RAiO

RA8906

8-Bit Micro-Controller

Version 2.4

November 9, 2004

RAiO Technology Inc.

Chapter	Table of Contents	Page

1. Overview

The RA8906 is a 8-bit e-MCU micro-controller with 64K-byte embedded Flash ROM. It supports multiple timer/counter sources, versatile interrupt-handling architecture and two built-in DAC's (Digital-to-Analog Converters). It provides a complete speech interface and 32K-bps ADPCM solution that make this chip an excellent choice as the embedded micro-controller for educational game or toy speech products.

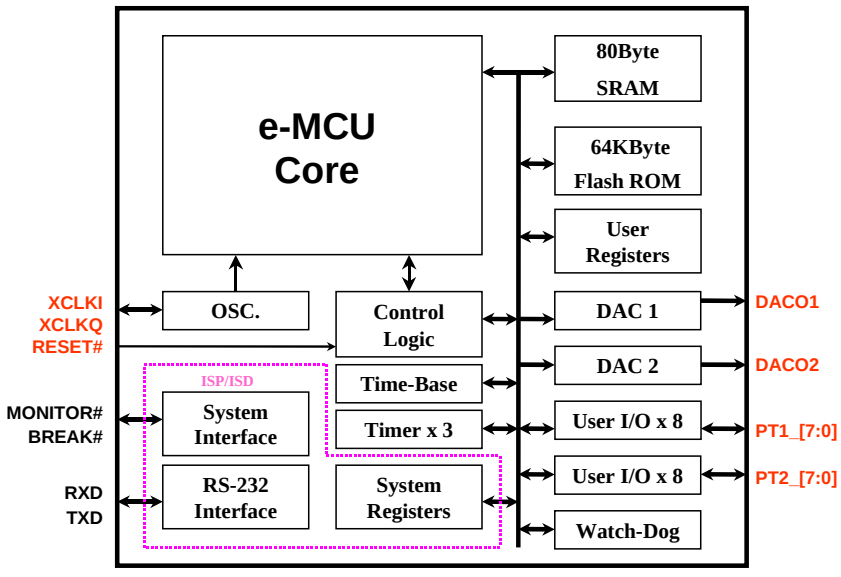
The built-in 4K-bytes ROM support the on-chip RAiO ICE Monitor program which controls the UART and enables the RS232 connection between the RA8906 and a PC host. The RA8906 support the ISP (In-System Programming) and ISD(In-System Debugging) functions. Users can download their programs as well as data from a PC host to the embedded Flash ROM of RA8906.

RAiO also support a windows based ICE driver for customers to very easy programming & debugging their program. Using RS232 connection between a speech toy to a PC host running RAiO's download utility program, toy makers are able to implement the Internet Game/Speech download features easily and give the toys multiple attractive characteristics.

2. Feature

- ◆ 8-bit e-MCU CPU
- ◆ 64K-byte Embedded Flash ROM
- ◆ 80-byte User SRAM
- ◆ Two 8-bit General Purpose I/O Ports
- ◆ Three 12-bit Timers
- ◆ Three Fixed Time-Bases (2KHz, 500Hz, 62Hz)
- ◆ Two 8-bit Current Mode DACs
- ◆ Support One Watch Dog Timer
- ◆ Low Power Consumption at Sleep Mode
- ◆ On-Chip ICE and ISP(In-System Programming) Supporting Programs.
- ◆ Support 32K bps ADPCM Solution
- ◆ Operating Voltage: 4.5V ~ 5.5V
- ◆ System Clock: 2/4/6MHz

3. Block Diagram



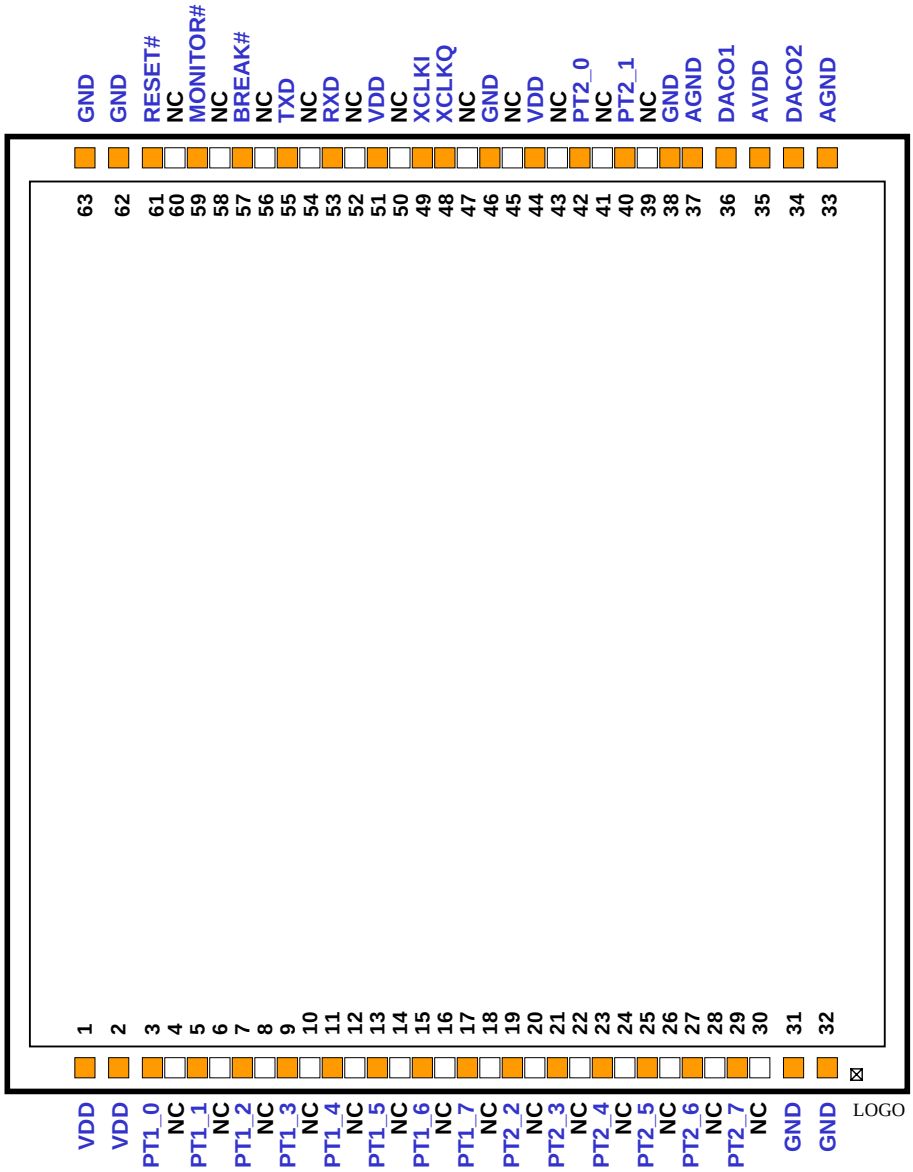
4. Package

4.1 PAD X/Y Coordinate

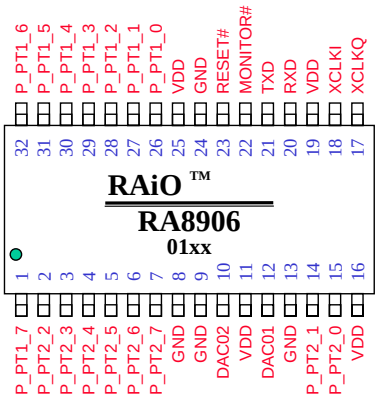
Order	PIN Name	X	Y
1	VDD	-1509.25	-2084.50
2	VDD	-1369.75	-2084.50
3	PT1_0	-1230.25	-2084.50
5	PT1_1	-1047.25	-2084.50
7	PT1_2	-864.25	-2084.50
9	PT1_3	-681.25	-2084.50
11	PT1_4	-498.25	-2084.50
13	PT1_5	-315.25	-2084.50
15	PT1_6	-132.25	-2084.50
17	PT1_7	50.75	-2084.50
19	PT2_2	233.75	-2084.50
21	PT2_3	416.75	-2084.50
23	PT2_4	599.75	-2084.50
25	PT2_5	782.75	-2084.50
27	PT2_6	965.75	-2084.50
29	PT2_7	1148.75	-2084.50
31	GND	1379.75	-2084.50
32	GND	1519.25	-2084.50

Order	PIN Name	X	Y
33	AGND	1570.80	2084.50
34	DACO2	1413.70	2084.50
35	AVDD	1270.10	2084.50
36	DACO1	1126.49	2084.50
37	AGND	969.40	2084.50
38	GND	871.00	2084.50
40	PT2_1	688.00	2084.50
42	PT2_0	505.00	2084.50
44	VDD	322.00	2084.50
46	GND	139.00	2084.50
48	XCLKQ	-60.50	2084.50
49	XCLKI	-180.50	2084.50
51	VDD	-363.50	2084.50
53	RXD	-546.50	2084.50
55	TXD	-729.50	2084.50
57	BREAK#	-912.50	2084.50
59	MONITOR#	-1095.50	2084.50
61	RESET#	-1278.50	2084.50
62	GND	-1418.00	2084.50
63	GND	-1557.50	2084.50

4.2 PAD Diagram

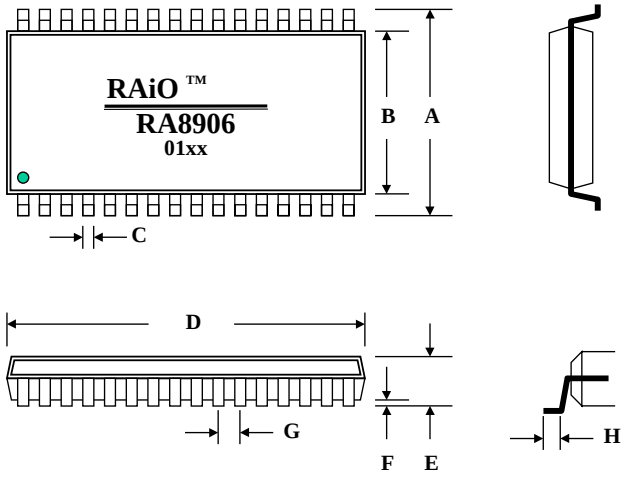


4.3 SOP-32Pin Package

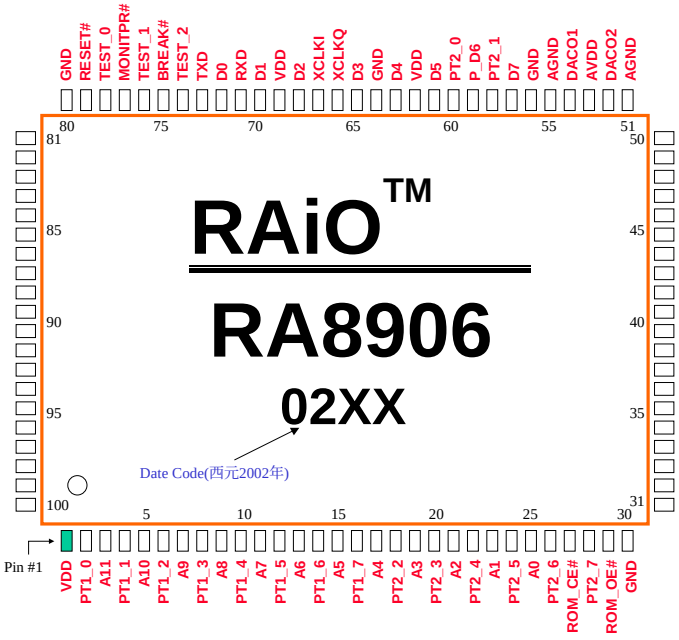


Symbols	Min.	Max.
A	0.530	0.580
B	0.437	0.450
C	--	0.015
D	0.799	0.815
E	--	0.120
F	0.004	0.014
G	--	0.050
H	0.016	0.050

