPD1	(DATASTB↓ latch mode)	4T		ns
BUSY delay time from DATASTB↓	101	toosevi			4T	ns
PD setup time (to DATASTB [†])	102	tspoosz	Input mode	45		ns
PD hold time (from DATASTB1)	103	THOSPD2	(DATASTB [↑] latch mode)	4T		កទ
BUSY delay time from DATASTB1	100	tooseyz			4T	ns
DATASTB↓ delay time from PD	105	topoos L		2T – 30	512T	ns
DATASTB setup time (to ACK↓)	106	tsdsak		0		ns
ACK input low-level width	107	twak L	Output mode	2⊤	····	ns
DATASTB setup time (to BUSYT)	108	t SDSBY		0		ns
BUSY input high-level width	109	twayn		2T	· · · · · · · · · · · · · · · · · · ·	ns
Port output delay time (from CLKOUT↓)	123	toke		8	50	ńs
Port input setup time (to CLKOUT↓)	124	tsek		25		ns
Port input hold time (from CLKOUTJ)	125	tнкр		16		ns
DMARQm high-level hold time (from ASTB↓)	126	thistog	Demand release mode; m = 0 or 1	0		ាទ
REFRQJ delay time from CLKOUT1	127	TOKAEL		0	25	ns
REFROT delay time from CLKOUTT	128	TOKREH		0	25	ns
RAS delay time from REFROL	129	TORERA		. nT – 5		กร
RD↓ delay time from ASTB↓	030	TOSTLAL		0.5T – 5		

n: Number of address wait states

N: Number of data wait states

Т: tсук

Remark Numbers in the Symbol column correspond to numbers in the timing charts.

🖿 6427525 0063248 148 📟

(3) µPD70433-16

PARAMETER	SY	MBOL	TEST CONDITIONS	MIN.	MAX.	UNIT
X1 input cycle time	\bigcirc	tcvx		31.25	250	ns
X1 input high-level width	2	twxn		12		ns
X1 input low-level width	3	two		12		ns
X1 input rise time		ton			5	ns
X1 input fall time	5	txr			5	ns
CLKOUT output cycle time	6	tсук		62.5	4000	ns
CLKOUT output high-level width	\bigcirc	twice		0.5T – 5		ns
CLKOUT output low-level width	8	twice		0.5T – 5		ns
CLKOUT output rise time	٩	tka			5	ns
CLKOUT output fall time	\bigcirc	tio			5	ns
Input rise time	(1)	two	*1		10	ns
	12	tm2	*2		20	ាន
Input fall sime	13	t#1	*1		10	ns
Input fall time	14	t#2	*2		20	កទ
Output rise time	15	ton			10	ns
Output fall time	16	tor			10	ns
CLKOUT delay time from X11	118	toxx	X2: open		18	ns
Address delay time from CLKOUT1	1	tok a		5	27	ns
	18	Тнкат		0		ns
Address hold time (from CLKOUT ¹)	19	thka2		0		ns
Address float delay time from CLKOUT1	20	t fka		tнказ	36	ns
Address setup time (to ASTB↓)	21	tsast		(n + 0.5)T - 25		กร
Address hold time (from ASTB↓)	22	thsta		0.5T – 15		ns
ASTB↓ delay time from CLKOUT↓	23	TOKSTL		0	22	ns
ASTB↑ delay time from CLKOUT↓	24	toksth		0	22	ns
ASTB high-level width	25	twsTH		(n + 1) T – 15		ns
RD↓ delay time from CLKOUT↑	26	EDKRL		0	22	ns
RD1 delay time from CLKOUT1	27	tokni		0	22	ns
RD low-level width	28	twal		(N + 1.5)T - 15		ns
RD↓ delay time from address float	29	TFARL		0		ns
Address delay time from RD1	(30)	TORA		0.5T		ns

n: Number of address wait states

N: Number of data wait states

Т : tсук

- * 1. Other than *2
 - 2. RESET, P10 NMI, X1, P11/INTP0 to P16/INTP5, P30/TxD0/SO0/SB0, P31/RxD0/SB1/SI0, P32/TxC/SKC0, P33/CTS0, P35/RxD1/SI1, P36/SCK1/CTS1

Remark Numbers in the Symbol column correspond to numbers in the timing charts.

