(Preliminary)

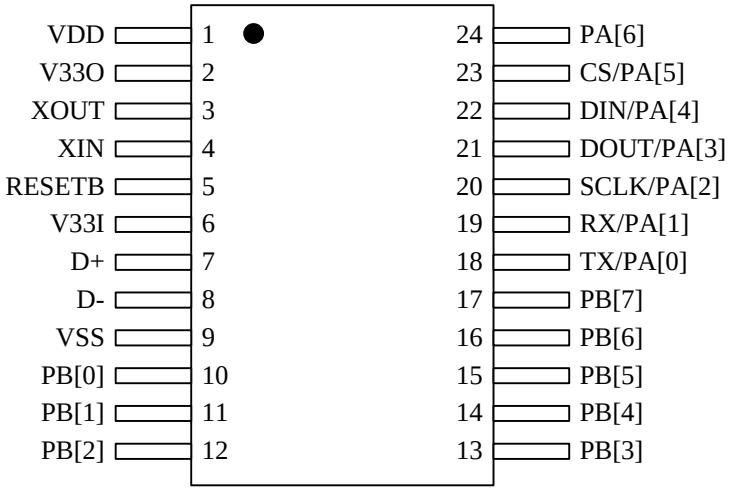
1. GENERAL DESCRIPTION

The MK8A01P device is an integrated micro-controller that includes an 8-bit RISC CPU core, 128-bytes SRAM, full speed USB interface and a 4K x 16 internal program OTP-ROM for USB general purpose application.

2. FEATURES

- USB 2.0 full speed
  - Fully compliant to USB 2.0 full speed specification
  - Built-in USB Transceiver
  - Support USB Suspend and Resume
  - One Control IN/OUT ,one Interrupt IN endpoints and two Bulk IN/OUT endpoint
- Building one 3.3 Voltage regulator
- Endpoint share with SRAM and has swap function (Can use software to set target endpoint)
- Built-in 128 byte internal SRAM
- Built-in 4K x 16 OTP (One Time Programming) ROM
- 6 or 12MHz instruction rate with 12MHz crystal oscillation (configure option)
- Built-in 4 or 8MHz (by option) clock output pin
- Two 8 bit auto reload timer interrupt which can cascade to be one 16 bit timer
- TX/RX serial I/O interface (UART) speed can up to 3Mbps
- SPI serial interface and speed up to 3 or 6MHz (configure option)
- I/O port
  - I/O port can be set to work on 3V
  - I/O port can be set to pull-up by register
  - 15 programmable bi-direction I/O with interrupt feature
- 18/20/24 pin package