Unit: Inch



4.4 PQFP-100Pin Package



5. Pin Description

Signal	I/O	Description
RESET#	IN	**External Hardware Reset, active low.** This pin is used to reset the system.
BREAK#	IN	**User Program Break, active low.** This signal is used to break the user's program from the ISD mode.
MONITOR#	IN	**Monitor Program Select, active low.** This signal is used to select the system boot from monitor program (ROM) or user program (Flash). This signal has to pull low when the user wants to download the data from PC or enter the ISP/ISD mode.
PT1_[7:0]	I/O	**8-bit I/O of Port 1** These signals are used as general-purpose I/O port. The PORT1[7] is also as the Timer1 clock source when REG $0Ah bit2-0 = '111' The PORT1[6] is also as the Port1 NMI or INT trigger source when REG $00h bit5 = '1', or REG $02h bit5 = '1'. PT1_[7:0] provide 8mA driving current.
PT2_[7:0]	I/O	**8-bit I/O of Port 2** These signals are used as general-purpose I/O port. The PT2_7 is also as the Timer2 clock source when REG $0Bh bit2-0 = '111' The PT2_6 is also as the Port2 NMI or INT trigger source when REG $00h bit4 = '1', or REG $02h bit4 = '1'. The PT2_3 is also as the wake-up trigger source when the chip entered the power saving mode. PT2_[7:2] provide 4mA driving current. PT2_[1:0] provide 16mA driving current.
DACO1	OUT	**DAC1 Current Output** In the DAC mode, this pin is the current output of DAC1. In the PWM mode, this pin is the output of PWM.
DACO2	OUT	**DAC2 Current Output** In the DAC mode, this pin is the current output of DAC2. In the PWM mode, this pin is the complementary output of PWM.
RXD	IN	**Receive Data** This is the received data input of UART. Normally it's connected to the RS232's TX of PC.
TXD	OUT	**Transmit Data** This signal is the transmitted data output of UART. Normally it's connected to the RS232's RX of PC.
XCLKI	IN	**Oscillator Input.** This is the input signal of internal oscillator.
XCLKQ	OUT	**Oscillator Output.** This is the output signal of internal oscillator.
A[11:0]	OUT	**Address Bus.** These pins are used for external system ROM address. Normally, users will not use these pins.
D[7:0]	IN	**Data Bus.** These pins are used for external system ROM data. Normally, users will not use these pins.
ROM_CE#	OUT	**ROM Chip Enable.** These pins are used for external system ROM chip enable. Normally, users will not use these pins.

ROM_OE#	OUT	**ROM Output Enable.** These pins are used for external system ROM output enable. Normally, users will not use these pins.
TEST_[2:0]	IN	**Test Pins**. These are for RAiO's internal testing purpose used in testing IC and ROM status. Normally, users will not use these pins.
VDD	PWR	**Power Supply Voltage.**
AVDD	PWR	**Analog Power Supply Voltage.**
GND	PWR	**Ground.**
AGND	PWR	**Analog Ground.**

6. Registers Description

Address	Register	7	6	5	4	3	2	1	0	R/W
$00h	NMI_MK	TM1	TM2	PT1_6	PT2_6	TM3	WTD	--	--	R/W
$01h	NMI_ST	TM1	TM2	PT1_6	PT2_6	TM3	WTD	--	--	R/W
$02h	INT_MK	TM1	TM2	PT1_6	PT2_6	TM3	TB-2K	TB-500	TB-62	R/W
$03h	INT_ST	TM1	TM2	PT1_6	PT2_6	TM3	TB-2K	TB-500	TB-62	R/W
$04h	TM1_H	--	--	--	--	D11	D10	D9	D8	R/W
$05h	TM1_L	D7	D6	D5	D4	D3	D2	D1	D0	R/W
$06h	TM2_H	--	--	--	--	D11	D10	D9	D8	R/W
$07h	TM2_L	D7	D6	D5	D4	D3	D2	D1	D0	R/W
$08h	TM3_H	--	--	--	--	D11	D10	D9	D8	R/W
$09h	TM3_L	D7	D6	D5	D4	D3	D2	D1	D0	R/W
$0Ah	TM1_CTL	TM1_EN	TM1_INT_EN	TM2_INT_EN	TM3_INT_EN	TM1_LOOP	CKS2	CKS1	CKS0	R/W
$0Bh	TM2_CTL	TM2_EN	2KHZ_INT_EN	500HZ_INT_EN	62HZ_INT_EN	TM2_LOOP	CKS2	CKS1	CKS0	R/W
$0Ch	TM3_CTL	TM3_EN	--	--	--	TM3_LOOP	CKS2	CKS1	CKS0	R/W
$0Dh	PT1_DAT	D7/TM1_CK	D6/PT1_TG	D5/IR	D4/PWM	D3	D2	D1	D0	R/W
$0Eh	PT1_DIR	D7	D6	D5	D4	D3	D2	D1	D0	R/W
$0Fh	PT1_MOD	D7	D6	D5	D4	D3	D2	D1	D0	R/W
$10h	PT2_DAT	D7/TM2_CK	D6/PT2_TG	D5/TX	D4/RX	D3/WK_UP	D2	D1	D0	R/W
$11h	PT2_DIR	D7	D6	D5	D4	D3	D2	D1	D0	R/W
$12h	DAC1_DAT	D7	D6	D5	D4	D3	D2	D1	D0	R/W
$13h	DAC2_DAT	D7	D6	D5	D4	D3	D2	D1	D0	R/W
$14h	DAC_CTL	DAT2	DAT1	OE2	OE1	PWM-DAC	--	DAC2-AUTO	DAC1-AUTO	R/W
$15h	PWM	PWM_EN	D6	D5	D4	D3	D2	D1	D0	R/W
$16h	FBANK	D7	D6	D5	D4	D3	D2	D1	D0	R/W
$17h	PWR_CTL	PT1	PT2	FLASH	STOP_CLK	HI_SPD	--	--	--	R/W
$18h	RST_ST	POR	RST#	WTD	S/W	--	--	--	--	R/W
$19h	CLK_CTL	CPU	DIV1	DIV0	WTD	TBS1	TBS0	32KS1	32KS1	R/W
$1Ah	FL_CTL	MD3	MD2	MD1	MD0	PROG	PA2	PA1	PA0	R/W
$1Bh	MISC_CTL	WDT_EN	WTD_LOOP	--	--	WTD_RST_EN	S/W_RST_EN	IR-EN	IR-DAT	R/W

[REG $00h]: NMI Mask Register, NMI_MK

Bit	Description	Reset	Default	Access
7	Timer 1 NMI Enable, 1: Enable, 0: Disable	0h	0h	R/W
6	Timer 2 NMI Enable, 1: Enable, 0: Disable	0h	0h	R/W
5	Port 1 bit6 NMI Enable, 1: Enable, 0: Disable	0h	0h	R/W
4	Port 2 bit6 NMI Enable, 1: Enable, 0: Disable	0h	0h	R/W
3	Timer 3 NMI Enable, 1: Enable, 0: Disable	0h	0h	R/W
2	Watch Dog NMI Enable, 1: Enable, 0: Disable	0h	0h	R/W
1-0	Reserved	--	--	--

Example:

```
LDA   #$C0
STA   $00h      ; permit Timer1and Timer2 to produce NMI Interrupt
```

[REG $01h]: NMI Status Register, NMI_ST

Bit	Description	Reset	Default	Access
7	Timer 1 NMI Indicate	0h	0h	R/W
6	Timer 2 NMI Indicate	0h	0h	R/W
5	Port 1 bit6 NMI Indicate	0h	0h	R/W
4	Port 2 bit6 NMI Indicate	0h	0h	R/W
3	Timer 3 NMI Indicate	0h	0h	R/W
2	Watch Dog NMI Indicate	0h	0h	R/W
1-0	Reserved	--	--	--

Example 1:

```
LDA   $01h      ; can be used to diagnose the source of NMI interrupt
```

Example 2:

```
LDA   #$00h
STA   $01h      ; to eliminate the interrupt indication of NMI
```

[REG $02h]: INT Mask Register, INT_MK

Bit	Description	Reset	Default	Access
7	Timer 1 INT Enable, 1: Enable, 0: Disable	0h	0h	R/W
6	Timer 2 INT Enable, 1: Enable, 0: Disable	0h	0h	R/W
5	Port 1 bit6 INT Enable, 1: Enable, 0: Disable	0h	0h	R/W
4	Port 2 bit6 INT Enable, 1: Enable, 0: Disable	0h	0h	R/W
3	Timer 3 INT Enable, 1: Enable, 0: Disable	0h	0h	R/W
2	2KHZ Time Base INT Enable, 1: Enable, 0: Disable	0h	0h	R/W
1	500HZ Time Base INT Enable, 1: Enable, 0: Disable	0h	0h	R/W
0	62HZ Time Base INT Enable, 1: Enable, 0: Disable	0h	0h	R/W

Example:

```
LDA   #$C0
STA   $02h      : permit Timer 1and Timer 2 to produce interrupt
```

[REG $03h]: INT Status Register, INT_ST

Bit	Description	Reset	Default	Access
7	Timer 1 INT Indicate	0h	0h	R/W
6	Timer 2 INT Indicate	0h	0h	R/W
5	Port 1 INT Indicate	0h	0h	R/W
4	Port 2 INT Indicate	0h	0h	R/W
3	Timer 3 INT Indicate	0h	0h	R/W
2	2KHZ Time Base INT Indicate	0h	0h	R/W

1	500HZ Time Base INT Indicate	0h	0h	R/W
0	62HZ Time Base INT Indicate	0h	0h	R/W

Example 1:

```
LDA   $03h      ; can be used to diagnose the source of NMI interrupt
```

Example 2:

```
LDA   #$00h
STA   $03h      ; to eliminate the interrupt indication of NMI
```

[REG $04h]: Timer 1 Count_H Register, TM1_H

Bit	Description	Reset	Default	Access
7-4	Reserved	-	-	-
3-0	Timer 1 Down Count Data – High Byte	Xh	Xh	R/W

[REG $05h]: Timer 1 Count_L Register, TM2_L

Bit	Description	Reset	Default	Access
7-0	Timer 1 Down Count Data – Low Byte	Xh	Xh	R/W

Example:

```
LDA   #$03
STA   $04h      ; set Timer1 Counter's High-Byte(Bit-11~8) at 03h
LDA   #$55
STA   $05h      ; set Timer1 Counter's Low-Byte(Bit-7~0) at 55h
                ; set Timer1 Counter at 355h
```

[REG $06h]: Timer 2 Count_H Register, TM2_H

Bit	Description	Reset	Default	Access
7-4	Reserved	--	--	--
3-0	Timer 2 Down Count Data – High Byte	Xh	Xh	R/W

[REG $07h]: Timer 2 Count_L Register, TM2_L

Bit	Description	Reset	Default	Access
7-0	Timer 2 Down Count Data – Low Byte	Xh	Xh	R/W

