8-bit MCU with 10-bit A/D Converter

Features

- 2.2V to 5.5V Input Voltage Range
- 8 I/O ports
- RAM size: 128 x 8 bits
- The STACK RAM is included.
- Program ROM size: 4K x 8 bits OTP
- 2 channels 10 bits A/D Converter input source.
- One set of 16-bit down count timer and one set of 8-bit timer.
- Operating temperature: -40 ~ +125 ℃
- Build-in Low Voltage Reset (LVR) circuit.
- Oscillator: Internal RC oscillation.

Description

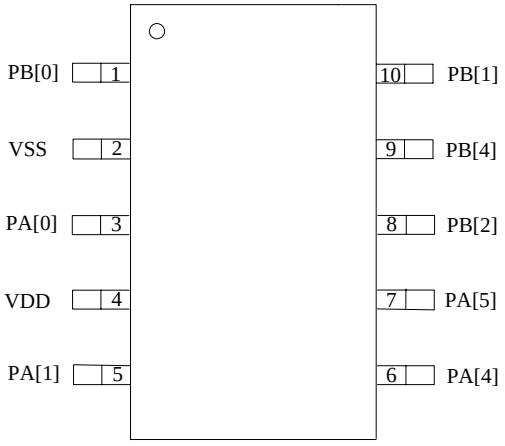
The EU1010 is an OTP type 8-bit micro-controller with 2 channels of 10-bit A/D converter using advanced CMOS process. The EU1010 is specially designed for various industrial field applications. The EU1010 incorporates two sets of 8-bit timer/counters, where timer0/1 is specially designed for PWM (Pulse Width Modulation) generator. There are 8 I/O ports in EU1010. Considering form factor and manufacturability, the EU1010 is packaged into 10-pin non-JEDEC-standard compact-size SOP.

Ordering Information

EU1010 – xxxxXX – XX X

Eorex MCU	OTP code	Package	Packing
1010 : 8 I/O ports	xxxx : Customer/Product code XX : OTP code serial no.	SU: SOP	R: Taping & Reel T: Tray U: Tube

Pin Assignment



SOP-10 (8 I/O port)

Pin Description

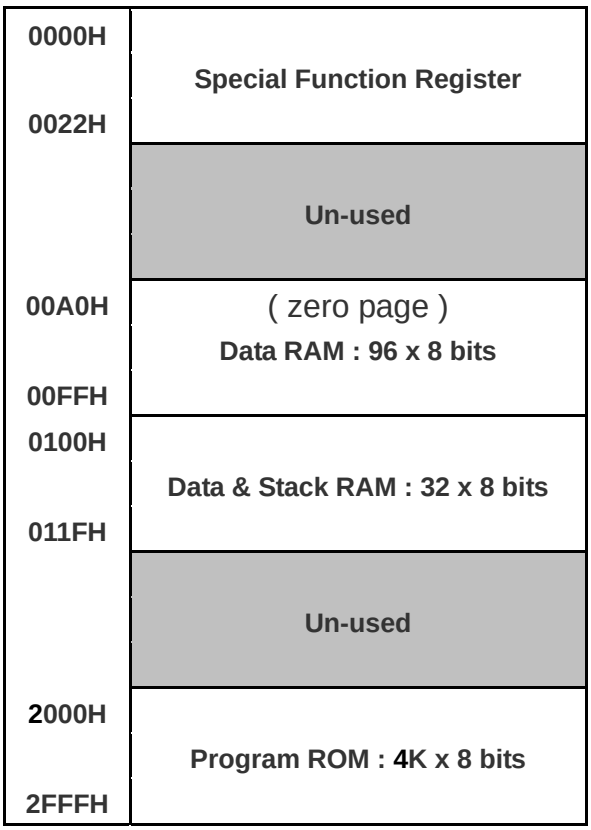
Pin Name	I/O	Function
V_{DD}	-	Positive power supply
V_{SS}	-	Ground
PA0~PA1 PA4~PA5	I/O	Input and output ports. In input mode, all PA pins could be set an internal pull-up resistor (R=100Kohm). In output mode, it could be optioned as CMOS or NMOS output independently. Also, PA0~PA3 could be optioned as 4 channels of 10-bit A/D converter input pins.
PB0~PB1 PB2 / PB4	I/O	Input and output ports. In input mode, it could have optioned internal pull-up resistor with 100Kohm. Also, it can be set as HALT and STOP mode released source when the input signal is changed from high to low. In output mode, it could be optioned as CMOS or NMOS output independently.