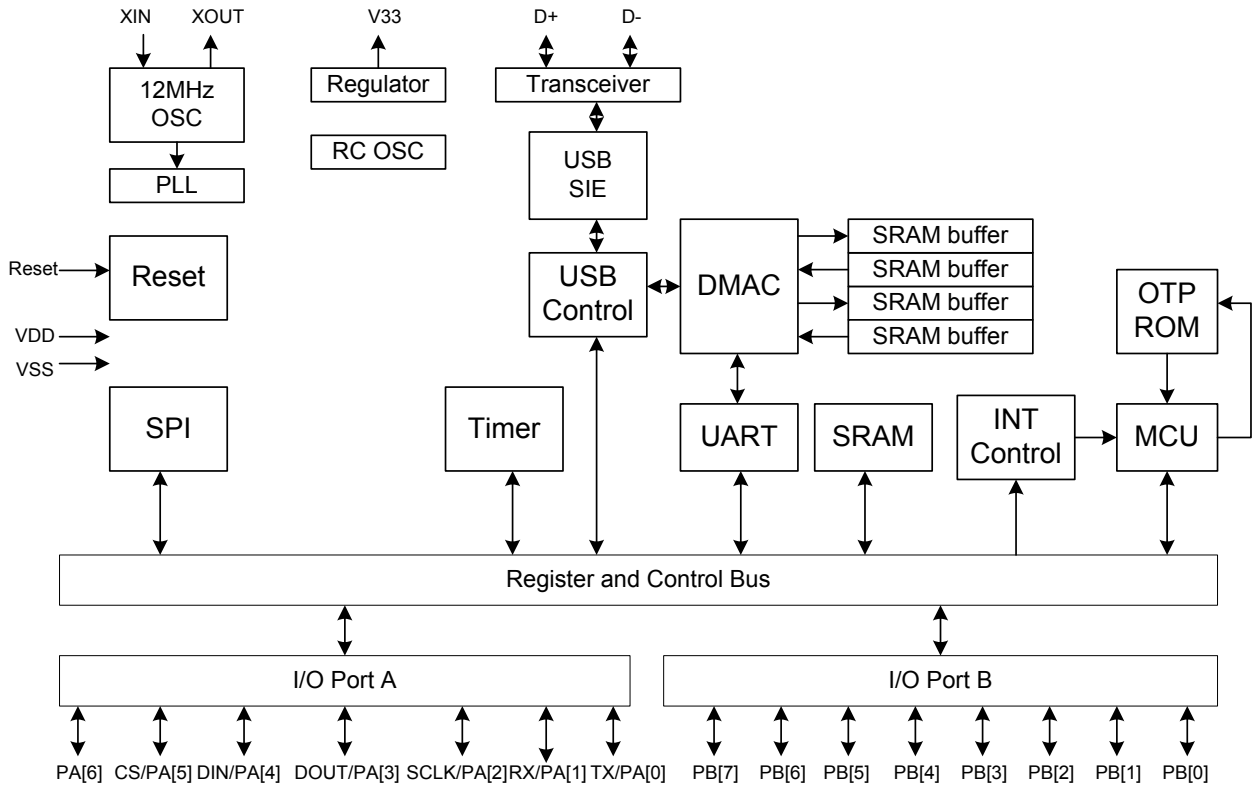
3. PIN DESCRIPTION



Package Types : SOP

Pin	I/O	Description
VDD	P	5V Power from USB cable
V33O	P	3.3V Power output
VSS	P	Ground
XOUT	O	Crystal out
XIN	I	Crystal in (12MHz)
RESETB	I	System Reset signal (Low active)
V33I	O	3.3V Power input (Connect to V33O)
D+	I/O	USB plus data line
D-	I/O	USB minus data line
TX/PA[0]	I/O	Asynchronous serial I/O transmission or Port A[0] by option
RX/PA[1]	I/O	Asynchronous serial I/O receiving or Port A[1] by option
SCLK/PA[2]	I/O	SPI serial clock or Port A[2] by option
DOUT/PA[3]	I/O	SPI data out or Port A[3] by option
DIN/PA[4]	I/O	SPI data in or Port A[4] by option
CS/PA[5]	I/O	SPI chip select or Port A[5] by option
CLKO/PA[6]	I/O	General purpose I/O, pull-high through software control Or 4M/8MHz clock output pin
PB[7:0]	I/O	General purpose I/O, pull-high through software control