Example:

```
LDA   #$02
STA   $06h      ; set Timer2 Counter's  High-Byte(Bit-11~8) at 02h
LDA   #$AA
STA   $07h      ; set Timer2 Counter's Low-Byte(Bit-7~0) at AAh
                ; set Timer2 Counter at 2AAh
```

[REG $08h]: Timer 3 Count_H Register, TM3_H

Bit	Description	Reset	Default	Access
7-4	Reserved	--	--	--
3-0	Timer 3 Down Count Data – High Byte	Xh	Xh	R/W

[REG $09h]: Timer 3 Count_L Register, TM3_L

Bit	Description	Reset	Default	Access
7-0	Timer 3 Down Count Data – Low Byte	Xh	Xh	R/W

Example:

```
LDA     #$0F
STA     $08h        ; set Timer2 Counter's High-Byte(Bit-11~8) at 0Fh
LDA     #$00
STA     $09h        ; set Timer2 Counter's Low-Byte(Bit-7~0) at 00h
                    ; set Timer2 Counter at F00h
```

[REG $0Ah]: Timer 1 Control Register, TM1_CTL

Bit	Description	Reset	Default	Access
7	Timer 1 Enable or Timer 1 Start 0: Disable/Stop, 1: Enable/Start	0h	0h	R/W
6	Timer 1 Interrupt Status Bit Write Enable 0: The timer1 interrupt indicate bit(REG_03h bit7) is inhibited to write. 1: The timer1 interrupt indicate bit(REG_03h bit7) is allowed to program.	0h	0h	R/W
5	Timer 2 Interrupt Status Bit Write Enable 0: The timer2 interrupt indicate bit(REG_03h bit6) is inhibited to write. 1: The timer2 interrupt indicate bit(REG_03h bit6) is allowed to program.	0h	0h	R/W
4	Timer 3 Interrupt Status Bit Write Enable 0: The timer3 interrupt indicate bit(REG_03h bit3) is inhibited to write. 1: The timer3 interrupt indicate bit(REG_03h bit3) is allowed to program.	0h	0h	R/W
3	Timer 1 Loop Control 0: Disable, 1: Enable	0h	0h	R/W
2-0	Timer 1 Input Clock Source Select Bit2 Bit1 Bit0 Clock Source --- 0 0 0 Fosc/2 0 0 1 Fosc/4 0 1 0 Fosc/16 0 1 1 Fosc/64 1 0 0 Fosc/256 1 0 1 Fosc/512 1 1 0 Fosc/4096 1 1 1 External I/O Port (PT1_7)	0h	0h	R/W

Example:

```
LDA     #$C0
STA     $0Ah        ; let Timer 1 be Enable/Start → Timer 1 only Count once
                    ; Timer 1's Clock Source set at Fosc/2
                    ; (if Fosc=4MHz, Timer 1's Clock Source will be 2 MHz)
                    ; permit Timer1 interrupt indicate bit(REG_03h bit7) be
```

written

[REG $0Bh]: Timer 2 Control Register, TM2_CTL

Bit	Description	Reset	Default	Access
7	Timer 2 Enable or Timer 2 Start 0: Disable/Stop, 1: Enable/Start	0h	0h	R/W

6	2KHz Time Base Interrupt Status Bit Write Enable 0: The 2KHz Time Base interrupt indicate bit(REG_03h bit2) is inhibited to write. 1: The 2KHz Time Base interrupt indicate bit(REG_03h bit2) is allowed to program.	0h	0h	R/W
5	500Hz Time Base Interrupt Status Bit Write Enable 0: The 500Hz Time Base interrupt indicate bit(REG_03h bit1) is inhibited to write. 1: The 500Hz Time Base interrupt indicate bit(REG_03h bit1) is allowed to program.	0h	0h	R/W
4	62Hz Time Base Interrupt Status Bit Write Enable 0: The 62Hz Time Base interrupt indicate bit(REG_03h bit0) is inhibited to write. 1: The 62Hz Time Base interrupt indicate bit(REG_03h bit0) is allowed to program.	0h	0h	R/W
3	Timer 2 Loop Control 0: Disable, 1: Enable	0h	0h	R/W
2-0	Timer 2 Input Clock Source Select Bit2 Bit1 Bit0 Clock Source --- 0 0 0 Fosc/2 0 0 1 Fosc/4 0 1 0 Fosc/16 0 1 1 Timer 1 Output 1 0 0 Fosc/256 1 0 1 Fosc/512 1 1 0 Fosc/4096 1 1 1 External I/O Port (PT2_7)	0h	0h	R/W

Example:

```
        LDA   #$29
        STA   $0Bh      ; let Timer 2 Loop Control be Enable → Timer2 will keep Counting
                        ; let Timer 2's Clock Source be Fosc/4 (if Fosc=4MHz, Timer2's
                        ; Clock Source will be 1 MHz)
                        ; permit 500Hz Time Base interrupt indicate bit(REG_03h bit1) be
                        ; written

        LDA   #$89
        STA   $0Bh      ; Timer 2 starts to Count
```

[REG $0Ch]: Timer 3 Control Register, TM3_CTL

Bit	Description	Reset	Default	Access
7	Timer 3 Enable or Timer 3 Start 0: Disable/Stop, 1: Enable/Start	0h	0h	R/W
6-4	Reserved	--	--	--
3	Timer 3 Loop Control 0: Disable, 1: Enable	0h	0h	R/W

2-0	Timer 3 Input Clock Source Select Bit2 Bit1 Bit0 Clock Source --- 0 0 0 Fosc/2 0 0 1 Fosc/4 0 1 0 Fosc/16 0 1 1 Fosc/64 1 0 0 Fosc/256 1 0 1 Fosc/512 1 1 0 Fosc/4096 1 1 1 Fosc/16384	0h	0h	R/W

Example:

```
LDA   #$09
STA   $0Ch        ; let Timer 3 Loop Control be Enable → Timer2 will keep Counting
                  ; let Timer 3's Clock Source be Fosc/4 (if Fosc=4MHz,
                  ; Timer3's Clock Source will be 1MHz)
LDA   #$89
STA   $0Ch        ; Timer 3 starts Counting.
```

[REG $0Dh]: Port 1 Data Register, PT1_DAT

Bit	Description	Reset	Default	Access
7	Output Data to Port 1 or Input from Port 1 This bit is also as the TM1 clock source when REG $0Ah bit2-0 == '111'	0h	0h	R/W
6	Output Data to Port 1 or Input from Port 1 This bit is also as the NMI or INT trigger source when REG $00h bit5 == '1', or REG $02h bit5 == '1'.	0h	0h	R/W
5	Output Data to Port 1 or Input from Port 1 This bit is also as the IR output when REG $1Bh bit1 == '1'.	0h	0h	R/W
4	Output Data to Port 1 or Input from Port 1 This bit is also as the PWM output when REG $35h bit7 == '1'.	0h	0h	R/W
3-0	Output Data to Port 1 or Input from Port 1	0h	0h	R/W

Note: If the Port1 is output mode then the read access date is from the register 0Dh. If the Port1 is input mode the read access is form Port1 I/O.

Example:

```
LDA   #$FF
STA   $0Eh        ; set Port1 Bit7~0 at output mode
LDA   #$AA
STA   $0Dh        ; output AAh to Port1 Bit7~0
```

[REG $0Eh]: Port 1 Direction Control Register, PT1_DIR

Bit	Description	Reset	Default	Access
7-0	Select the Output or Input Mode of Port 1 0: Input Mode, 1: Output Mode	0h	0h	R/W

Example:

```
LDA   #$F0
STA   $0Eh        ; set Port1 Bit7~4 at output mode, Bit3~0 at input mode.
```

[REG $0Fh]: Port 1 Output Mode Select Register, PT1_MOD

Bit	Description	Reset	Default	Access
7-0	Select the Output Mode for CMOS or Open-Drain Mode 0: CMOS Mode, 1: Open-Drain Mode When the Open-Drain Mode was be selected then the output data is controlled by the direction control register.	0h	0h	R/W

Example:

```
LDA   #$FF
STA   $0Eh      ; set Port1at output mode
LDA   #$F0
STA   $0Fh      ; set Port1 Bit7~4 at Open-Drain output mode,
                ; Bit3~0 is CMOS Mode output mode
```

[REG $10h]: Port 2 Data Register, PT2_DAT

Bit	Description	Reset	Default	Access
7	Output Data to Port 2 or Input from Port 2 This bit is also as the TM2 clock source when REG $0Bh bit2-0 = '111'.	0h	0h	RW
6	Output Data to Port 2 or Input from Port 2 This bit is also as the NMI or INT trigger source when REG $00h bit4 = '1', or REG $02h bit4 = '1'.	0h	0h	R/W
5-0	Output Data to Port 2 or Input from Port 2	0h	0h	R/W

Note: If the Port2 is output mode then the read access date is from the register 0Dh. If the Port2 is input mode then the read access is form Port2 I/O.

Example:

```
LDA   #$FF
STA   $11h      ; set Port2 Bit7~0 at output mode
LDA   #$55
STA   $10h      ; output 55h to Port2 Bit7~0
```

[REG $11h]: Port 2 Direction Control Register, PT2_DIR

Bit	Description	Reset	Default	Access
7-0	Select the Output or Input Mode of Port 2 0: Input Mode, 1: Output Mode	0h	0h	R/W

Example:

```
LDA   #$F0
STA   $11h      ; set Port2 Bit7~4 at output mode, Bit3~0 at input mode
```

[REG $12h]: DAC 1 Data Register, DAC1_DAT

Bit	Description	Reset	Default	Access
7-0	8-Bit DAC 1 Output Data	0h	0h	R/W

Example:

```
LDA   #$18
STA   $14h      ; set DAC1 Enable
LDA   #$80
STA   $12h      ; output 80h to DAC1 → Output middle current
```

[REG $13h]: DAC 2 Data Register, DAC2_DAT

Bit	Description	Reset	Default	Access

7-0	8-Bit DAC 2 Output Data	0h	0h	R/W

Example:

```
LDA    #$28
STA    $14h        ; set DAC2 Enable
LDA    #$FF
STA    $13h        ; output FFh to DAC2 → Output maximum current
```

[REG $14h]: DAC Control Register, DAC_CTL

Bit	Description	Reset	Default	Access
7-6	Reserved	--	--	R/W
5	DAC 2 Output Enable 0: Disable, 1: Enable	0h	0h	R/W
4	DAC 1 Output Enable 0: Disable, 1: Enable	0h	0h	R/W
3	DAC or PWM Select 0: PWM Mode, 1: DAC Mode	0h	0h	R/W
2	Reserved	--	--	--
1	DAC2 Auto Mode 1: Enable, 0: Disable	0h	0h	R/W
0	DAC1 Auto Mode 1: Enable, 0: Disable	0h	0h	R/W

Example:

```
LDA    #$00
STA    $14h        ; set DAC1, DAC2 Disable
```

[REG $15h]: PWM Data Register, PWM

Bit	Description	Reset	Default	Access
7	PWM Enable 0: Disable 1: Enable	Xh	Xh	R/W
6-0	PWM Duty Cycle Control	Xh	Xh	R/W

Example:

```
LDA    #$30
STA    $14h        ; set DAC at PWM Mode
LDA    #$FF
STA    $15h        ; set PWM Enable, produce 50% Duty Cycle of PWM Pulse
```