MCU Description

Item	Abbreviation	Description
Accumulator	ACC	The EU1010 has an 8-bit register used for data hold, exchange, arithmetic, transfer and I/O operation.
Index Register	(X,Y)	In EU1010 that builds in two index registers (X and Y). These two index registers could be used to count program steps or to provide an index value to be used in generating as effective address. When executing an instruction which specifies indexed addressing, the MCU fetches the operation code and the base address, and modifies the address by adding the index register to its prior to perform the desired operation. Per or post-index of indirect address is possible.
Special Function Register	SFR	The EU1010 includes some special function control registers in zero-page. For more detailed information about the special function registers, please refer to next section.
Stack pointer register	SP	The stack pointer is an 8-bit register that is used to control the addressing of the variable-length stack. The stack pointer is automatically incremented and decremented under control of the microprocessor the perform stack manipulations under direction of either the program or interrupts. The stack allows a simple implementation of nested subroutines and multiple level interrupts. The stack pointer is initialized by the user's software.
Program counter	PC	The 14-bit program counter register provides the address that step the microprocessor through sequential program instructions.
Program Flag Register	P	The 8-bit status flag register contains seven status flags. Some of the flags are controlled by the program and others may be controlled both by the program and the MCU. Instruction set contains a number of conditional branch instructions that are designed to allow testing of these flags.

Memory

The EU1010 has a 14-bit program counter which can totally address 16K x 8 bits. All the special function registers, data RAM, stack RAM and program ROM are assigned in this area as the following diagram.



Figure_A Memory mapping diagram

Program ROM

The EU1010 contains 4Kx8 bit program OTP and its addressing size is assigned from $2000H to $2FFFH. After power on reset, the data restored in addresses $2FFCH and $2FFDH are loaded into program counter. It means that RESET vector address is located in $2FFCH and $2FFDH.

Data RAM and STACK RAM

The EU1010 data RAM are located from $A0H to $11FH. All the data RAM area from $A0H to $0FFH could be accessed by zero-page addressing mode. That STACK RAM could be accessed from $100H to $11FH. Actually, the area of $100H ~ $11FH, could be accessed as data RAM or STACK RAM. User needs to preset stack pointer (SP) after power on reset. Program Counter (PC) should be initialized after power on reset that ROM address $2FFCH and $2FFDH content will be loaded into Program Counter.

Special Function Register (SFR)

Special function register area is during the addresses from $00H to $22H. These special function registers control all I/O and timer function setting. Some of memory addresses are not defined and un-used. It is unnecessary for the users to read or write data from these undefined areas. User should follow the default value or do not access these undefined area. All its functions are listed in next sections.

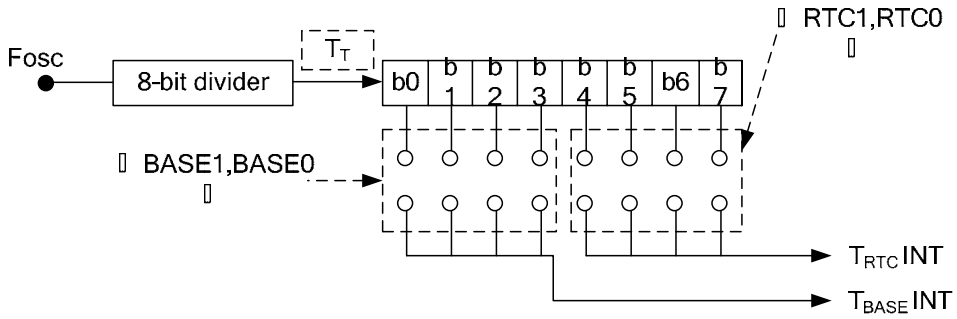
Interrupt and Timers

Address	00H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	INTEN	ADINT	#	#	T1INT	T0INT	PBINT	TBINT	RTCINT
Read or Write		R/W	#	#	R/W	R/W	R/W	R/W	R/W
Default Value		0	#	#	0	0	0	0	0

The EU1010 builds in 8 interrupt sources: Real Time Clock interrupt, Time Base interrupt, PB input ports falling edge interrupt, Timer0 underflow interrupt, Timer1 underflow interrupt, A/D converter interrupt and RESET interrupt. By setting control register INTEN (00H) that can enable or disable corresponding interrupt sources. Interrupt source, interrupt vector address mapping and its priority are shown in the following table.

Address	Interrupt Source	Priority
2FFF0H/2FF1H	ADC interrupt	7
2FF2H/2FF3H	Timer1 interrupt	6
2FF4H/2FF5H	Timer0 interrupt	5
2FF6H/2FF7H	PB port interrupt	4
2FF8H/2FF9H	Time base interrupt	3
2FFAH/2FFBH	RTC interrupt	2
2FFCH/2FFDH	Reset	1
2FFEH/2FFFH	Reserved	0

RTC timer, Base timer interrupt and Watch Dog timer



Address	03H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	UCCLK	#	BASE1	BASE0	RTC1	RTC0	RCSEL2	RCSEL1	RESEL0
Read or Write		W	W	W	W	W	W	W	W
Default Value		0	0	0	0	0	0	0	0

Real time counter

The EU1010 build-in a real time counter interrupt and its clock source is Fosc. Its frequency could be optioned by UCCLK.4 & UCCLK.3 that could select real time interrupt frequency. For example, if the clock source come from ExCLK (when CKSEL=1) and the input frequency is 32,768Hz, select “[RTC1, RTC0] = [0, 1]” and it will cause 2Hz interrupt when real time counter interrupt enabled (RTCINT=1) & (RTCEN=1).