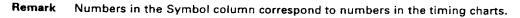
L427525 0063249 084 🖿

PARAMETER	SY	MBOL	TEST CONDITIONS	MIN.	MAX.	UNIT
ASTBT delay time from RDT, TORDT	119	TORSTH		0		ns.
RD1, IORD1 delay time from WRL1, WAH1, IOWR1	120	town		0		ns
DEX delay time from CLKOUT	31	tokox		0	27	ns
DEX hold time (from CLKOUT↓)	32	tнкox	· · · · · · · · · · · · · · · · · · ·	0		ns
Data input setup time (to CLKOUT↓)	33	tsok		11		ns
Data input hold time (from CLKOUT↓)	34	thicda		0		ns .
WRJ delay time from CLKOUTJ	35	torm		0	22	ns
WRT delay time from CLKOUT↓	36	tokwh		0	22	ns
WR low-level width	37	tww		(N + 1)T – 12		ns
Data output delay time from CLKOUT1	38	токо		3	27	ns
Data output hold time (from CLKOUT↓)	39	theow		0		ns
ASTB [↑] delay time from WR [↑]	40	towsth		0	-	ns
RAS↓ delay time from CLKOUT↑		TOKAAL		nT	nT + 22	пs
RAST delay time from CLKOUTT	42	tokrah		0	22	ns
RAS high-level width	43	twaah		(n + 1)T – 15		ns
RAS↑ delay time from WRH↓, WRL↓	121)	towrah		(N + 0.5)T - 10		ns
Address setup time (to RAS↓)	122	tsaral		nT – 12	·······	ns
READY setup time (to CLKOUT↓)	44	ts Ryhk		18	· · · · · · · · · · · · · · · · · · ·	ns
READY hold time (from CLKOUT↓)	45	THKAYL		12		ns
READY setup time (to CLKOUT↓)	46	tsayı k		18		ns
READY hold time (from CLKOUT↓)	(47)	tнклүн		12		ns
RESET low-level width	48	twrsl1	STOP release/power-on reset	30		ms
	49	twrsl2	System reset	1000 + 2T		ns
NMI high-level width	50	twnih		5		μs
NMI low-level width	51	twnil		5		μs
INTPm setup time (to CLKOUT↓)	52	tsiak	m = 0 to 3	25		ns
INTPm high-level width	53	twich	m = 0 to 3	10T		ns
INTPm low-level width	54	twiai	m = 0 to 3	10T		ns
POLL setup time (to CLKOUTJ)	55	tsplk		25		ns
HLDRQ setup time (to CLKOUT↓)	56	tsнак		25		ns
HLDAK↓ delay time from CLKOUT1	চ্চ	tdkha –		0	27	ns
HLDAKJ delay time from bus float	58	TECHA		0		ns
Bus output delay time from HLDAK1	<u>(59)</u>	tonac		T - 22.5		ns

n : Number of address wait states

N: Number of data wait states

Т: tсук



PARAMETER	SY	MBOL	TEST CONDITIONS	MIN.	MAX.	UNIT
HLDAKT delay time from HLDRQ	60	tohoha		0.5T – 15	3.5T + 35	ns
Bus output delay time from HLDRQ↓	61	toнас		0.5T + 45		ns
HLDRQ low-level width	62	twhat		2T		កទ
HLDAK low-level width	63	twhal		3T - 10		n s
BUSLOCK delay time from CLKOUTT	64	T DK8L		0	27	ns
DMARQm setup time (to CLKOUT↓)	65	tsdax	Except demand release mode; m = 0 or 1	25		ns
DMARQm high-level width	66	twdaн	Except demand release mode; m = 0 or 1	2Т		ns
DMARQm low-level width	67	twoau	Except demand release mode; m = 0 or 1	2Τ		ns
DMARQm setup time (to CLKOUT ¹)	68	tskoa	Demand release mode; m = 0 or 1		5	ns
DMARQm low-level hold time (from CLKOUT↓)	69	t нкра	Demand release mode; m = 0 or 1	12		ns
DMAAKm1 delay time from CLKOUTT	70	TOKDA	m = 0 or 1	0	27	ns
DMAAKm low-level width	71	twoa:	m = 0 or 1	(3 + n + N)T – 10		ns
TCEm↓ delay time from CLKOUT↑	72	toкте	m = 0 or 1	0	27	ns
TCEm low-level width	73	twicl	m = 0 or 1	T – 10		ns
TOUT high-level width	74	twтон		8T - 10		ns
TOUT low-level width	75	twtol	· · · · · · · · · · · · · · · · · · ·	8T – 10		ns
WDTOUT low-level width	76	twwtl		32T – 10		ns
SCK cycle time	(77)		Input	8T		ns
		tcysk	Output	8 ⊤ – 10		ns
SCK high-level width			Input	4T - 10		ns
	(78)	. twskh	Output	4T – 10		ns
SCK low-level width			Input	4T 10		ns
	(79)	twskl	Output	4T – 10		ns
SI, SB setup time (to SCKT)	80	tsssk		50		ns
SI, SB hold time (from SCKT)	81	tHSKS		150		កទ
SO, SB delay time from SCK↓	82	tosks#1	IOE mode (CMOS push-pull output)	0	90	រាទ
	83	tdsksb2	SBI mode (open-drain output, RL = 1 kΩ)	0	190	ns
SB high-level hold time (from SCK1)	84	tHSKSB	001	4T		ns
SB low-level setup time (to SCKJ)	85	tssøsk	SBI mode	4T		ns
SB high-level width	86	twsвн	· · · · · · · · · · · · · · · · · · ·	4T		ns
SB low-level width	(87)	twssi		4T		ns

n: Number of address wait states

N: Number of data wait states

Т: tсук

Remark Numbers in the Symbol column correspond to numbers in the timing charts.