[REG $16h]: Flash ROM Bank Select Register, FBANK

Bit	Description	Reset	Default	Access
7-0	Embedded Flash ROM Bank Select for Normal or ISP Mode Bit7 6 5 4 3 2 1 0 Bank Select ---------- 0 0 0 0 0 0 0 0 Bank 0 0 0 0 0 0 0 0 1 Bank 1 0 0 0 0 0 0 1 0 Bank 2 0 0 0 0 0 0 1 1 Bank 3	0h	0h	R/W

Example:

```
LDA    #$01
STA    $16h        ; select Flash's Bank1 (each Bank is 16Kbyte)
```

[REG $17h]: Power Saving and Reset Control Register, PWR_CTL

Bit	Description	Reset	Default	Access
7	Port 1 Tri-state 0: Normal Mode, 1: Power Saving Mode	0h	0h	R/W
6	Port 2 and TXD Tri-state 0: Normal Mode, 1: Power Saving Mode	0h	0h	R/W
5	Flash Power Down 0: Normal Mode, 1: Power Saving Mode	0h	0h	R/W
4	Stop the clock. 0: Normal Mode, 1: Power Saving Mode This power saving mode can be wakeup by 3 ways: ◆ Manual Power-Off then Power-On ◆ Reset Pin Reset ◆ Port 2 bit3 Falling Trigger Reset. If the Register17 bit4~7 is set to '1111', then the RA8906 will enter the standby mode.	0h	0h	R/W
3	Flash Operation Mode 0: Low Speed, 1: Hi Speed	0h	0h	R/W
2-0	Reserved	--	--	--

Example:

```
LDA     #$F0
STA     $17h        ; enter Power Saving Mode
```

[REG $18h]: Reset Status Register, RST_ST

Bit	Description	Reset	Default	Access
7	Power On Reset Indicate	1h	0h	R/W
6	RESET (RESET#) Pin Reset Indicate	0h	0h	R/W
5	Watch-Dog Reset Indicate	0h	0h	R/W
4	Software Reset Indicate	0h	0h	R/W
3-0	Reserved	--	--	--

Example:

```
LDA     $18h        ; can be used to diagnose the source of the Reset
```

[REG $19h]: Clock Control Register, CLK_CTL

Bit	Description	Reset	Default	Access
7	CPU Clock Select 0: Normal CPU Clock 1: Low Speed CPU Clock Controlled by Bit6-5 of This Register	0h	0h	R/W
6-5	CPU Clock Divisor Bit6 Bit5 Low Speed CPU Clock --- 0 0 Fosc/2 0 1 Fosc/64 1 0 Fosc/256 1 1 Fosc/65536	0h	0h	R/W
4	Watch Dog Timer Clock Source Select 0: Fosc / 2^{17} 1: Fosc / 2^{20}	0h	0h	R/W

3-2	Internal Time Base and IR Modulation Clock Source Bit3 Bit2 Clock Source --- 0 0 2KHZ, 500HZ, 62HZ, IR (Fosc = 8MHz) 0 1 2KHZ, 500HZ, 62HZ, IR (Fosc = 4MHz) 1 0 2KHZ, 500HZ, 62HZ, IR (Fosc = 2MHz) 1 1 2KHZ, 500HZ, 62HZ, IR (Fosc = 1MHz) **NOTE: If Fosc =6MHz** Bit1 Bit0 IR Clock Source Internal Time Base Clock Source --- 0 0 23.58KHz 732.6HZ, 183.2HZ, 22.89HZ 0 1 46.73KHz 1466HZ, 366.3HZ, 45.79HZ 1 0 92.46KHz 2933HZ, 732.6HZ, 91.58HZ 1 1 188KHz 5875HZ, 1466HZ, 183.1HZ	0h	0h	R/W
1-0	Internal 32KHz(INT32K) Clock Source Bit1 Bit0 Clock Source --- 0 0 Fosc/256 (Fosc = 8MHz) 0 1 Fosc/128 (Fosc = 4MHz) 1 0 Fosc/64 (Fosc = 2MHz) 1 1 Fosc/32 (Fosc = 1MHz)	0h	0h	R/W

Example:

```
LDA   #$E0
STA   $19h      ; set CPU Clock at Fosc/65536
                ; Watch Dog Timer Clock is Fosc/2^17
```

[REG $1Ah]: Flash Control Register, FL_CTL

Bit	Description	Reset	Default	Access

7-4	Flash Operation Mode Mode Bit7 Bit6 Bit5 Bit4 Flash Operation Mode ---------- **0** 0 0 0 0 Normal Read **1** 0 0 0 1 Flash Whole Chip Erase **2** 0 0 1 0 Flash Block Erase REG[16] bit1 bit0 CPU-A13 Erased Block ---------- 0 0 0 0-8KB (bank 0) 0 0 0 8-16KB (bank 0) 0 1 0 16-32KB (bank 1) 1 0 0 32-48KB (bank 2) 1 1 0 48-64KB (bank 3) **3** 0 0 1 1 Erase Verify **4** 0 1 0 0 Program_8 **5** 0 1 0 1 Program_16 **6** 0 1 1 0 Program verify **7** 0 1 1 1 Program Lock, Sbyte, BPSW, BPC, Config **8** 1 0 0 0 Special Read (Verify Programmed Lock, Sbyte, BPSW, BPC, Config, ID **9** 1 0 0 1 Erase Sbyte, BPSW, BPC **10** 1 0 1 0 Verify Erased Lock, Sbyte, BPSW, BPC, Config **11** 1 0 1 1 Flash Test Mode **12** 1 1 0 0 Chip Erase End **13** 1 1 0 1 Block Erase End **14** 1 1 1 0 Initialize **15** 1 1 1 1 Reserved	0h	0h	R/W
3	Flash Program Mode 0: Normal mode, 1: Program mode	0h	0h	R/W
2-0	PA[2:0] for Flash Operation Mode 7, 8, 9 and 10	0h	0h	R/W

[REG $1Bh]: Misc. Control Register, MISC_CTL

Bit	Description	Reset	Default	Access
7	Watch Dog Enable 0: Disable, 1: Enable	0h	0h	R/W
6	Watch Dog Loop Control 0: Disable, 1: Enable	0h	0h	R/W
5-4	Reserved	--	--	--
3	Watch Dog Reset Enable 1: Enable, 0: Disable	0h	0h	R/W
2	Software Reset 1: Reset CPU, Write this bit high will cause CPU Reset. This bit will be clear automatically after reset. 0: Disable.	0h	0h	R/W
1	IR Enable, 0: Disable, 1: Enable The modulation frequency is selected by REG19 bit3-2.	0h	0h	R/W

0	IR Data	0h	0h	R/W

Example 1:

```
LDA   #$80
STA   $1Bh      ; set Watch Dog Enable
```

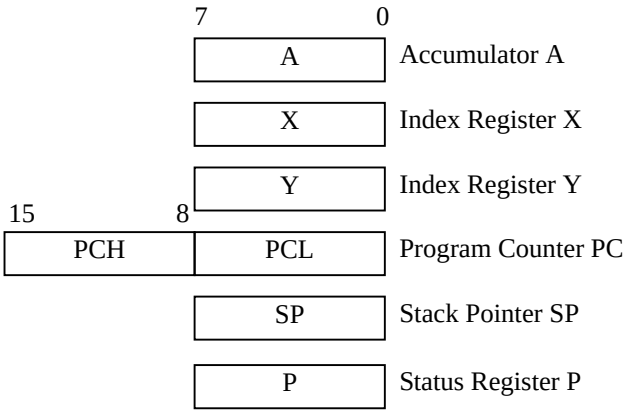
Example 2:

```
LDA   #$04
STA   $1Bh      ; produce Software Reset → reset CPU
```

7. Function Description

7.1 e-MCU CPU

RA8906 contains a 8-bit e-MCU which has 14 different types of addressing modes. It can provide 11 types of instruction function and total 96 instructions. The framework of the register contains one Accumulator, one Status register, two Index registers, one Stack Pointer, and one Program Counter. The explanation of these register is as following:



Accumulator (A): This is the only register that can directly do arithmetic and logical operation, and is used to save one of the operand of addition, subtraction, AND, OR and EOR and the operation result.

Index Register (X): X Register is mainly used as an Index Register to save serial data in the memory. As the program can instruct the content to add one or subtract one, however, X Register is often used as a regular counter.

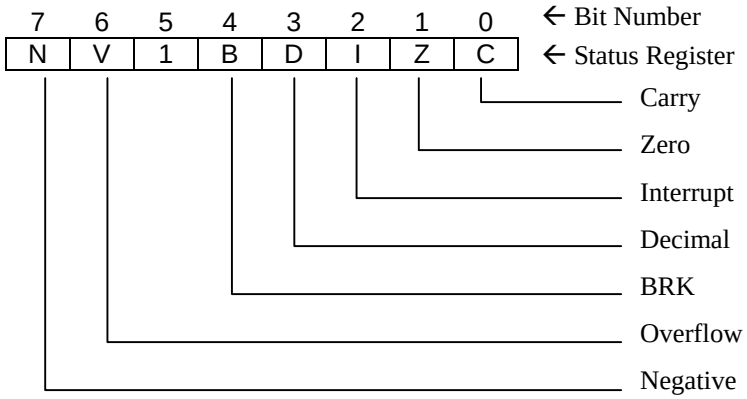
Index Register (Y): Y Register can also be used as Index Register, similar to X Register.

Program Counter (PC): Program Counter indicates the current address executed.

Stack Pointer (SP): Program design makes MCU transfer to the instruction set of another part of memory address to go on after executing to some point. Before the real transfer takes place, e-MCU saves the current instruction set's next instruction address to be executed -- Return Address. After finishing the new instruction set, the control will return to the latest address left. Return Address is saved in a pile of special memory area called Stack, which is a part of memory addresses assigned to receive the Return Address. In e-MCU, Stack is designed in the first page (address 0100 to 01FF) of the memory. As this design limit assures the second highest bit set of Stack Address to be always 01(hexadecimal), Stack Addressing Register, Stack Pointer in other words, is only 8 bits long.

Of all the mentioned Registers in e-MCU MCU, Accumulator, X Register and Y Register can be used freely by program designers to save temporary data, communicate with the memory and be used as counters or general registers of multi-functions.

Status Register P: Status Register in e-MCU MCU consists of seven usable bits. Among them, five are Status Flags, which provide relevant information on the executed results of the former instructions (mostly the preceding instruction). The other two are control bits.



Carry Flag (C): Carry Flag is used to save any carry from addition operation, borrow from subtraction operation, or bits removed from operands when doing shift-operation. Carry Flag can also reflect the result of compare-operation.

Zero Flag (Z): Zero Flag is used to show if the operation comes to zero or not. If it is the case, the Flag value is 1. Otherwise it is 0.

BRK Flag (B): BRK Flag shows that the interrupt request received from e-MCU MCU results either from 'interrupt" instruction or the interrupt of external equipment.

Overflow Flag (V): Overflow Flag is used only in arithmetic operation of signed-value. When we add two "same-signed-value" or subtract two "different-signed-value" and produce the arithmetic operation of bigger than used in signed-value, or when we add two "same-signed-value" or subtract two "different-signed-value" and produce a result of bigger than +127 or less than -128, then the Overflow Flag is 1.

Negative Flag (N): Negative Flag is used to show if the arithmetic operation of sign value produces a result of negative figures or not. If it is the case, the Flag value is 1. Otherwise, it is 0. In fact, it can also be used to show the content of the highest bit in Accumulator.

Interrupt Flag (I): IRQ Disable bit (I) is used to reject interrupt requests from the external equipment when MUC is not ready to do interrupt service for some reasons.