RTC1	RTC0	T_{RTC}
0	0	$T_T/32$
0	1	$T_T/64$
1	0	$T_T/128$
1	1	$T_T/256$

Base Timer

Base timer is optioned from [BASE1, BASE0] and it will cause a period timer base interrupt enabled (TBINT=1) & (TBEN=1).

BASE1	BASE0	T_{BASE}
0	0	$T_T/2$
0	1	$T_T/4$
1	0	$T_T/8$
1	1	$T_T/16$

MCU clock selection

With a built-in internal RC oscillator, the MCU frequency (Fosc) is 4MHz ±3% @4.5V. MCU system clock (Fsys) offers 4MHz, 2MHz, 1MHz, 500KHz or 250KHz by MCU clock options bit; RCSEL2~0. Please refer to the following table.

Address	**03H**	**bit7**	**bit6**	**bit5**	**bit4**	**bit3**	**bit2**	**bit1**	**bit0**
Name	**UCCLK**	**#**	**BASE1**	**BASE0**	**RTC1**	**RTC0**	**RCSEL2**	**RCSEL1**	**RCSEL0**
Read or Write		**#**	**W**	**W**	**W**	**W**	**W**	**W**	**W**
Default Value		**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

The MCU system clock could be switched by setting RCSEL2~0. First, pre-set RCSEL2~0 register and then execute “HALT” instruction. After then, MCU wake-up from HALT mode, MCU system frequency is already changed to the new setting frequency.

RCSEL2	**RCSEL1**	**RCSEL0**	**Frequency of MCU system clock (Fsys)**
0	**0**	**0**	**250KHz**
0	**0**	**1**	**500KHz**
0	**1**	**0**	**1MHz**
0	**1**	**1**	**2MHz**
1	**#**	**#**	**4MHz (Fosc)**

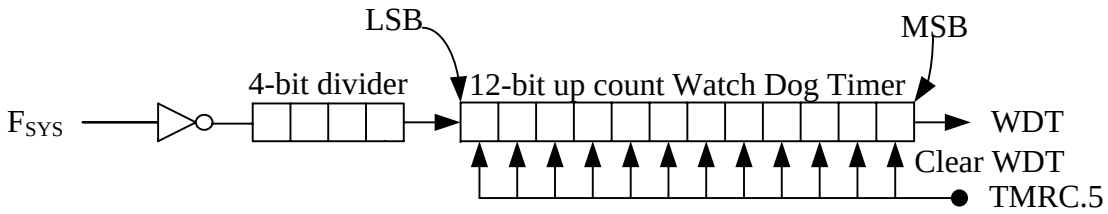
TMRC register

Address	01H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	TMRC	WDT Divide		WDTEN	WDTF	TBEN	TBF	RTCEN	RTCF
Read or Write		W	W	R/W	R/W	R/W	R/W	R/W	R/W
Default Value		0	0	0	#	0	#	0	#

*TMRC.0(RTCF) : real time counter transient flag. Once TRTC signal is transient, this flag will be set as RTCF=1 by hardware. This bit could be cleared by software.
*TMRC.1(RTCEN) : real time counter enable/disable flag.
RTCEN = 1, enable real time counter;
RTCEN = 0, disable real time counter.
*TMRC.2(TBF) : base timer transient flag. Once TBASE signal is transient, this flag will be set as TBF=1 by hardware. This bit could be cleared by software.
*TMRC.3(TBEN) : Base timer enable/disable flag.
TBEN = 1, enable base timer;
TBEN = 0, disable base timer.

Watchdog Timer

Watchdog timer block diagram is shown as figure_B. The clock source comes from CPU system clock.



Figure_B

Note:

*Once TMRC.5 (WDTEN) is set as “1”, the watchdog timer will start to count till the watchdog timer overflows, and then the TMRC.4 (WDTF) is set as “1”. Meanwhile, CPU will have a warm reset by hardware and the data in addresses $2FFCH and $2FFDH will be loaded into program counter. Watchdog timer can be cleared by setting TMRC.5 (WDTEN=0). **Please note well that the EU1010 watchdog timer is preset as disable after power on reset. Once watchdog timer is enabled by setting TMRC.5=1, watchdog timer won’t be stopped by software. Set TMRC.5=0 will just clear watchdog timer counter.**

WDT divider		
TMRC (01H)	bit7	bit6
FSYS/2	0	0
FSYS/4	0	1
FSYS/8	1	0
FSYS/16	1	1

PB ports interrupt

PB6~0, in input mode, could be optioned as external interrupt source by setting PBINT (00H.2) = 1. When the interrupt enabled and external signal changed from high to low, the PB port interrupt will take into action and its interrupt vector is $2FF6H and $2FF7H.