4. BLOCK DIAGRAM



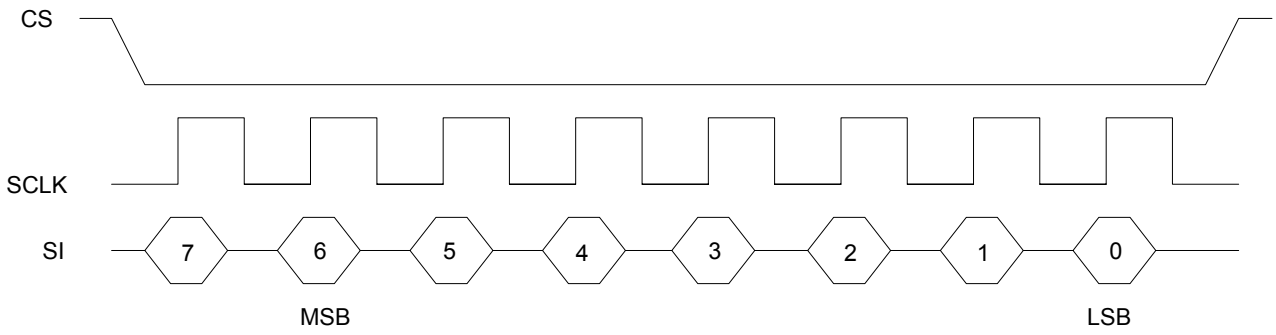
5. FUNCTION DESCRIPTION

5.1 Timer 0/1

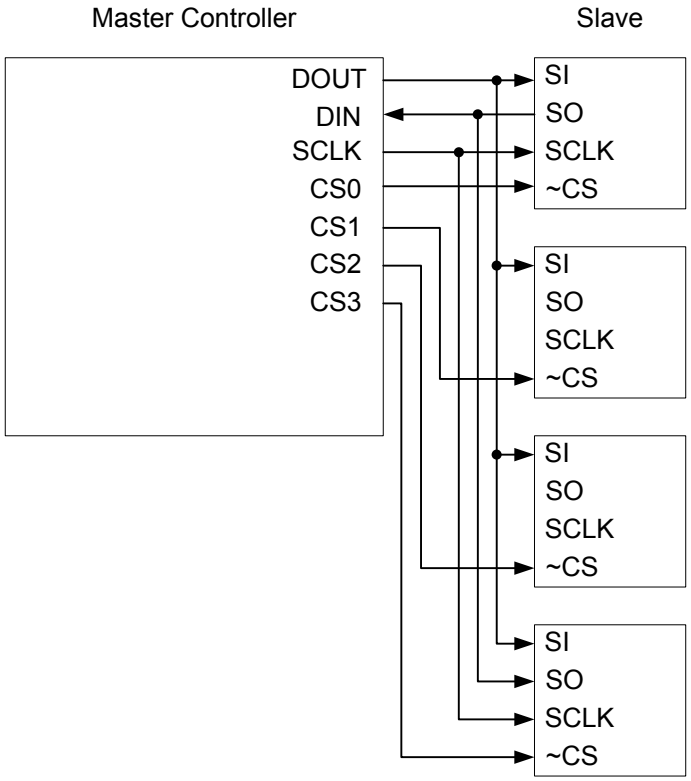
- The timer counter contains two 8 bits programmable one-shot/auto-reload count-up timer and can be cascaded into an one 16 bits timer
- Pre-scalar range can be set 6 or 12M/1, /8, /16, /32, /64, /128, /256
- Timer capture interrupt

5.2 SPI

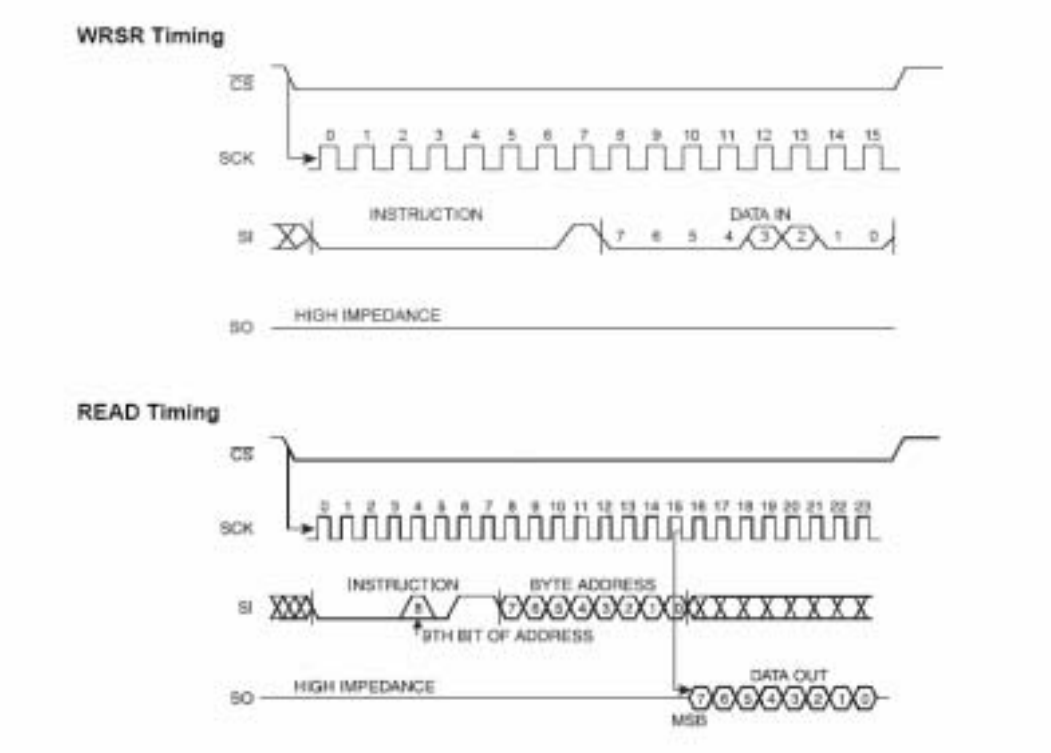
- SPI clock will be 3MHz or 6MHz by option
- 8 bits transmit buffer, 8 bits receive buffer and 8 bits shift register
- Shift out complete interrupt happen
- Just only Master at mode 0