Decimal Flag (D): Decimal mode bit (D) is used to control the arithmetic/logic unit of e-MCU and make it act like binary adder or decimal adder. In binary form, ALU regards the operand as a decimalist with two BCD (decimal) figures. Namely, the result of arithmetic operation is the same as that of normal decimal operation. If the value of Decimal mode bit (D) is 1, then the ALU of e-MCU will act in decimal mode.

e-MCU's 14 addressing modes

No	Mode	Format	Example
1	**Immediate Addressing Mode**	#aa	LDA #$55
2	**Absolute Addressing Mode**	aaaa	LDA $2000
3	**Zero Page Addressing Mode**	aa	LDA $30
4	**Implied Addressing Mode**	--	DEX
5	**Accumulator Addressing Mode**	Acc	LSR A

6	**Absolute Indexed Addressing Mode**	aaaa, X[Y]	LDA $2000, Y
7	**Zero-Page Indexed Addressing Mode**	aa, X[Y]	LDA $20, X
8	**Absolute Indirect Addressing Mode**	(aaaa)	JMP ($2000)
9	**Zero-Page Indirect Addressing Mode**	(aa)	LDA ($20)
10	**Absolute Indexed Indirect Addressing Mode**	(aaaa, X)	LDA ($2000, X)
11	**Zero-Page Indexed Indirect Addressing Mode**	(aa, X)	LDA ($20, X)
12	**Zero-Page Indirect Indexed Addressing Mode**	(aa), Y	LDA ($20), Y
13	**Relative Addressing Mode**	aaaa	BEQ $20
14	**Stack Addressing Mode**	--	PHA

aa = 2 hex digits as $FF
aaaa = 4 hex digits as $FFFF

e-MCU's 11 instruction function and usable instructions

NO.	Instructions Group	Instructions
1	**Load and Store**	LDA, LDX, LDY STA, STX, STY
2	**Arithmetic**	ADC, SBC,
3	**Increment and Decrement**	INC, INX, INY DEC, DEX, DEY
4	**Logical**	AND, OR, EOR
5	**Jump, Branch, Compare, and Test Bits**	JMP BRA, BCS, BEQ, BNE, BMI, BPL, BVC, BVS CMP, CPX, CPY BIT, TRB, TSB, BBRn*, BBSn*
6	**Shift and Rotate**	ASL, LSR ROL, ROR
7	**Transfer**	TAX, TAY, TXA, TYA
8	**Stack**	TSX, TXS, PHA, PHX, PHY, PHP, PLA, PLX, PLY, PLP
9	**Subroutine**	JSR, RTS, RTI
10	**Set and Reset (Clear)**	CLC, CLD, CLI, CLV, RMBn* SEC, SED, SEI, SMBn*
11	**Other Instructions**	NOP, BRK

*n = 0~7

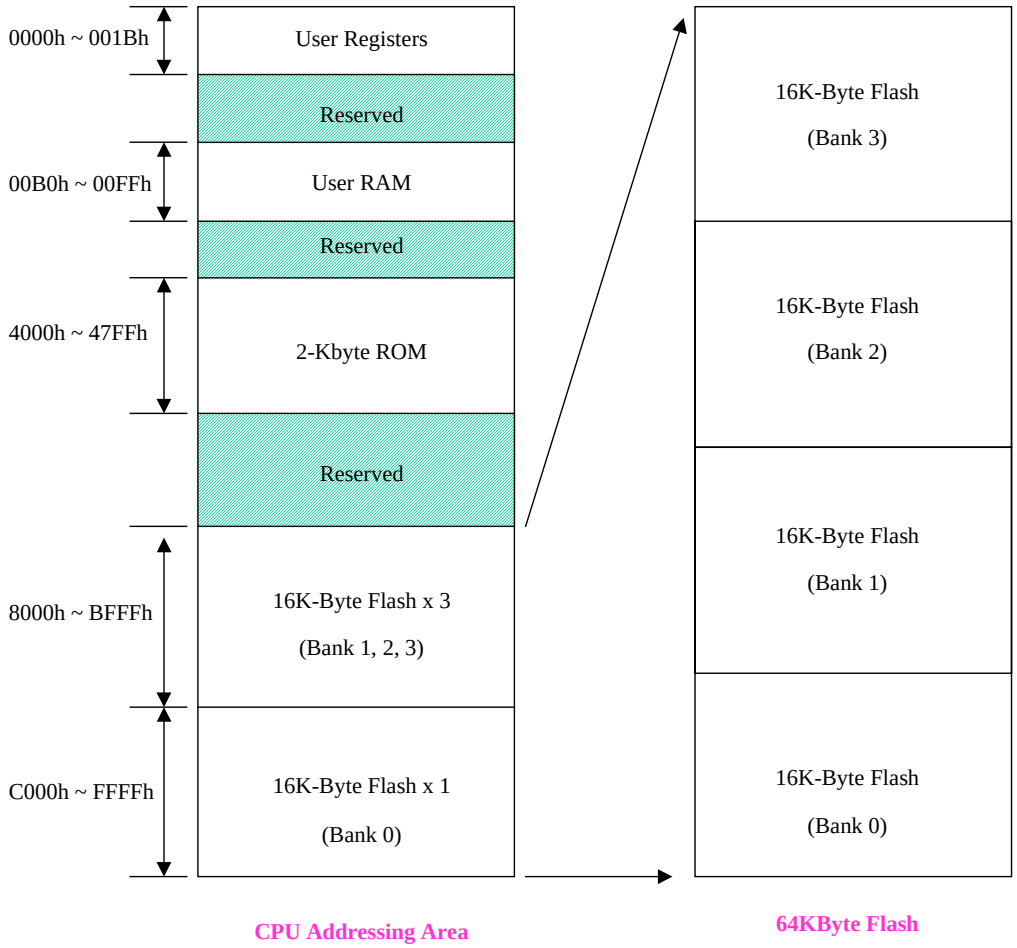
If you want to understand more about the Addressing Mode, Instruction Set, and design skills of e-MCU, please refer to the user manuals of e-MCU (Instruction Set and Addressing Mode) and RICE-2000.

7.2 Memory & Registers

RA8906 contains 28Bytes' User Register, 80Bytes' User SRAM, and 64Bytes' Flash ROM, and the Addressing range is as following drawing. We have introduced 28Bytes' User Register at Chapter 6, as regards 80Bytes' User SRAM is addressing at B0~FF. Because both User Register and SRAM are inside Zero Page 00~FF, users can take advantage of Zero Page Addressing Mode that provided by e-MCU to access User Register and User SRAM.

The maximum addressing range of e-MCU is 64Kbytes, so 64Kbyte's Flash ROM should be separated into different Banks while accessing. Bank0 decodes to C000~FFFF, Bank1~3 decode to 8000~BFFF. Users can program REG$16 to set different Banks.
For Example:

```
LDA    #$03
STA    $16h        ; select Flash's Bank3 (each Bank is 16Kbyte)
```



Unused Area, the ROM Address of 4000~47FF is for System used, and regular User can use it.

7.3 CPU Clock & Interrupt

Through REG $19h's bit7-5 can control the CPU Clock of e-MCU. If set bit4 of REG $17h to high, then CPU clock will stop. Users will do this when they want to enter Power saving mode.

Example 1:

```
LDA   #$85      ; If Use 4MHz Xtal
STA   $19h      ; set CPU Clock at 2MHz (Fosc/2 → 4MHz/2)
                ; Watch Dog Timer Clock at 15Hz(Fosc/217 ( 4MHz/217)
```

Example 2:

```
LDA   #$F0
STA   $17h      ; enter Power Saving Mode:
                ; PT1, PT2 ( Tri-state
                ; Flash ( Power down
                ; CPU ( Off
```

CPU Clock

e-MCU provides two Interrupt Modes: NMI(Non-Maskable Interrupt) and Maskable Interrupt(INT), REG$00h and $02h are used to control those Interrupt sources whether can produce Interrupts or not. REG$010h and $03h are used to diagnose the source of Interrupt.

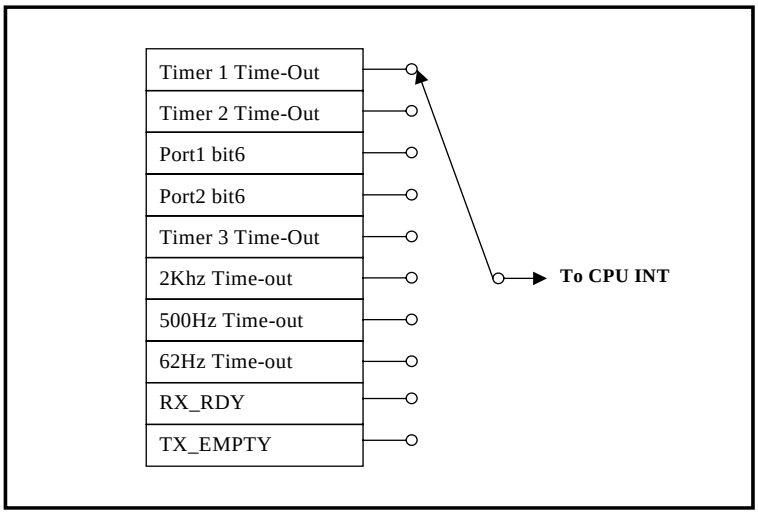
Example 1:

```
LDA   #$C0
STA   $00h      ; permit Time1 and Time2 produce NMI Interrupt
LDA   #$C0
STA   $02h      ; permit 2Khz Time base produce INT Interrupt
```

Example 2:

```
LDA   $03h      ; can be used to diagnose the source of interrupts.
  :
  :
LDA   #$00h
STA   $01h      ; eliminate the instruction of NMI interrupt
STA   $03h      ; eliminate the instruction of NMI interrupt
```

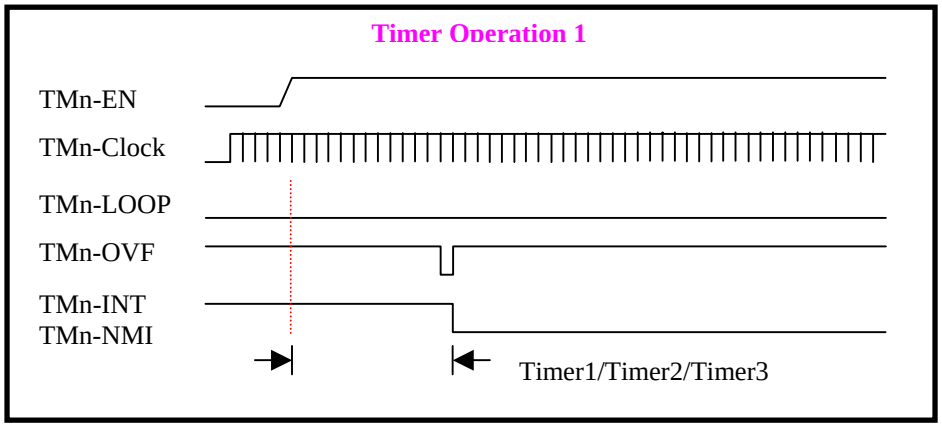
CPU NMI Trigger Source

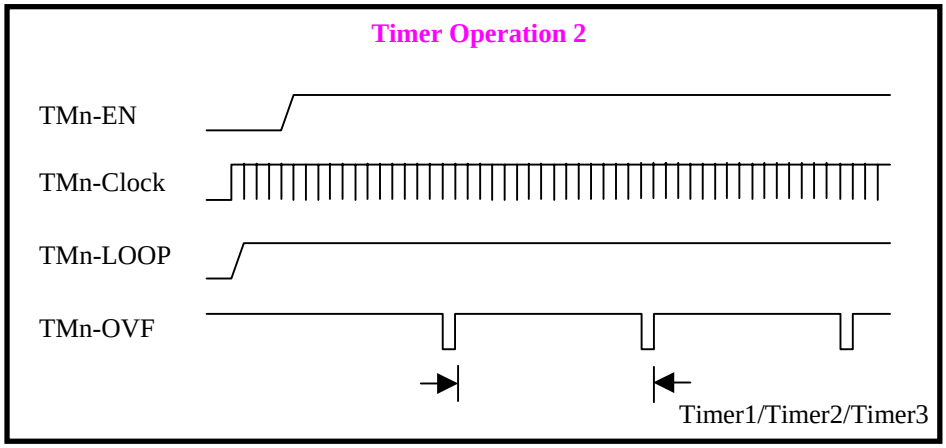


CPU INT Trigger Source

7.4 Timer & Time Base

The RA8906 supports three independent 12-bits down-counter timers. When the time-out occurred then the internal timer overflow signal would generate and drive the interrupt or non-mask interrupt to low. Through setting Register, User can control Timer to count once (like Operating 1 of the following drawing) or to repeat counting (like Operation 2)





The Clock sources of RA8906's three 12Bit Timers are all independent. The Timer clock source is controlled by the bit2-0 of register 0Ah, 0Bh & 0Ch.