🖬 6427525 OO63251 732 🎟

PARAMETER	SY	'MBOL	TEST CONDITIONS	MIN.	MAX.	UNIT
CTS high-level width	88	twcтн		2T		ns
CTS low-level width	89	tworl		2T		ns
Transmit/receive data cycle	90	tcvo	UART	32T		ns
TxC output clock cycle	91	teve		32T		ns
TxC output clock high-level width	92	twch		16T - 10		ns
TxC output clock low-level width	93	twci	UART	16T - 10		ns
TxD delay time from TxC↓	94	tot CTD		0	90	ns
TxD delay time from CTS↓	95	tocrro			2tcvc	ns
DATASTB setup time	96	tsosk	Input mode	25		ns
	97	twosLi	Input mode	2T		ns
DATASTB low-level width	98	twosL2	Output mode	2T – 10	512T	ns
PD setup time (to DATASTB↓)	99	tspddsi	Input mode	45		ns
PD hold time (from DATASTB↓)	100	tHOSPOT	(DATASTB↓ latch mode)	4T		ns
BUSY delay time from DATASTB↓	101	tooseyi			4T	ns
PD setup time (to DATASTB1)	102	tseoosz	Input mode	45		ns
PD hold time (from DATASTB [↑])	103	tHOSPO2	(DATASTB1 latch mode)	4T		ns
BUSY delay time from DATASTB1	104	toosey2			4T	ns
DATASTB↓ delay time from PD	105	TDPDDSL		2T - 30	512T	ns
DATASTB setup time (to ACK↓)	106	tsdsak		0		ns
ACK input low-level width	107	twakl	Output mode	2T		ns
DATASTB setup time (to BUSYT)	108	tsosey		0		ns
BUSY input high-level width	109	twayn		2Т		ns
Port output delay time (from CLKOUT↓)	123	tok#		8	50	ns.
Port input setup time (to CLKOUT↓)	124	tsex		-25		ns
Port input hold time (from CLKOUT↓)	125	tнкр		16		ns
DMARQm high-level hold time (from ASTB↓)	126	thstdo	Demand release mode; m = 0 or 1	0		ns
REFRQ↓ delay time from CLKOUT1	127	TOKAEL		0	25	ns
REFROT delay time from CLKOUTT	128	t dkneh		0	25	ns
RAS delay time from REFRQ1	129	TDAERA		nT - 5		ns
RD↓ delay time from ASTB↓	(130)	TOSTLAL		0.5T – 5		ns

n: Number of address wait states

N: Number of data wait states

Т: tсук

Remark Numbers in the Symbol column correspond to numbers in the timing charts.

146

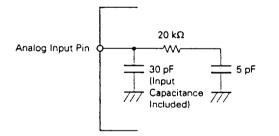
PARAMETER	SYMBOL	TEST CONDITIONS	MIN.	TYP.	MAX.	UNIT
Resolution			8			Bit
Total error #1		-3.4 V S AVREF S AVOD			0.8	%
,		4.5 V S AVREF S AVDO			0.6	%
Quantization error					±1/2	LSB
Conversion time		80 ns ≤ T ≤ 125 ns (for μPD70433, 70433-12)	160T			ns
	tconv	65 ns \leq T \leq 125 ns (for μ PD70433-16)				
		125 ns ≤ T ≤ 250 ns	120T			ńs
Sampling time		80 ns ≤ T ≤ 125 ns (for µPD70433, 70433-12)	32T			ns
	tsamp	65 ns ≤ T ≤ 125 ns (for μPD70433-16)				
		125 ns ≤ T ≤ 250 ns	24T			ns
Analog input voltage			-0.3		AVAEF + 0.3	v
Analog input impedance	P	Non-sampling		1000		MΩ
	RAN	Sampling		+2		
Reference voltage	AVREF		3.4		AVDO	v
AVREF CUTTENT	Alref	T = 80 ns		1.5	5.0	mA

A/D CONVERTER CHARACTERISTICS (Ta= -40 to +85 °C, Vob = +5.0 V ± 10 %, AVss = 0 V, Vob - 0.5 V \leq AVob \leq Vob)

Т: tсук

* 1. Excluding quantization error

2. Analog input impedance is identical with the equivalent circuit shown below. (The values in the figure are not guaranteed, but are TYP. values)



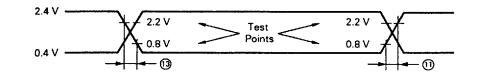
DATA MEMORY STOP MODE LOW SUPPLY VOLTAGE DATA RETENTION CHARACTERISTICS (TA = -40 to +85 °C)

PARAMETER	SYMBOL	TEST CONDITIONS	MIN.	MAX.	UNIT
Data retention supply voltage	115 VODDR	······································	2.5	5.5	v
Supply voltage rise time	(116) tavo	· · · · · ·	200		μs
Supply voltage fall time	(117) t evo	ummati ,	200		μs

Remark Numbers in the Symbol column correspond to number in the timing chart.

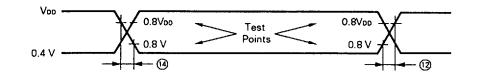
🔳 6427525 0063253 505 🔳

AC Test input Waveform *1



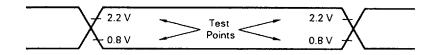
* 1. Except *2

AC Test Input Waveform *2

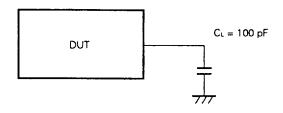


 RESET, P10/NMI, X1, P11/INTP0 to P16/INTP5, P30/TxD0/SO0/SB0, P31/RxD0/SB1/SI0, P32/TxC/SCK0, P33/CTS0, P35/RxD1/SI1, P36/SCK1/CTS1

AC Test Output Test Points



Load Conditions

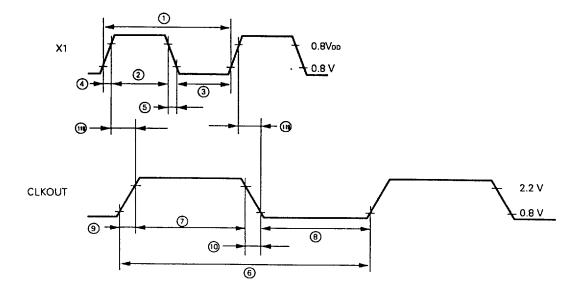


Note If the load capacitance exceeds 100 pF due to the configuration of the circuit, the load capacitance of this device should be reduced to 100 pF or less by insertion of a buffer, etc.