- **Connection**



- **Waveform**



5.3 USB engine

- Fully compliant to USB 2.0 / Full Speed (12 Mbps) specification.
- Complete device configuration
- EP 0 for control IN/OUT. EP 0 have 8 bytes receive buffer and 8 bytes transmit buffer.
- EP 1 for bulk IN
- EP 2 for bulk OUT
- EP 3 for interrupt IN. EP 3 have 8 bytes transmit buffer.
- EP 1/2 bulk IN/OUT transaction data can be access via DMA
- Maximal allowable bulk IN/OUT transaction data buffer are 4x64 bytes

5.4 UART

- Operating at 48Mhz
- No handshake support
- Sampling the incoming data with x16 frequency
- 8 bits buffer and 8bits shift register for transmitter, and receiver also
- 14 bits prescaler for baud rate generator
- 8 bits RX time-out register
- Transmit/receive complete interrupt, RX time out interrupt
- Data Format and Baud Rate

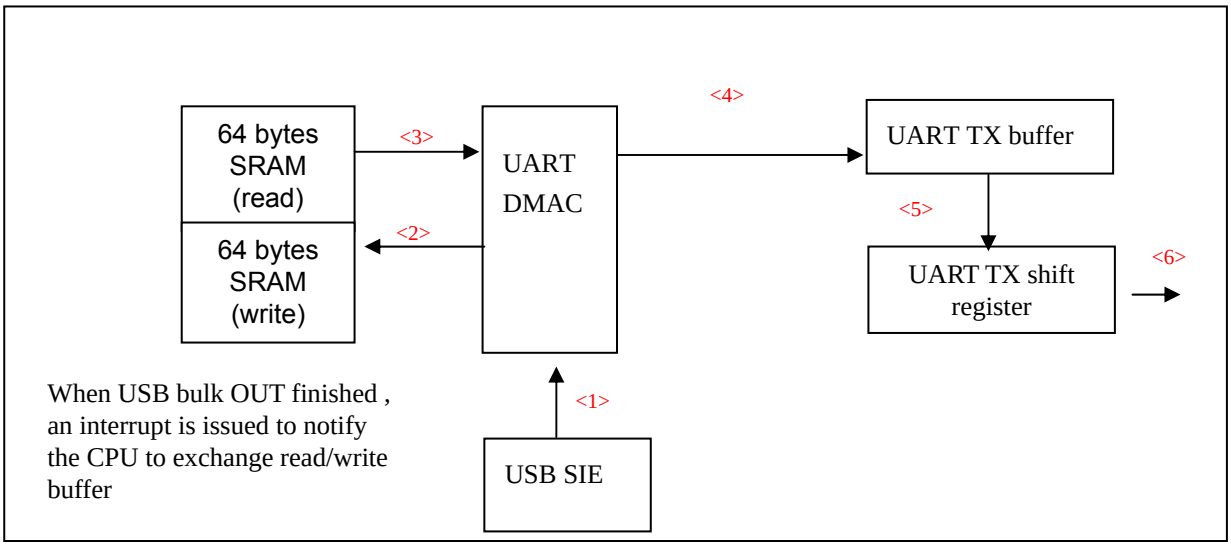
Data bits	7,8
Stop bits	1,2
Parity type	Odd, Even, None
Baud rate	300,600,1200,1800,2400,4800,7200,9600,14400 19200,28800,38400,56000,57600,115200,128000 230400,460800,921600,……3M

- **Baud rate error**

Standard Baud Rate	Actual Baud Rate in MU1001	Error [%]
1200	1200	0
2400	2400	0
4800	4792	-0.16
9600	9615	+0.17
19200	19231	+0.17
28800	28846	+0.17
38400	38462	+0.17
57600	57692	+0.17
115200	115385	+0.17
3M	3M	0