A falling edge signal at PB ports will wake up CPU from HALT or STOP mode. When PB port interrupt is enabled (PBINT=1), CPU will wake up from HALT or STOP mode, and serve PB port interrupt first and then execute next instruction. If PB port interrupt is disabled, CPU will just be waked up and then execute next instruction only. User should check which PB port the falling edge signal comes from by PBF control register. If the falling edge is from PB0, the PBF.0 will be set to “1” by hardware. These flags could be cleared by software.

Address	02H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	PBF	#	PBF6	PBF5	PBF4	PBF3	PBF2	PBF1	PBF0
Read or Write		#	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default Value		#	0	0	0	0	0	0	0

Timer0

Timer0 is an 8-bit down count timer. Its clock source comes from CPU main-oscillator (FOSC) or ExCLK, which is listed in figure 4-3. User can preset timer0 counter by setting data into timer0 preload buffer T0BF(04H). The data read from T0BF(04H) will be the current count of timer0.

Timer0 will down count by every input clock when T0EN=1. When timer0 down count from 00H to FFH, T0F will be set to “1” and if T0INT =1, the timer0 interrupt will occur. Timer0 will automatically reload data from T0BF/04H (timer0 preset buffer). Therefore, user can preset timer0 new data into T0BF(04H) before timer0 underflow and cause different interrupt time duties. **That is, timer0 data will be loaded from T0BF buffer after T0EN bit is set as “1” or timer0 underflows.**

Address	04H	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Name	T0BF	1/0	1/0	1/0	1/0	1/0	1/0	1/0	1/0
Read or Write		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default Value		#	#	#	#	#	#	#	#

Address	05H	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Name	T0C	T0EN	T0F	T0CK	#	#	Timer0 pre-scaler		
Read or Write		R/W	R/W	R/W	#	#	R/W	R/W	R/W
Default Value		0	#	#	#	#	#	#	#

Note:

*T0EN : timer0 enable flag

T0EN = 1, load T0BF content (preload buffer data) into timer0 and enable timer0 start to down count.

T0EN = 0, stop timer0 counting. User can get timer0 data by reading T0BF register. (LDA T0BF)

* when T0EN=0, write data to T0BF(04H), data will be directly passed to timer0 counter.

T0F : timer0 underflow flag

T0F = 1, timer0 underflow;

T0F = 0, timer0 not underflow.

*T0CK : timer0 clock source option bit, need **set to 0 in EU1010.**

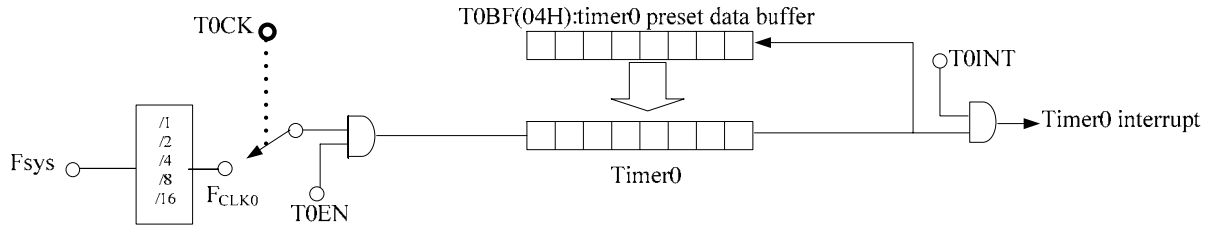
T0CK = 0, clock source from Fosc.

***There is no default data with T0CK bit, user should preset this bit after power on reset.**

bit2	bit1	bit0	Timer0 pre-scaler(F_{CLK0}=)
0	0	0	F_{SYS}/1
0	0	1	F_{SYS}/2
0	1	0	F_{SYS}/4
0	1	1	F_{SYS}/8
1	#	#	F_{SYS}/16

Note:

***Bit2~0 are timer0 clock source selection bits. They must follow the setting listed below. If the bit2=1, it would be divided by 16 while bit1 and bit0 could be any data. When T0CK=1, don't care bit2~0 data.



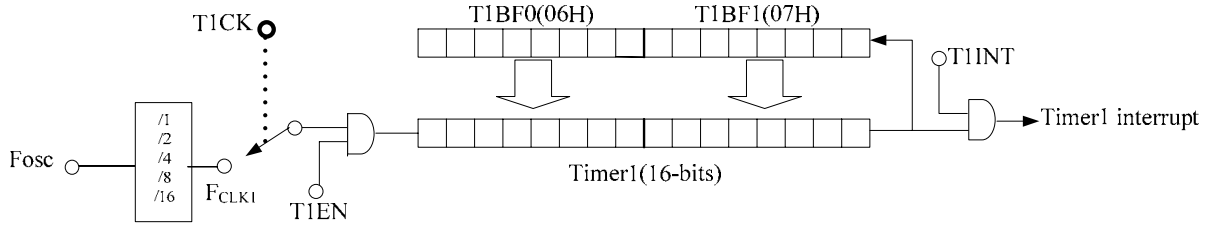
Figure_C

Timer1

Address	06H	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Name	T1BF0	1/0	1/0	1/0	1/0	1/0	1/0	1/0	1/0
Read or Write		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default Value		#	#	#	#	#	#	#	#

Address	07H	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Name	T1BF1	1/0	1/0	1/0	1/0	1/0	1/0	1/0	1/0
Read or Write		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default Value		#	#	#	#	#	#	#	#

Timer1 is a 16-bit down count timer. T1BF0(06H)/low byte and T1BF1(07H)/high byte are timer1 preload buffer. Timer1 clock source could come from CPU main oscillator (Fosc), refer to figure_D. Timer1 can work in two kinds of operating mode. One mode is normal 16-bit timer/counter mode and the other one is specially designed for different interrupt time period.