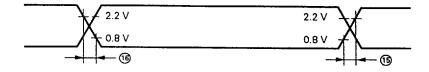
Remark DUT: Measured device

64275250063254441 🎟

Clock Input/Output Timing

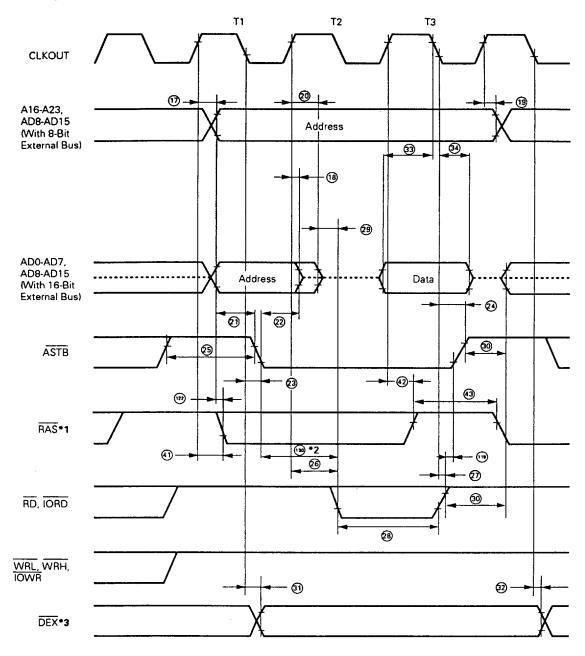


Output Waveform (Except CLKOUT)

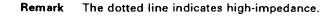


6427525 0063255 388 🔳

Read Timing



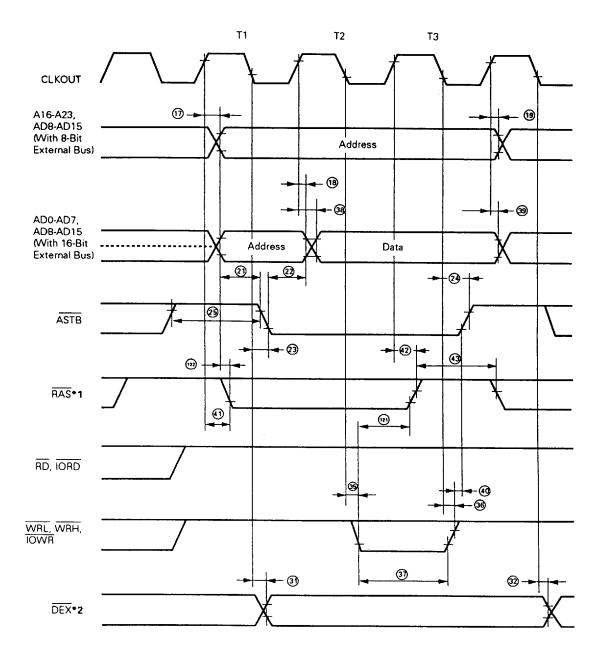
- * 1. Only activated when memory block 1 or 4 (set by the MBS register) is accessed.
 - 2. Applies only to the µPD70433-12, 70433-16.
 - 3. Only valid when the external bus width is 16 bits.



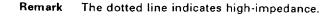
150

6427525 0063256 214 🔳

Write Timing

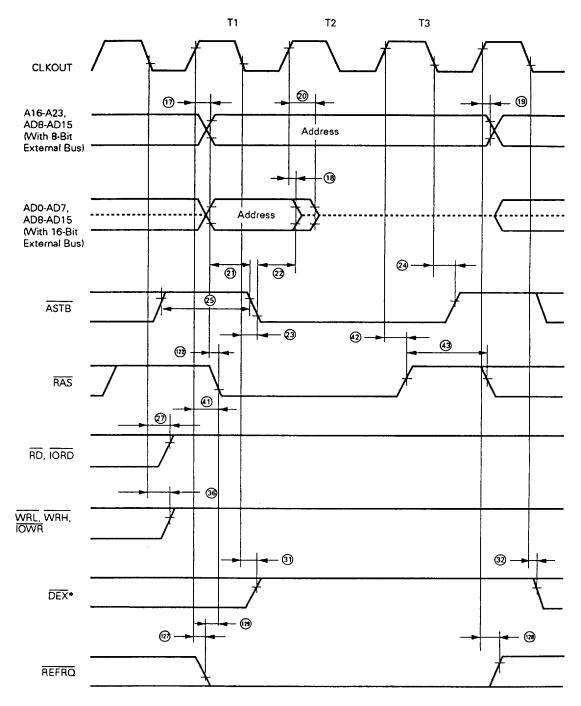


- * 1. Only activated when memory block 1 or 4 (set by the MBC register) is accessed.
 - 2. Only valid when the external bus width is 16 bits.

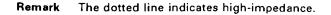


64275250063257150 🖿

Refresh Timing



* Only valid when the external bus width is 16 bits.