5.5 DMA Control

- **Bulk OUT data flow**

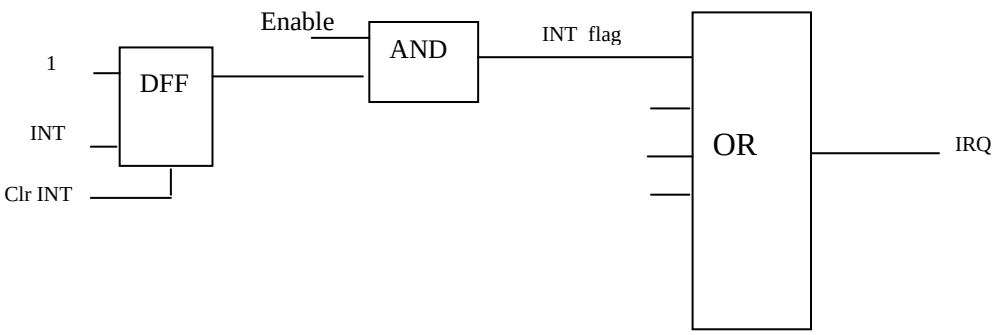


- 4 DMA channels to access the 4 blocks SRAM at same time
- Interface
  - Input [1:0] type.
    - 00: USB=> RAM
    - 01: UART=> RAM
    - 10: RAM=> USB
    - 11: RAM=> UART
  - Input [5:0] dma_tx_len
  - Input dma_auto
  - Input dma_start
  - Output [5:0] dma_rx_len
  - Output dma_int
- During DMA operating, CPU can not access the data of these 4 blocks SRAM

5.6 Interrupt

- **Interrupt source**
  - **USB control OUT (EP0) interrupt**
  - **USB control IN (EP0) interrupt**
  - **USB bulk IN (EP1) interrupt**
  - **USB bulk OUT (EP2) interrupt**
  - **USB interrupt IN (EP3) interrupt**
  - **USB reset interrupt**
  - **USB resume interrupt**
  - **USB suspend interrupt**
  - **DMA ch0 interrupt**
  - **DMA ch1 interrupt**
  - **DMA ch2 interrupt**
  - **DMA ch3 interrupt**
  - **SPI interrupt**
  - **UART TX interrupt**
  - **UART RX interrupt**
  - **UART RX time out interrupt**
  - **Timer0 interrupt**
  - **Timer1 interrupt**
  - **GPIO interrupt**
- **Single interrupt vector**
- **Clear interrupt before enable it**

- **Function block**



5.7 Power-down mode

When USB host issue a suspend command or some reason need to save the power, S/W can use SLEEP instruction to enter the power-down mode. In this mode, the crystal oscillator is stopped. The MK8A01P exits the power down mode using one of the following methods:

- USB bus resume
- USB bus reset
- External enabled GPIO interrupt

6. MEMORY MAP

Program address

0x0000	Reset vector
0x0004	Interrupt vector
0x0008 0x0FFF	4K program ROM

Register/SRAM address

0x00 0x5F	Register		
0x60 0x7F	32 bytes SRAM		
0x80 0xBF	96 bytes SRAM bank = 00	64 bytes SRAM bank = 01 (DMA ch0)	64 bytes SRAM bank = 10 (DMA ch2)
0xC0 0XDF		64 bytes SRAM bank = 01 (DMA ch1)	64 bytes SRAM bank = 10 (DMA ch3)
0xE0 0xFF			

(Preliminary)

7. REGISTER

Register name	Addr	R/W	Rst	Description
CONFIG_L	0x1000	R	111	Bit2: 0: code protect 1: no code protect Bit1: 0: CPU clock = 6Mhz 1: CPU cock = 12Mhz Bit0: 0: RESET pin is a normal function input pin 1: RESET pin is an external reset pin
CONFIG_H	0x1000	R		Reserved
INDF	0x00	R/W		Addressing this location uses contents of FSR to address data memory
PCL	0x01	R/W		Program counter (PC) Low
PCH	0x02	R/W		Program counter (PC) High
STATUS	0x03	R/W		Bit4: Reserved Bit3: Reserved Bit2: ALUZ , Zero bit Bit1: ALUDC , Digit carry/borrow bit Bit0: ALUC , Carry/Borrow bit
FSR	0x04	R/W		
RAM_BANK	0x05.7~ 0x05.6	R/W	0	SRAM bank select 00: user SRAM 01: DMA SRAM block 0,1 10: DMA SRAM block 2,3
	0x05.5~ 0x05.0			Reserved
USB_INT (0x06)				
EN_EP0_IN_I	0x06.7	R/W	0	Enable EP0 IN interrupt. (control transfers) 0 : disable 1: enable
EN_EP0_OUT_I	0x06.6	R/W	0	Enable EP0 OUT interrupt. (control transfers) 0: disable 1: enable
EN_EP1_IN_I	0x06.5	R/W	0	Enable EP1 IN interrupt. (bulk transfers) 0: disable 1: enable
EN_EP2_OUT_I	0x06.4	R/W	0	Enable EP2 OUT interrupt. (bulk transfers) 0: disable 1: enable
EN_EP3_IN_I	0x06.3	R/W	0	Enable EP3 IN interrupt (interrupt transfers) 0: disable 1: enable
EN_RST_I	0x06.2	R/W	0	Enable USB bus reset interrupt. 0: disable 1: enable
EN_RSM_I	0x06.1	R/W	0	Enable USB resume interrupt. 0: disable 1: enable
EN_SUSP_I	0x06.0	R/W	0	Enable USB suspend interrupt. 0: disable 1: enable
CH_MISC_INT_EN(0x07)				