Figure_D

Timer/counter mode

If T1C.4 (08H.4)=PWM1=0, timer1 works as 16-bit timer/counter mode, register timer1 will down count by every input clock when timer1 turns on by T1EN=1. When timer1 is down count from 0000H to FFFFH, T1F bit will be set to “1”. At the same time, if T1INT=1, the timer1 interrupt occurs. Timer1 will automatically reload data from preload buffer T1BF0(06H) and T1BF1(07H). Therefore, user can preset T1BF0(06H) and T1BF1(07H) buffer data before different interrupt time duties caused by timer1

underflow. It means that timer1 data will be loaded from T1BF0(06H) and T1BF1(07H) buffer after T1EN bit is set as “1” or timer1 underflow.

Address	08H	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Name	T1C	T1EN	T1F	**T1CK**	**PWM**	#	`Timer1 pre-scaler		
Read or Write		W	R/W	**R/W**	**W**	#	W	W	W
Default Value		0	#	#	**0**	#	#	#	#

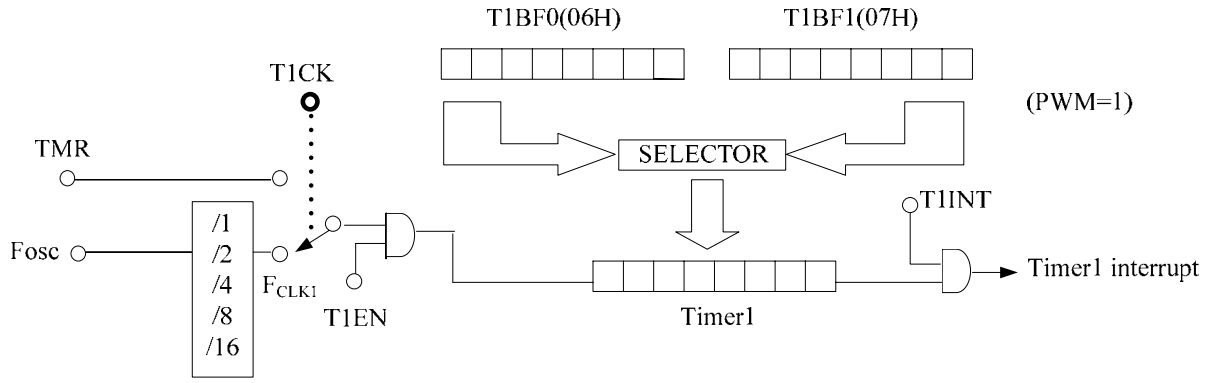
Note:

*T1EN : timer1 enable flag
T1EN = 1, enable timer1 start to down count;
T1EN = 0, stop timer1 down count. (when T1EN=0 and write data into T1BF0(06H) and T1BF1(07H) , the data will be directly passed to timer1 counter.)
*T1F : timer1 underflow flag
T1F = 1, timer1 underflow;
T1F = 0, timer1 not underflow.

Bit2~bit0 are timer1 clock source selection bits. It must follow the setting as below.

bit2	bit1	bit0	Timer1 pre-scaler (FCLK1=)
0	0	0	Fosc/1
0	0	1	Fosc/2
0	1	0	Fosc/4
0	1	1	Fosc/8
1	#	#	Fosc/16

PWM Mode



Set T1C.4=1(PWM=1), Timer-1 will work in PWM mode. User can preset T1BF0 and T1BF1. Timer1 interrupt will have two kinds of time duties based on T1BF0 and T1BF1 data.

I/O ports

PA ports (PA0, PA1, PA4 and PA5) are 4-bits I/O ports. System can either output data by writing data into PA(10H) ports or read data from input mode by reading PA(10H) ports.

When PA ports are set in output mode, it can be optioned by software as CMOS or NMOS output. Set in input mode, PA ports can be optioned as internal pull up or input with floating status. Also, PA6~0 could be optioned as A/D converter analog signal input pins. Please refer to A/D Converter section.

Address	10H	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Name	PA	#	#	1/0	1/0	#	#	1/0	1/0
Default Value		#	#	#	#	#	#	#	#

PAIO register: control PA port either as input mode or output mode. PAIO register can be set partially bits in input mode and partial bits in output mode.