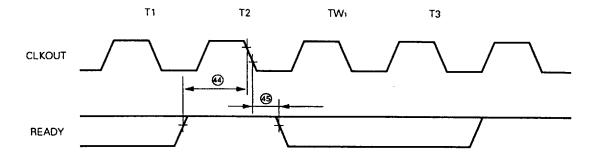


152

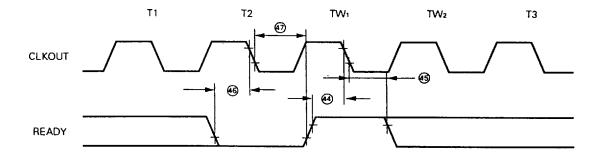
🖬 6427525 OO63258 O97 🖿

Ready Input Timing

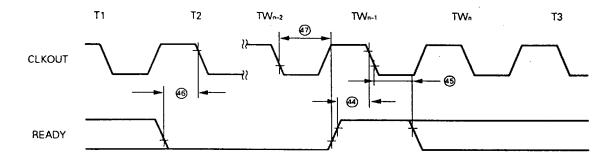
(1) 1 data wait inserted



(2) 2 data waits inserted



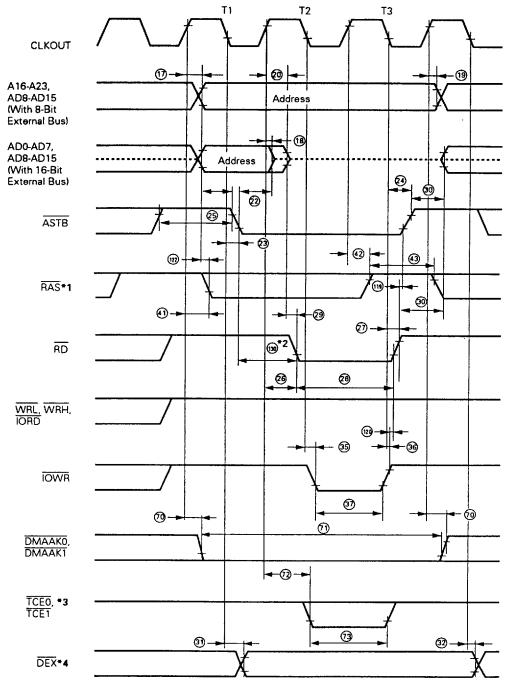
(3) n data waits inserted (n \ge 3)



Remark The READY input becomes valid when the corresponding field of the PWCn register (n = 0 or 1) is other than "00" (binary).



DMA Timing (External Memory \rightarrow External I/O)



- * 1. Only activated when a DMA transfer is performed on memory block 1 or 4 (set by the MBC register).
 - 2. Applies only to the µPD70433-12, 70433-16.
 - 3. The bus is activated at the last transfer in intelligent DMA mode-2, 2-channel operating mode (stop in termination), or memory-to-memory transfer mode (stop in termination).
 - 4. Only valid when the external bus width is 16 bits.

Remark The dotted line indicates high-impedance.

154

6427525 0063260745 🖿

Τ1 T2 T3 CLKOUT A16-A23, AD8-AD15 0 ୭ Address With 8-Bit External Bus) AD0-AD7. 13 AD8-AD15 Address With 16-Bit ⑳ External Bus) 60 0 2 Ø ASTB න 42 43 1 T RAS*1 ٩ 1 RD, IOWR 0 3 59 WRL, WRH 60 3 0 120 *2 IORD 130 ହ Ø 3 @ 1 0 DMAAKO, DMAAK1 **-**12→ TCE0, *3 TCE1 **(73**) 9 3 DEX*4

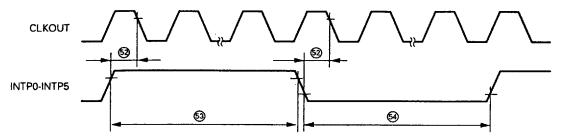
DMA Timing (External I/O → External Memory)

- 1. Only activated when a DMA transfer is performed on memory block 1 or 4 (set by the MBC register).
- 2. Applies only to the µPD70433-12, 70433-16.
- 3. The bus is activated at the last transfer in intelligent DMA mode-2, 2-channel operating mode (stop in termination), or memory-to-memory transfer mode (stop in termination).
- 4. Only valid when the external bus width is 16 bits.

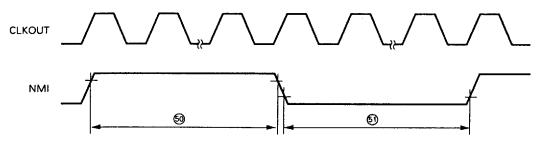
Remark The dotted line indicates high-impedance.

6427525 0063261 681 🛲

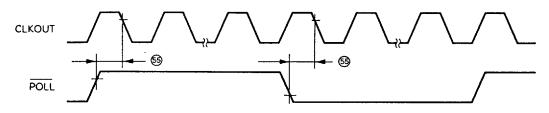
INRPm Input Timing (m = 0 to 5)



NMI Input Timing



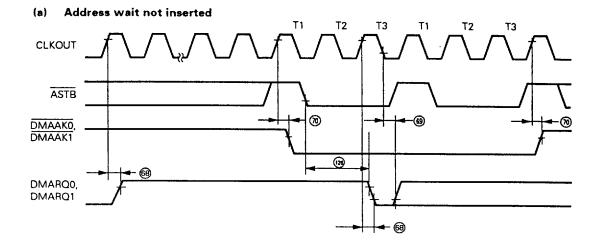
POLL Input Timing



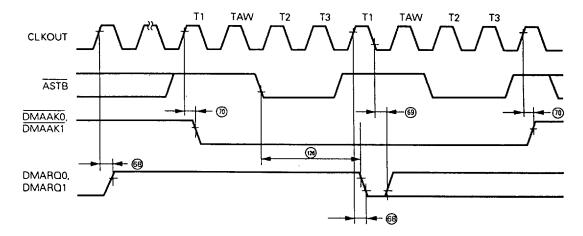
64275250063262518 📟

DMAROm Input Timing (m = 0 or 1)

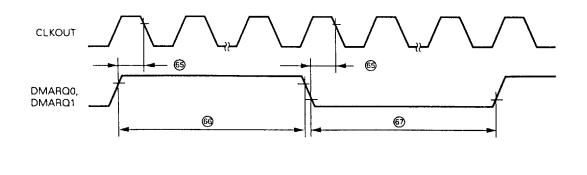
(1) In demand release mode (I/O-to-memory transfer)