Example:

```
LDA   #$29
STA   $0Bh        ; set Timer 2 Loop Control at Enable ( Timer2 will keep
                  ; Counting
                  ; set Timer 2's Clock Source at Fosc/4 (if Fosc=4MHz,
                  ; Timer2 's Clock Source is 1MHz) permit 500Hz Time Base
                  ; interrupt indicate bit(REG_03h bit1) be written
LDA   #$89
STA   $0Bh        ; Timer 2 starts Counting
```

When Timer occurs Overflow, CPU usually is informed by Interrupts. Timer1~3 can let CPU produce NMI or INT 's Interrupt.

Timer1/Timer2/Timer3 Clock Source

Except 3 timers, RA8906 also support 3 base timers for user to generate interrupt: 2Khz, 500Hz and 62Hz. The usage of them is much easier than the Timer, and only produces INT. Because users might use different Xtal frequency (2MHz , 4MHz or 6MHz), users can control Time Base's Clock Source by REG$19's bit3-2.

Example 1:

```
LDA   #$85        ; If Use 4MHz Xtal
STA   $19h        ; set CPU Clock at 2MHz (Fosc/2 ( 4MHz/2)
                  ; Watch Dog Timer Clock at 15Hz(Fosc/217 ( 4MHz/217)
```

; Time Base = 2Khz, 500Hz, 62Hz

Timer Base Clock Source

7.5 Watch Dog Timer

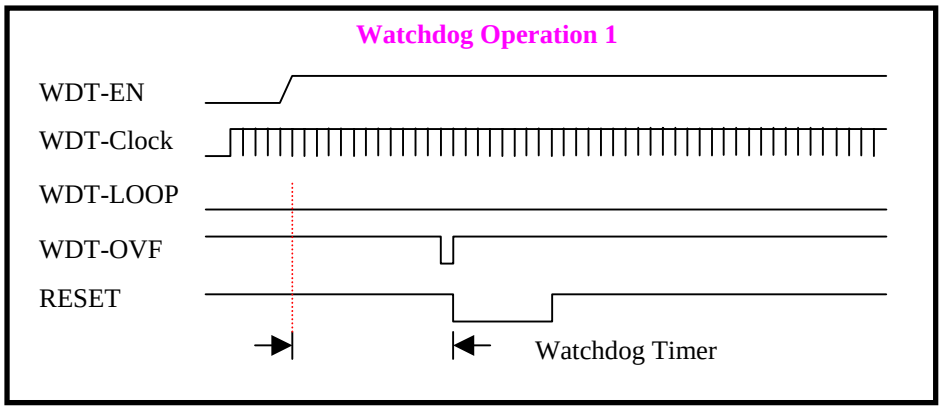
The watchdog timer is a 4-bit down counter and its operation is same as the timers except the watchdog may generate CPU-Reset. Watch-Dog's Operation mode has two kinds, the same as Timer. When the WTD-LOOP is disabled then the watchdog timer overflow will generated one time. If the WTD-LOOP is enable then the watchdog timer overflow will generated continuous.

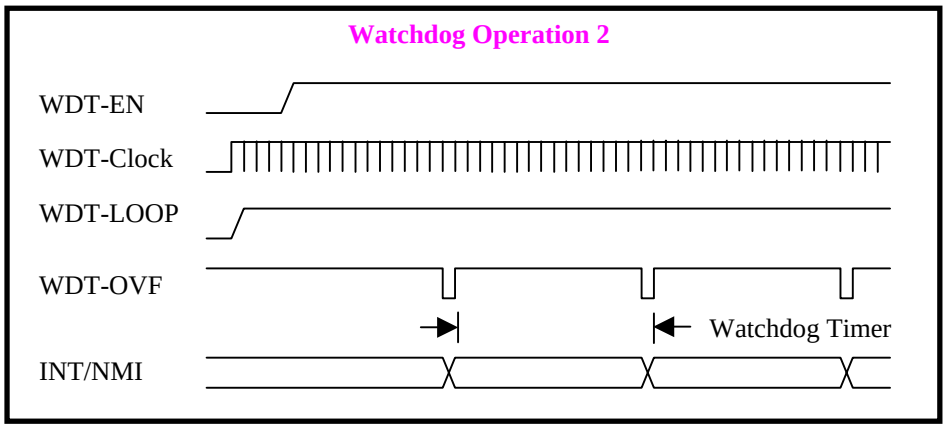
Example 1:

```
LDA    #$80
STA    $1Bh        ; set Watch Dog Enable
```

Example 2:

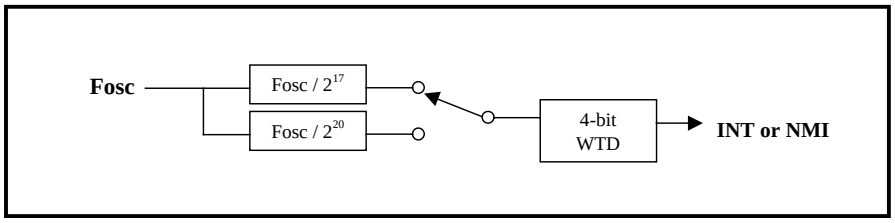
```
LDA    #$04
STA    $1Bh        ; produce Software Reset → set CPU
```





When Timer occurs Overflow, CPU usually is informed by Interrupts. Mentioned before: Watch-Dog can let CPU produce NMI Interrupt.

The Watchdog clock source is controlled by the bit4 of REG $19h.



Watch Timer Clock Source

7.6 I/O Port

RA8906 provides tow independent I/O ports. Each I/O port has 8 bits that can all be set as Input or Output. Among which, Port 1 can be set as Open-Drain.

Example 1:

```
LDA   #$FF
STA   $0Eh      ; set Port1 Bit7~0 at output mode
LDA   #$AA
STA   $0Dh        ; output AAh to Port1 Bit7~0
```

Example 2:

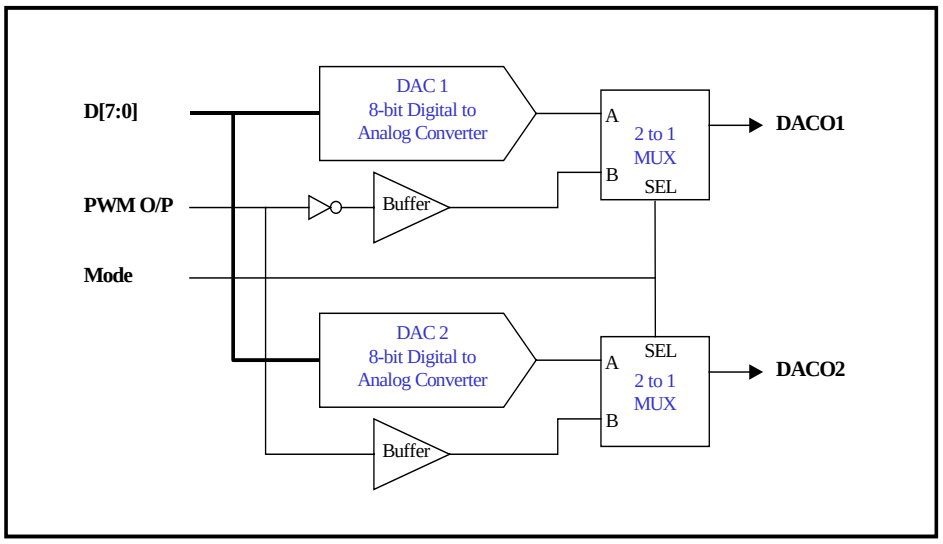
```
LDA   #$FF
STA   $0Eh      ; set Port1at output mode
LDA   #$F0
STA   $0Fh      ; set Port1 Bit7~4 at Open-Drain output mode,
                ; Bit3~0 at CMOS Mode output mode.
```

Example 3:

```
LDA   #$FF
STA   $11h      ; set Port2 Bit7~0 at output mode
LDA   #$55
STA   $10h      ; output 55h to Port2 Bit7~0
```

Some I/O ports can provide other functions. For example, the bit-6 of Port1 and Port2 can provide Falling-edge's trigger, let CPU produce NMI or INT's interrupt. The bit-7 of Port1 and Port2 can be used as Timer1&Timer2's Clock source. After RA8906 enter into Power Saving mode, bit-3 of Port2 can be used as the Trigger source of a wake-up

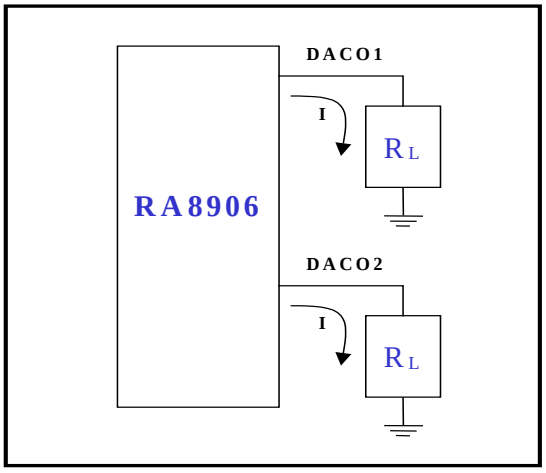
7.7 DAC



DAC and PWM

The RA8906 support two 8-bit current types DAC. In the DAC mode, the maximum output current is 5mA as well as the data is “FF”. If the data is ‘00’ then the output current is zero.