Address	11H	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Name	PAIO	#	#	1/0	1/0	#	#	1/0	1/0
Default Value		#	#	1	1	#	#	1	1

Note:

*PAIO.n = 1, set as input mode.
PAIO.n = 0, set as output mode.

(Please be noted that PAIO.0 default data is "0".)
That n = 0, 1, 4 and 5.

PACN register: control output mode as CMOS or NMOS output.

Address	12H	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Name	PACN	#	1/0	1/0	1/0	1/0	1/0	1/0	1/0
Default Value		#	0	0	0	0	0	0	0

Note:

*PACN.n = 1, set as CMOS output;

PACN.n = 0, set as NMOS output.

That n = 0, 1, 4 and 5.

PAPH register: enable/disable PA ports internal pull high when PA port is set as input mode.

Address	13H	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Name	PAPH	#	1/0	1/0	1/0	1/0	1/0	1/0	1/0
Default Value		#	0	0	0	0	0	0	0

Note:

*PAPH.n = 1, internal pull high resister enable;

PAPH.n = 0, internal pull high resister disable, where n = 0, 1, 4 and 5.

PB ports

PB ports (PB0, PB1, PB2 and PB4) are 4bit I/O ports. User can either output data by writing data into PB(14H) ports. Or read data at input mode by reading PB(14H) ports.

Address	14H	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Name	PB	#	#	#	1/0	#	1/0	1/0	1/0
Default Value		#	#	#	#	#	#	#	#

PBR register

Address	15H	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Name	PBR	#	#	#	1/0	#	1/0	1/0	1/0
Default Value		#	#	#	0	#	0	0	0

Note:

*PBR.n = 1, enable Port B Interrupt and HALT or STOP mode released by input falling edge signal;

PBR.n = 0, disable Port B Interrupt and HALT or STOP mode release, where n = 0, 1, 2 and 4.

*Setting PBR.n register as “1” can enable to release HALT or STOP mode by PB port input signal from high to low. After HALT or STOP mode released, the oscillator will oscillate at the same time. If PBINT=1, PB-interrupt subroutine will start. If PBINT=0, the program counter will execute the next instruction after HALT or STOP.

PBIO register: control PB port as input or output mode. Also, it can be set as partial bits in input mode and partial bits in output mode.

Address	16H	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Name	PBIO	#	#	#	1/0	#	1/0	1/0	1/0
Default Value		#	#	#	1	#	1	1	1

Note:

*PBIO.n = 1, set as input mode;

PBIO.n = 0, set as output mode.;

That n =0, 1, 2 and 4.

PBMD register

Address	17H	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Name	PBMD	#	#	#	1/0	#	1/0	1/0	1/0
Default Value		#	#	#	0	#	0	0	0

Note:

*When PBIO corresponding bit is optioned as output mode (PBIO.n=0), PBMD.n will work as CMOS or NMOS option bit.
(PBIO.n=0, PB.n works as output mode)
PBMD.n = 0, PB.n is NMOS output.
PBMD.n = 1, PB.n is CMOS output.

*When PBIO corresponding bit is optioned as input mode (PBIO.n=1), PBMD.n will work as internal pull-up resistor enable bit or disable bit.
(PBIO.n=1, PB.n works as input mode)
PBMD.n = 0, disable PB.n internal pull-up resistor,
PBMD.n = 1, enable PB.n internal pull-up resistor.
That n = 0, 1, 2 and 4.

A/D Converter

The EU1010 is built-in with 4 channels of 10-bit Analog to Digital Converter. The analog signal input pins are shared from PA3~0. These options are controlled by ADIN(09H) and ADCC(0AH) registers.

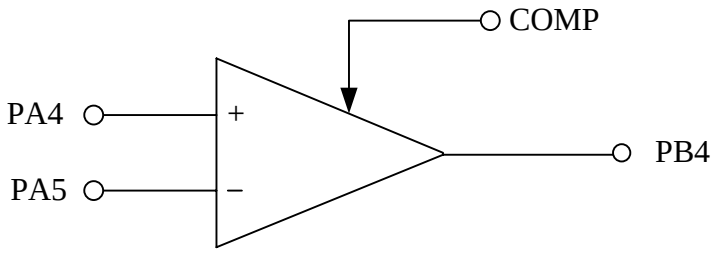
ADIN register:

PA3~PA0 ports could be optioned as A/D converter analog signal input pins. User can select partial or all **PA3**~PA0 pins as analog input pins by **ADIN3**~0 for corresponding bit.

Address	09H	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Name	ADIN	#	#	#	**COMP**	ADIN3	ADIN2	ADIN1	ADIN0
Read or Write		#	#	#	**W**	W	W	W	W
Default Value		#	#	#	**0**	0	0	0	0

Note:

*ADIN3~0: option of PA3~PA0 as analog signal input pin.
*ADINn = 1, set PA3~0 pin as analog signal input pin;
ADINn = 0, set PA3~0 as normal input/output pin, where n=3~0.
* When PA3~0 is set as analog input pin(s), PA ports control register will be inactive to access. That is, when PA port is set as analog signal input pin, all PA I/O port functions will be blocked out.