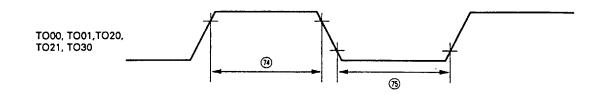
(b) Address wait inserted



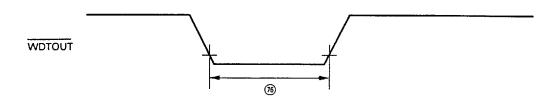
(2) In the mode other than demand release mode



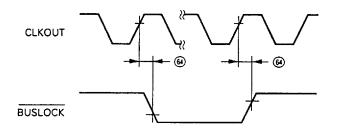
Timer Output Timing



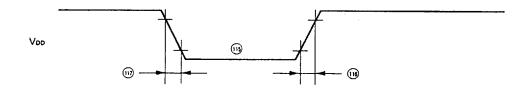
WDTOUT Output Timing



BUSLOCK Output Timing



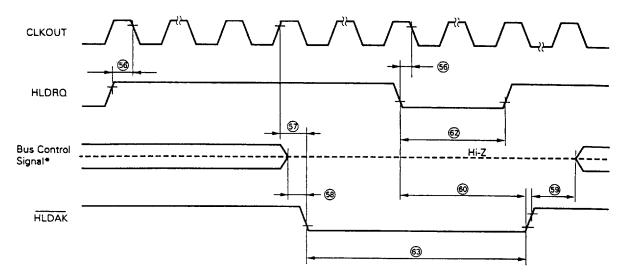
Data Retention Timing (STOP Mode)





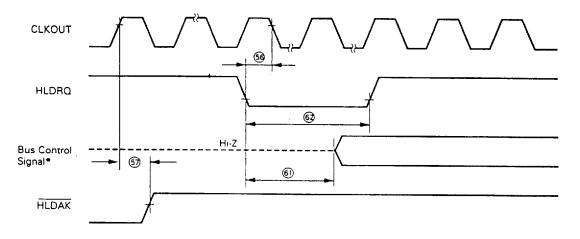
Hold Request/Acknowledge Timing





· ASTB, RD, WRH, WRL, DEX, RAS, BUSLOCK, IORD, IOWR, AD0 to AD15, A16 to A23

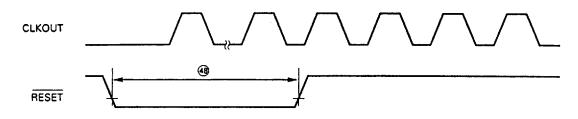
(2) Release of hold mode for refresh cycle insertion



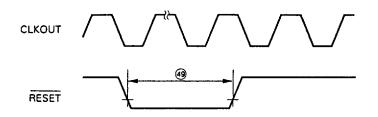
* ASTB, RD, WRH, WRL, DEX, RAS, BUSLOCK, IORD, IOWR, AD0 to AD15, A16 to A23

RESET Input Timing

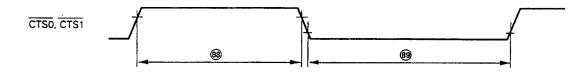
(1) STOP mode release/power-on reset



(2) System reset



CTSm Input Timing (m = 0 or 1)