EN_CH0_I	0x07.7	R/W	0	Enable DMA channel 0 interrupt. 0: disable 1: enable
EN_CH1_I	0x07.6	R/W	0	Enable DMA channel 1 interrupt. 0: disable 1: enable
EN_CH2_I	0x07.5	R/W	0	Enable DMA channel 2 interrupt. 0: disable 1: enable
EN_CH3_I	0x07.4	R/W	0	Enable DMA channel 3 interrupt 0: disable 1: enable
EN_SPI_I	0x07.3	R/W	0	Enable SPI interrupt 0: disable 1: enable
EN_UART_TX_I	0x07.2	R/W	0	Enable UART transmit interrupt 0: disable 1: enable
EN_UART_RX_I	0x07.1	R/W	0	Enable UART receive interrupt 0: disable 1: enable
EN_UART_RX_TOUT_I	0X07.0	R/W	0	Enable UART receive time out interrupt 0: disable 1: enable
TM_INT_EN(0x08)				
EN_TM0_I	0x08.7	R/W	0	Enable 8 bits Timer 0 interrupt 0: disable 1: enable
EN_TM1_I	0x08.6	R/W	0	Enable 8 bits or 16 bits Timer 1 interrupt 0: disable 1: enable
EN_GPIO_I	0x08.5	R/W	0	Enable GPIO interrupt. Use PA_I_E(0x44) and PB_I_E(0x46) to choose which GPIO is selected to be an external interrupt 0: disable 1: enable
EN_GBL_I	0x08.4	R/W	0	Enable Global interrupt 0: disable 1: enable
	0x08.3~ 0x08.0		0	
USB_INT_FLAG(0x09)				
EP0_IN_I	0x09.7	R/W	0	EP0 IN interrupt flag, write 0 to clear (control transfers)
EP0_OUT_I	0x09.6	R/W	0	EP0 OUT interrupt flag, write 0 to clear (control transfers)
EP1_IN_I	0x09.5	R/W	0	EP1 IN interrupt flag, write 0 to clear (bulk transfers)
EP2_OUT_I	0x09.4	R/W	0	EP2 OUT interrupt flag, write 0 to clear (bulk transfers)
EP3_IN_I	0x09.3	R/W	0	EP3 IN interrupt flag, write 0 to clear (interrupt transfers)
RST_I	0x09.2	R/W	0	USB bus reset interrupt flag, write 0 to clear
RSM_I	0x09.1	R/W	0	USB resume interrupt flag, write 0 to clear