Note:

*COMP = 0, disable internal comparator;
COMP = 1, enable internal comparator.
When COMP=1, PA4 and PA5 will act as comparator input pin, and then comparator output pin will be connected to PB4 internally.

ADCR register

Address	0AH	bit7	bit6	bit5	bit4	Bit3	bit2	bit1	bit0
Name	ADCR	ADEN	**STC**	#	ADS1	ADS0	#	#	#
Read or Write		W	R/W	#	W	W	#	#	#
Default Value		0	#	#	#	#	#	#	#

Note:

*ADCR.7(ADEN) : enable A/D converter

ADCR.7 = 1, turn on or enable A/D converter;
ADCR.7 = 0, turn off or disable A/D converter.

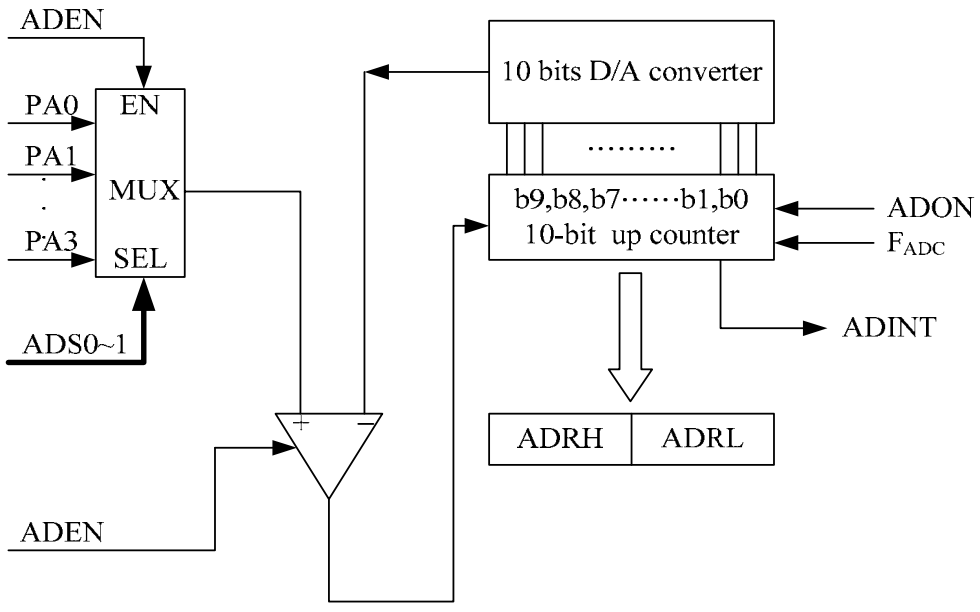
*ADCR.6(**STC**) : A/D converter finished interrupt flag, write a high to this bit to start ADC conversion.
ADCR.6 = 1, A/D converter translation is in BUSY.
ADCR.6 = 0, A/D converter is standby status.

ADS1/bit4	ADS0/bit3	Select pin
0	0	PA0
0	1	PA1
1	0	PA2
1	1	PA3

Note:

* ADCR.4~3(ADS1~0) : select acting analog channel input
* User can preset A/D converter sampling rate from ADRL.2~0 control register. The sampling clock comes from internal ring oscillator. The reference sampling rate setting is listed below.

A/D configuration and control block



Note:

*User should select PA3~0 ports as analog signal input pin(s) by setting ADIN.3~0. Set ADCC.7

(ADEN) = 1 to enable A/D converter circuit, and then select input pin from ADCC.4~3 (ADS1/ADS0). The converter clock comes from $F_{ADC.}$ After system completed an A/D converting cycle, ADC interrupt takes into action (if ADINT=1). User can read the converting data from ADRH and ADRL registers. [ADRH + ADRL] are totally 10 bits in 2 bytes control registers. These 2 bytes of register will keep the last A/D convert data. When A/D conversion are all completed, user should turn off A/D converter by setting ADCC.7 (ADEN) =0.

ADSP2~0 control register: A/D Converter sampling rate setting

Address	0BH	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Name	ADRL	Ad1	Ad0	#	#	#	**ADSP2**	**ADSP1**	**ADSP0**
Read or Write		R	R	#	#	#	W	W	W
Default Value		#	#	#	#	#	#	#	#

ADC sampling rate	ADSP2	ADSP1	ADSP0
8 Hz	0	0	0
16 Hz	0	0	1
32 Hz	0	1	0
64 Hz	0	1	1
128 Hz	1	0	0
256 Hz	1	0	1
512 Hz	1	1	0
1024 Hz	1	1	1

Address	0CH	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Name	ADRH	Ad9	Ad8	Ad7	Ad6	Ad5	Ad4	Ad3	Ad2
Read or Write		R	R	R	R	R	R	R	R
Default Value		#	#	#	#	#	#	#	#

Low Voltage Reset Circuit (or Power Failed Detector) 2.4V

The EU1010 is built in Low-Voltage-Reset circuit to detect and against power noise. The low voltage reset will be active when Vdd is dropped lower than 2.4V. Note that, in STOP mode, low voltage reset function will be disabled by hardware. The low voltage reset function will be active when CPU is not working in STOP mode.