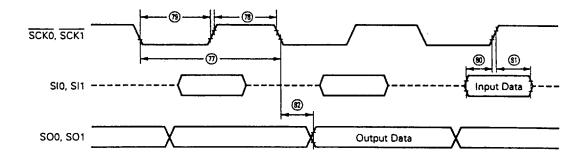


1**6**0

🖿 6427525 0063266 163 🎟

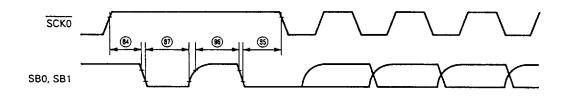
Serial Interface Timing

(1) 3-wire serial I/O mode

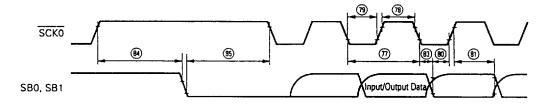


(2) SBI mode

Bus release signal transfer timing



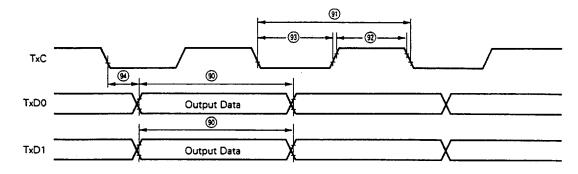
Command signal transfer timing



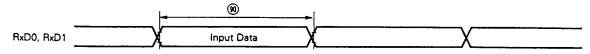
🔲 6427525 0063267 OTT 📰

(3) UART mode

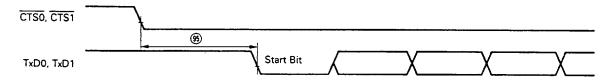
Transmit timing



Receive timing



Transmission enale timing

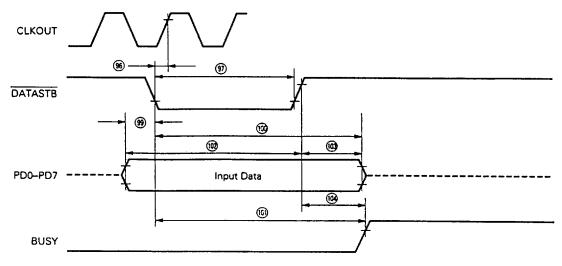


162

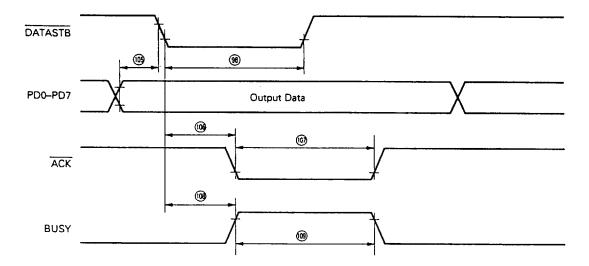
🔳 6427525 0063268 T36 🎟

Parallel Interface Timing

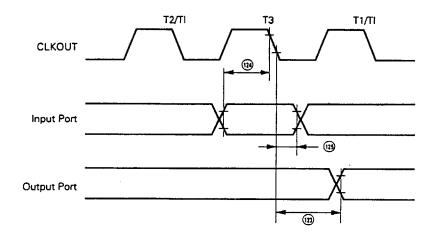
(1) Input mode



(2) Output mode



Port Input/Output Timing

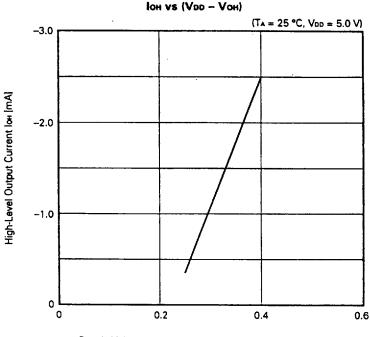


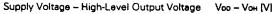


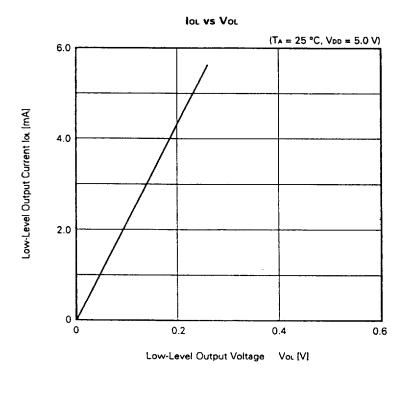
🗰 6427525 OO63270 694 🎟

19. CHARACTERISTIC CURVES (FOR REFERENCE ONLY)

(1) µPD70433GD

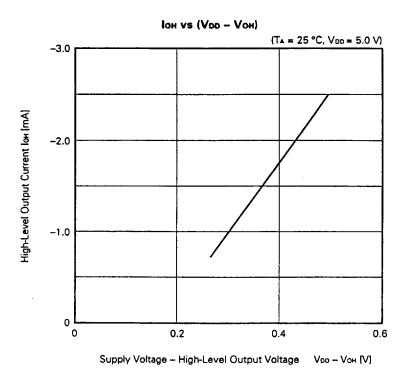


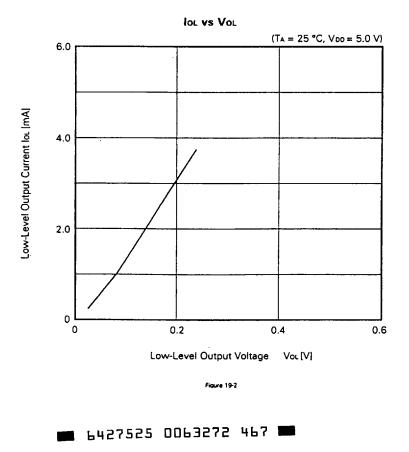




🖬 6427525 OO63271 520 📟

(2) µPD70433GD/R/GJ-12, 70433GD/R/GJ-16



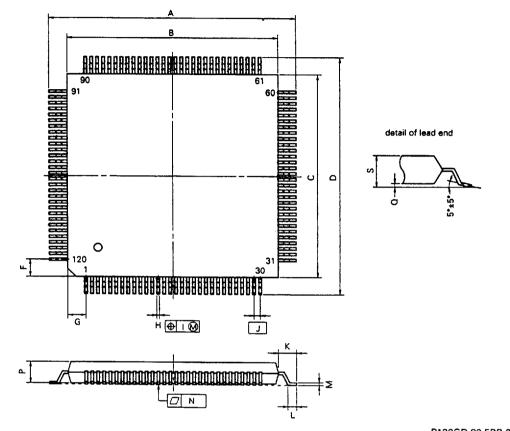


Powered by ICminer.com Electronic-Library Service CopyRight 2003

166

20. PACKAGE DRAWINGS

120 PIN PLASTIC QFP (28)



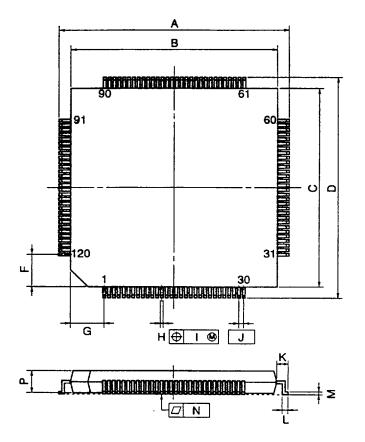
NOTE

Each lead centerline is located within 0.15 mm (0.006 inch) of its true position (T.P.) at maximum material condition.