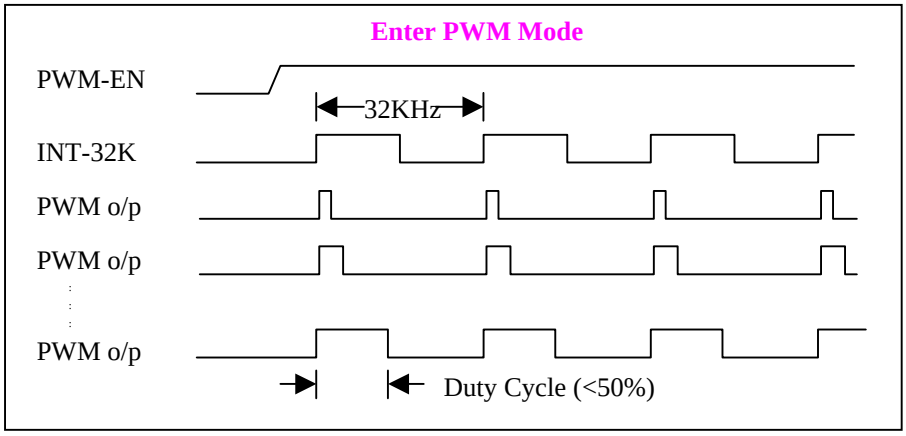
In the PWM mode, the DACO1 and DACO2 are two complementary current output. The maximum current is 50mA. The following is the application diagram for both modes.



DAC Application

Example:

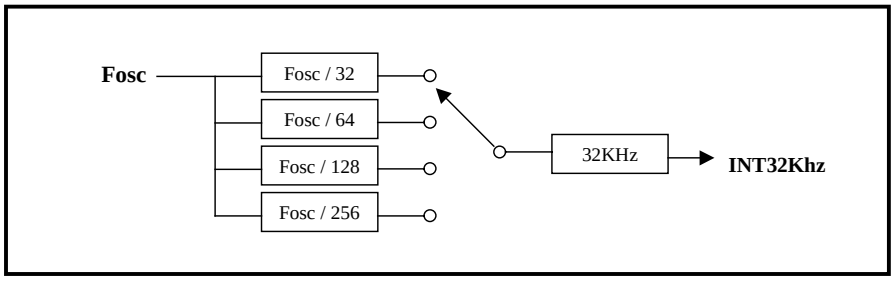
```
LDA    #$28
STA    $14h        ; set DAC2 Enable
LDA    #$FF
STA    $13h        ; output FFh to DAC2 → Output maximum current
```



When the bit3 of REG $14h is set to high, then the PWM mode is enable. The duty cycle of PWM output is controlled by REG $15h bit6~0. The internal 32Khz-clock source is controlled by the bit1-0 of REG $19h.

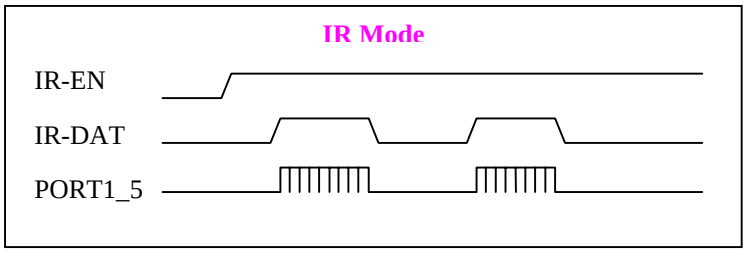
Example:

```
LDA    #$30
STA    $14h        ; set DAC at PWM Mode
LDA    #$FF
STA    $15h        ; set PWM Enable, producing 50% Duty Cycle's PWM Pulse
```



Internal 32Khz Clock Source

7.8 IR

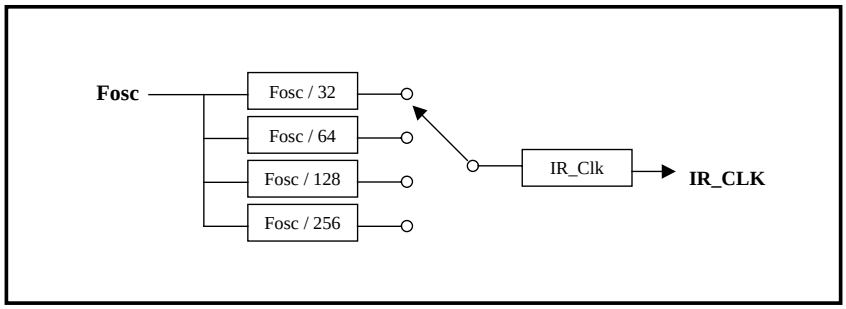


When the bit1 of REG $1Bh(IR-EN) is set to high, then the IR signal is driven through the port1_5. The IR

modulation frequency is controlled by TBS1 and TBS0. (bit2 & bit3 of REG $19h)

Register 19

Bit3	Bit2	IR-Clock
0	0	Fosc/256 (if Fosc-8MHz then IR-CLK = 31.25k)
0	1	Fosc/128
1	0	Fosc/64
1	1	Fosc/32



IR Modulation Clock Source

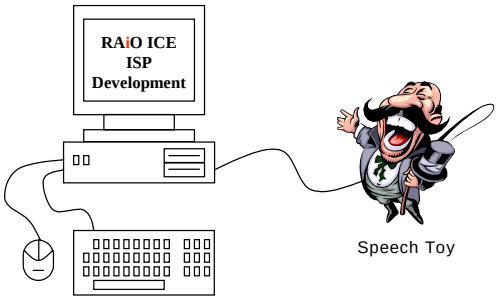
8. Develop Environment

8.1 Develop Mode

The RA8906 support the ISP(In-System Programming) and ISD(In-System Debugging) functions for customer to develop their system. Users can download their programs as well as data from a PC host to the embedded Flash ROM of RA8906.

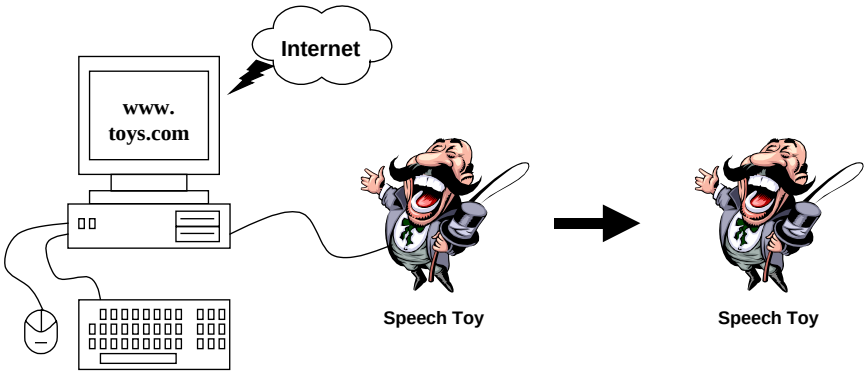
-***ISP/ISD Mode*** is entered when the MONITOR# pin having been pulled down to ground voltage level. The on-chip Monitor program together with RAiO's ICE(RICE-2000) Utility Program running on a PC will be executed to support ICE debugging and ISP download of user programs from the PC Host.

The RAiO ICE(RICE) is a complete develop system which based on Win95/Win98 system. The target of RICE is very easy used for customer to plan, design and debug for their program. And, It helps customers to save a lot of time for development. Because the RICE is used real chip mode so it's 100% software compatible with the mass production.



Develop Program from ISP Mode
(Customers)

-***User Mode*** is entered when the MONITOR# pin has been pulled up to logic high voltage level. User application programs can be executed only in this mode. The end-user can download the application program or data from the customer's website through the PC interface. Because the program/data was stored in the flash so the application device of customer(such as speech toy) will operate independent that after disconnect with the PC.



Down load from Internet
(End-Users)

A Portable Speech Toy

The interface of RA8906 and RICE is using RS232. If users use 4MHz or 2MHz's XTAL, then output Baud rate will be 38.4KBps. If user use 6MHz's XTAL, then the output Baud rate can reach **115.2KBps**.

8.2 RAiO ICE

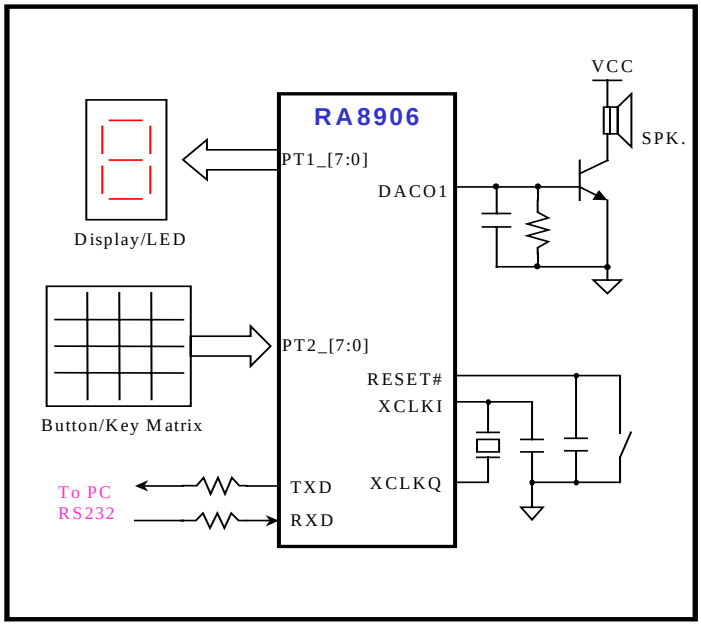
RICE-2000(RICE for short) is a full-completed environment developed by RAiO especially for RA89XX series. The major reason for developing RICE is give fully convenience to program designers who are using RA89XX IC, and let them enjoy consistent and friendly design environment at planning, designing and debugging. In RICE environment, it saves a great deal of developing time by not only providing Editor for users to do direct coding, but also providing many Hot-Key functions for users to do direct compiling, linking, and downloading. Since RA89XX series carry e-MCU micro-processor and a framework of ISP(In-System-Programming), ISD(In-System Debugging), then this simple and reliable environment of RICE can let program designers to proceed design and debug in Real Chip. Moreover, the mass-production ICs are ready for clients to do planning and designing directly without diverse traits happened between developing time and mass production period.

In the meanwhile, in order to support integrated speech interface, RICE provides a solution of 32K-bps ADPCM for programmers to easily combine programs and speech files. If you want to have more information and program design skills of RA8906, please refer to the user manual of RICE-2000.