HALT mode

The EU1010 will enter HALT mode by setting an instruction as “STA $0EH”. The data in ACC could be any data among #00H~#FFH. In HALT mode, CPU core will suspend and just hold at that program counter. All the internal circuit will suspend, except system clock and timer/counter kept running. HALT mode could be released from timer underflow or PB ports input signal from high to low. When halt mode is released, program counter will execute next instruction after “STA $0EH”. If timer interrupt is enabled, it serves timer interrupt first, and then execute next instruction after “STA $0EH”. When HALT mode is released by PB ports falling-edge, program counter execute next instruction. If PB interrupt is enabled, PBINT =1, PB interrupt subroutine will be activated.

Address	0EH	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Name	HALT	#	#	#	#	#	#	#	#

STOP mode

When executed “STA $0FH”, whole chip will enter stop mode. The data in ACC could be any data of #00H~#FFH. In stop mode, system clock and timer/counter will be stopped. At this condition, operating current could be down less than 1μA. Only input signal from high to low of PB ports can release the chip from STOP mode. When STOP mode is released by PB ports, program counter will execute next instruction of “STA $0FH”. If PB interrupt is enabled, PBINT=1, PB interrupt subroutine will be activated first.

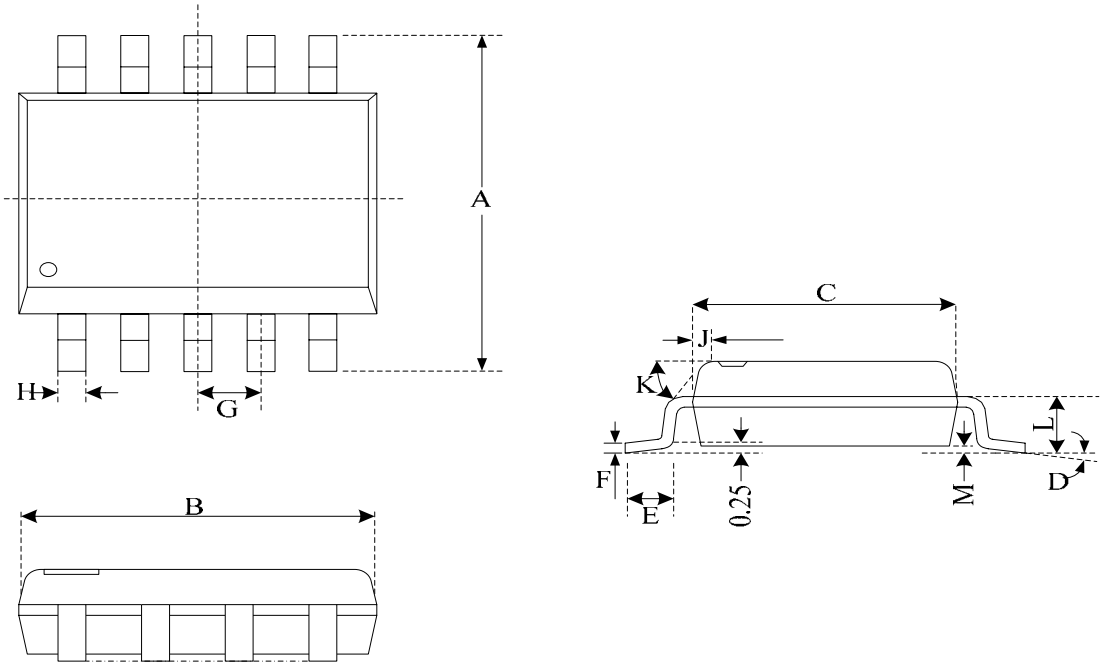
Address	0FH	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Name	STOP	#	#	#	#	#	#	#	#

MCU system clock

This CPU is built-in internal RC oscillator whose frequency is 4MHz. The tolerance of the frequency is within **3%**.

Package Description

SOP-10L



REF.	DIMENSIONS	
	Millimeters	
	Min	Max
A	5.80	6.20
B	4.80	5.00
C	3.80	4.00
D	0°	8°
E	0.40	0.90
F	0.19	0.25
M	0.10	0.25
H	0.30	0.44
L	1.35	1.75
J	0.375 REF.	
K	45°	
G	1.00 TYP.	

Revision History

Revision 0.1 (Jul 2007)
- First release

Revision 0.2 (Aug. 2007)....
- Pin Description .

Revision 0.3 (Apr. 2008)....
- EU101X .-> EU1010

Revision 0.4 (Jul. 2008)....
- Modify Memory / Address

Revision 0.5 (Feb. 2009)....
- Modify -40C~125C

Revision 0.6 (Feb. 2009)....
- Modify ordering information
  (xxxx = customer/product code)