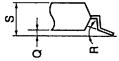
		P120GD-80-5BB-3
ITEM	MILLIMETERS	INCHES
А	32.0±0.4	1.260±0.016
в	28.0±0.2	1.102+0.009
С	28.0±0.2	1.102+0.009
D	32.0±0.4	1.260±0.016
F	2.4	0.094
G	2.4	0.094
н	0.35±0.10	0.014+0.004
Ι	0.15	0.006
J	0.8 (T.P.)	0.031 (T.P.)
к	2.0±0.2	0.079-0.009
L	0.8±0.2	0.031+0.009
м	0.15+0.10	0.006±0.004
N	• 0.10	0.004
Ρ	3.7	0.146
a	0.1±0.1	0.004±0.004
S	4.0 MAX.	0.157 MAX.

🔲 6427525 0063273 3T3 🎆

120 PIN PLASTIC QFP (FINE PITCH) (20)



detail of lead end



NC)TE
----	-----

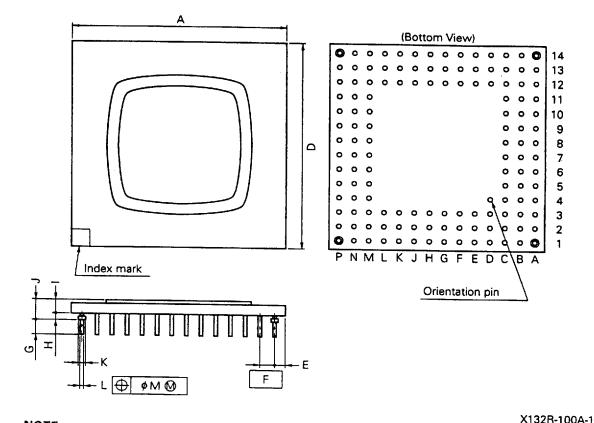
Each lead centerline is located within 0.10 mm (0.004 inch) of its true position (T.P.) at maximum material condition.

ITEM	MILLIMETERS	INCHES
<u> </u>	22.0±0.2	0.866±0.008
9	20.0±0.2	0.787+0.009
С	20.0±0.2	0.787+0.009
D	22.0±0.2	0.866±0.008
F	2.75	0.108
G	2.75	0.108
н	0.22+0.05	0.009±0.002
1	0.10	0.004
J	0.5 (T.P.)	0.020 (T.P.)
к	1.0±0.2	0.039+0.009
L	0.5±0.2	0.020+0.008
м	0.17+0.03	0.007+0.001 -0.003
N	0.10	0.004
P	2.7	0.106
a	0.125±0.075	0.005±0.003
R	5°±5°	5°±5°
S	3.0 MAX.	0.119 MAX.
		S120GJ-50-3EB-2

168

6427525 0063274 23T

132 PIN CERAMIC PGA



NOTE

Each lead centerline is located within ϕ 0.5 mm (ϕ 0.020 inch) of its true position (T.P.) at maximum material condition.

ITEM	MILLIMETERS	INCHES
А	35.56±0.3	1.400±0.012
D	35.56±0.3	1.400±0.012
E	1.27	0.050
F	2.54 (T.P.)	0.100 (T.P.)
G	2.8±0.3	0.110±0.012
н	0.9 MIN.	0.035 MIN.
1	2.95	0.116
J	4.57 MAX.	0.180 MAX.
к	Ø1.2±0.2	\$0.047 ^{+0.009}
L	Ø0.46±0.05	¢0.018±0.002
м	0.254	0.010

🖿 6427525 OO63275 176 🖿

21. RECOMMENDED SOLDERING CONDITIONS

This product should be soldered and mounted under the conditions recommended in the table below.

For details of recommended soldering conditions, refer to the information document "Semiconductor Device Mounting Technology Manual" (IEI-1207).

For soldering methods and conditions other than those recommended, contact our sales personnel.

Table 21-1. Surface Mount Type Soldering Conditions

μ PD70433GD-5BB :	120-Pin Plastic QFP (28 × 28 mm)
	120-Pin Plastic QFP (28 × 28 mm)
μPD70433GJ-xx-3EB :	120-Pin Plastic QFP (Fine Pitch) (20 $ imes$ 20 mm)

Soldering Method	Solderring Conditions	Recommended Condition Symbo
Infrared reflow	Package peak temperature: 235 °C, Duration: 30 sec. max. (at 210 °C or above), Number of times: Within twice, Time limit: 7 days* (thereafter 36 hours 125 °C prebanking required)	IR35-367-2
VPS	Package peak temperature: 215 °C, Duration: 40 sec. max. (at 200 °C or above), Number of times: Within twice, Time limit: 7 days* (thereafter 36 hours 125 °C prebanking required)	VP15-367-2
Wave soldering	Solder bath temperature: 260 °C or less, Time: 10 sec. max., Number of times: Once, Time limit: 7 days* (thereafter 35 hours 125 °C prebanking required), Preheating temperature: 120 °C max. (Package surface temperature)	WS60-367-1
Partial heating	Pin temperature: 300 °C or below, Duration: 3 sec. max. (per pin row)	

For the storage period after dry-pack decompression, storage conditions are max. 25 °C, 65 % RH.

Note Use of more than one soldering method should be avoided (except in the case of partial heating method).

Table 21-2. Insertion Type Soldering Conditions

μPD70433R-xx : 132-Pin Ceramic PGA

Soldering Method	Solderring Conditions
Wave soldering (lead part only)	Solder temperature: 260 °C or less, Duration: 10 sec. max.
Partial heating	Pin temperature: 300 °C or less, Duration: 3 sec. max. (per pin row)

Note Wave soldering is used on the lead part only, and care must be taken to prevent solder from coming into direct contact with the body.

170

🔳 6427525 0063276 002 📕