9. Application

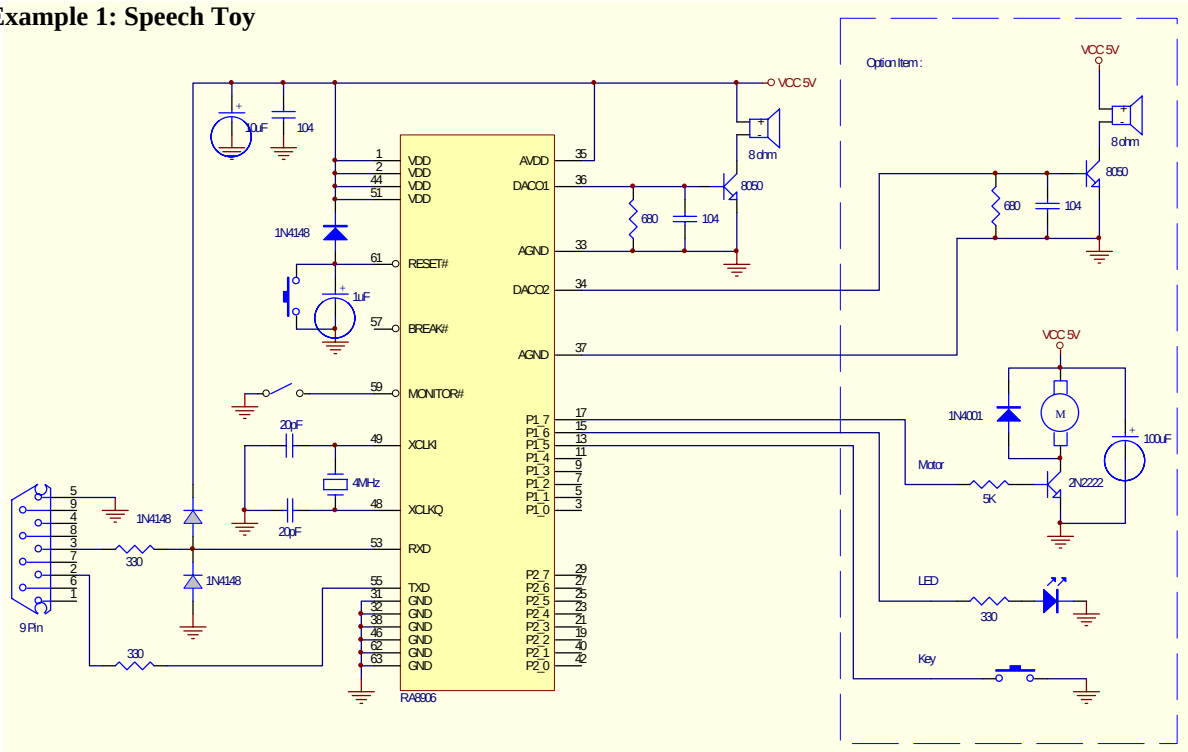
9.1 Application Notes

The following Block diagram is the basic application circuit of RA8906. We also give three examples on the user manual of RICE-2000 to let users have more understanding of RA8906 and the develop environment of RICE-2000, and then start to proceed program designing and product developing. The first example is a simple I/O control, using e-MCU instructions to control the movement of the LEDs. The second and third examples are for speech demonstration. Because RA89XX series and RICE provide a solution of 32K-bps ADPCM, we put two basic examples for programmers for the sake of easily combine programs and speech files. Please refer to the user manual of RICE-2000 if you needed.

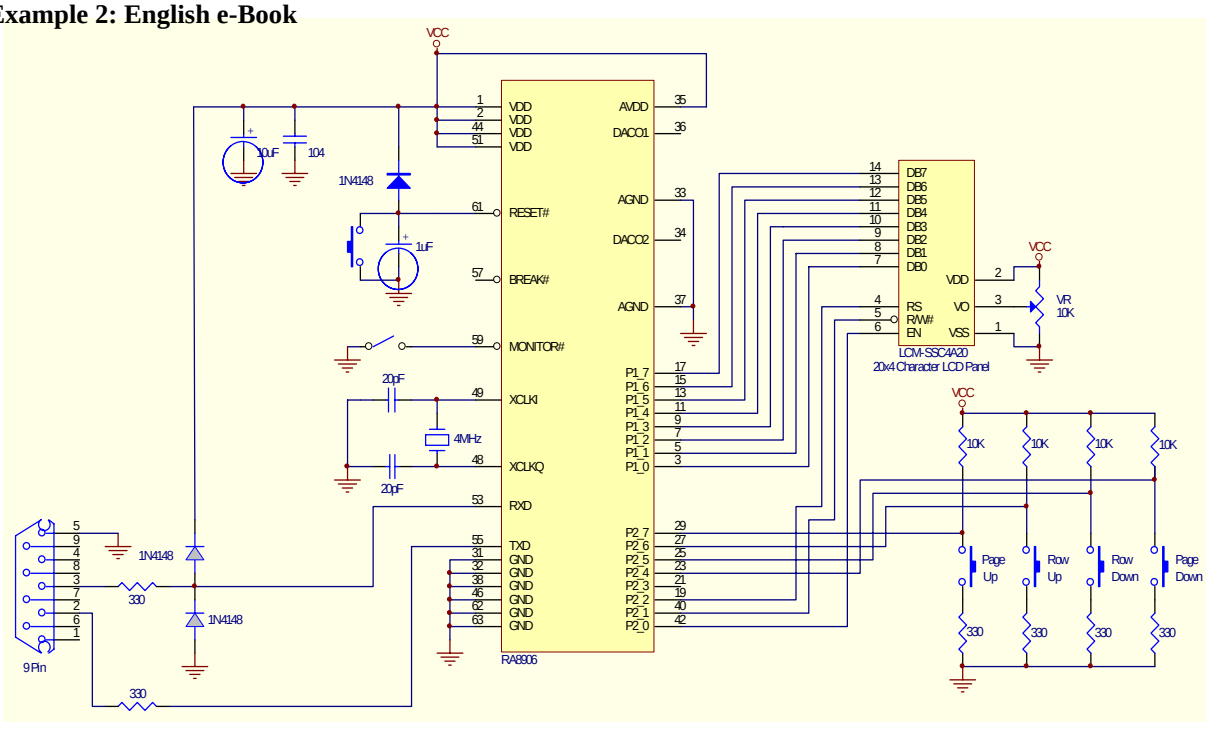


9.2 Application Circuit

Example 1: Speech Toy



Example 2: English e-Book



9. Electrical Characteristic

Operating Temperature Range ..... $0^{O}C$ to + $75^{O}C$
Storage Temperature Range ..... - $55^{O}C$ to +$140^{O}C$
Lead Temperature Range (soldering, 10 seconds)..... +$300^{O}C$
Positive Voltage on any pin, with respect to Ground ..... V_{IO} + 0.3V
Negative Voltage on any pin, with respect to Ground ..... -0.3V
Maximum V_{IO}..... +6V
Maximum V_{CC} ..... +6.5V

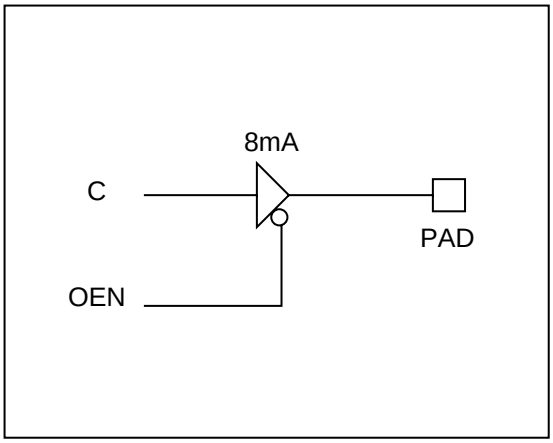
* Stresses above those listed above could cause permanent damage to the device. This is stress rating only and functional operation of the device at any other condition above those indicated in the operation sections of this specification is not implied.

9.1 DC Electrical Characteristics

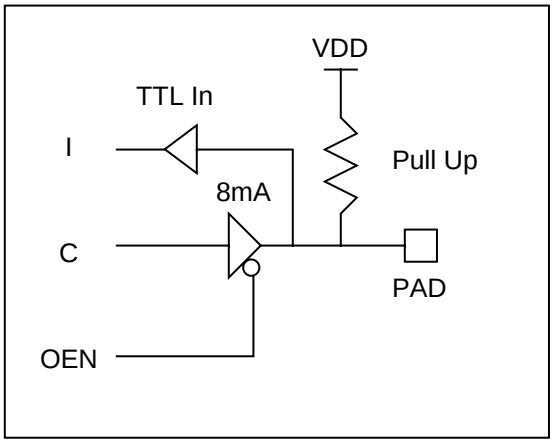
(T_A = $0^{O}C$ - $75^{O}C$, V_{CC} =4.5~5.5V)

Parameter	Symbol	Min	Typ.	Max	Units	Comments
Input Buffer (RXD)						
Low Input Level	V_{ILI}	2.0		0.8	V	TTL Levels
High Input Level	V_{IHI}				V	
Schmitt Input Buffer (RESET#, BREAK#, MONITOR#)						Schmitt Trigger Schmitt Trigger
Low Input Level	V_{ILIS}	2.2		0.8	V	
High Input Level	V_{IHIS}				V	
Schmitt Trigger Hysteresis	V_{HYS}		250		mV	
Output Buffer (TXD)						
Low Output Level	V_{OL}			0.5	V	I_{OL} = 12mA (24mA)
Output Leakage	I_{OL}	-10		+10	uA	V_{IN} = 0 to V_{CC} (Note 1)
I/O Buffer (PT1_[7:0])						
Low Output Level	V_{OL}			0.5	V	I_{OL} = 8mA
High Output Level	V_{OH}	2.4			V	I_{OH} = -4mA
Output Leakage	I_{OL}	-10		+10	uA	V_{IN} = 0 to V_{CC} (Note 1)
I/O Buffer (PT2_[1:0])						
Low Output Level	V_{OL}			0.5	V	I_{OL} = 16mA
High Output Level	V_{OH}	2.4			V	I_{OH} = -8mA
Output Leakage	I_{OL}	-10		+10	uA	V_{IN} = 0 to V_{CC} (Note 1)
I/O Buffer (PT2_[7:2])						
Low Output Level	V_{OL}			0.4	V	I_{OL} = 4mA
High Output Level	V_{OH}	2.4			V	I_{OH} = -2mA
Output Leakage	I_{OL}	-10		+10	uA	V_{IN} = 0 to V_{IO} (Note 1)

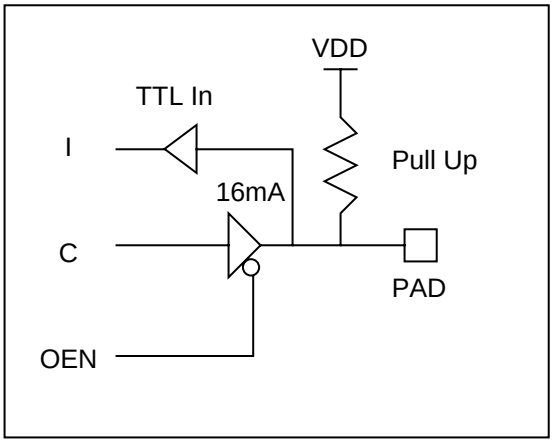
Capacitance T_A = $25^{O}C$; Fc = 4MHz; V_{CC} = 5V



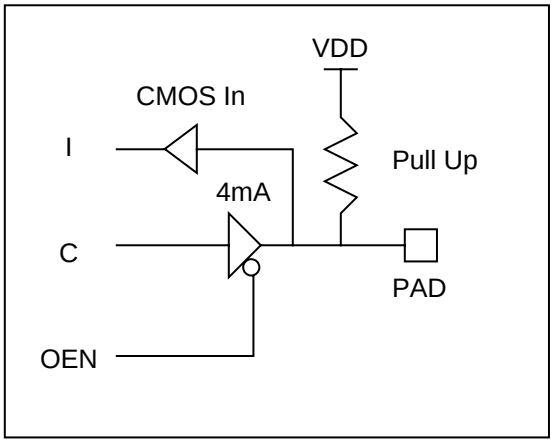
TXD



I/O Buffer PT1_[7:0]



I/O Buffer PT2_[1:0]



I/O Buffer PT2_[7:2]

9.2 Power Consumption

Fosc = 2MHz, @5V

Operation Mode	Min.	Typ.	Max.	Unit
Sleep Mode	0.5	1	--	μA
Normal Mode	--	10	12	mA
Download Mode	--	22	24	mA

10. Copyright Notice

10.1 Disclaimer

The information contained herein has been carefully prepared by RAiO to present as guidance to the application of RAiO's products. RAiO has done its best to make this specification correct. However, RAiO does not make any warrant, expressly or impliedly, to user free from defects or patent non-infringing of the technologies described in this specification whatever typographical errors, circuit errors, logic errors, design errors …etc. which may exist and unaware of by RAiO. User shall bear the risk itself and RAiO shall not assume any responsibility or liability of any kind of damages arising there from RAiO.

10.2 Copyright