SUSP_I	0x09.0	R/W	0	USB suspend interrupt flag, write 0 to clear
CH_MISC_INT_FLAG(0x0A)				
CH0_I	0x0A.7	R/W	0	DMA channel 0 interrupt flag, write 0 to clear
CH1_I	0x0A.6	R/W	0	DMA channel 1 interrupt flag, write 0 to clear
CH2_I	0x0A.5	R/W	0	DMA channel 2 interrupt flag, write 0 to clear
CH3_I	0x0A.4	R/W	0	DMA channel 3 interrupt flag, write 0 to clear
SPI_I	0x0A.3	R/W	0	SPI interrupt flag, write 0 to clear
UART_TX_I	0x0A.2	R/W	0	UART transmit interrupt flag, write 0 to clear
UART_RX_I	0x0A.1	R/W	0	UART receive interrupt flag, write 0 to clear
UART_RX_TOUT_I	0X0A.0	R/W	0	UART receive time out interrupt flag, write 0 to clear
TM_INT_FLAG(0x0B)				
TM0_I	0x0B.7	R/W	0	8 bits Timer 0 interrupt flag, write 0 to clear
TM1_I	0x0B.6	R/W	0	8 bits or 16 bits Timer 1 interrupt flag, write 0 to clear
GPIO_I	0x0B.5	R/W	0	GPIO interrupt flag, write 0 to clear
	0x0B.4~ 0x0B.0		0	
DMA				
CHX_TYPE(0x0C)				
CH0_TYPE	0x0C.7 ~ 0x0C.6	R/W	01	DMA channel 0 access type 00: USB => SRAM 01: UART => SRAM 10: SRAM => USB 11: SRAM => UART
CH1_TYPE	0x0C.5 ~ 0x0C.4	R/W	01	DMA channel 1 access type 00: USB => SRAM 01: UART => SRAM 10: SRAM => USB 11: SRAM => UART
CH2_TYPE	0x0C.3 ~ 0x0C.2	R/W	01	DMA channel 2 access type 00: USB => SRAM 01: UART => SRAM 10: SRAM => USB 11: SRAM => UART
CH3_TYPE	0x0C.1 ~ 0x0C.0	R/W	01	DMA channel 3 access type 00: USB => SRAM 01: UART => SRAM 10: SRAM => USB 11: SRAM => UART
CHX_EN(0x0D)				
CH0_AUTO	0x0D.7	W	0	DMA ch0 auto swap enable. Only the idle DMA channel is allowable to enable the auto swap mode. After enabled, this DMA channel will wait for the UART receive data full and then swap to be UART RX DMA channel. This mode is helpful to prevent the UART RX data lose if F/W is busy to swap the DMA channel manually. 0: disable 1: enable
CH1_AUTO	0x0D.6	W	0	DMA ch1 auto swap enable. Only the idle DMA channel is allowable to enable the auto swap mode. After enabled, this DMA channel will wait for the UART receive data full and then swap to be UART RX DMA channel. This mode is helpful to prevent the UART RX data lose if F/W is busy to swap the DMA channel manually. 0: disable

				1: enable
CH2_AUTO	0x0D.5	W	0	DMA ch2 auto swap enable. Only the idle DMA channel is allowable to enable the auto swap mode. After enabled, this DMA channel will wait for the UART receive data full and then swap to be UART RX DMA channel. This mode is helpful to prevent the UART RX data lose if F/W is busy to swap the DMA channel manually. 0: disable 1: enable
CH3_AUTO	0x0D.4	W	0	DMA ch3 auto swap enable. Only the idle DMA channel is allowable to enable the auto swap mode. After enabled, this DMA channel will wait for the UART receive data full and then swap to be UART RX DMA channel. This mode is helpful to prevent the UART RX data lose if F/W is busy to swap the DMA channel manually. 0: disable 1: enable
CH0_ABORT	0x0D.3	W	0	DMA ch0 abort. Write “1” to abort the designated DMA channel and its setting will reset to the default value.
CH1_ABORT	0x0D.2	W	0	DMA ch1 abort. Write “1” to abort the designated DMA channel and its setting will reset to the default value.
CH2_ABORT	0x0D.1	W	0	DMA ch2 abort. Write “1” to abort the designated DMA channel and its setting will reset to the default value.
CH3_ABORT	0x0D.0	W	0	DMA ch3 abort. Write “1” to abort the designated DMA channel and its setting will reset to the default value.
CH0_RX_LEN	0x0E	R	0	DMA channel 0 receive length. Maximal length is 64 bytes
CH1_RX_LEN	0x0F	R	0	DMA channel 1 receive length. Maximal length is 64 bytes
CH2_RX_LEN	0x10	R	0	DMA channel 2 receive length. Maximal length is 64 bytes
CH3_RX_LEN	0x11	R	0	DMA channel 3 receive length. Maximal length is 64 bytes
CH0_TX_LEN	0x12	W	0	DMA channel 0 transmit length. Maximal length is 64 bytes and 0 will send an zero length data packet. Cleared by H/W after transmission complete
CH1_TX_LEN	0x13	W	0	DMA channel 1 transmit length. Maximal length is 64 bytes and 0 will send an zero length data packet. Cleared by H/W after transmission complete
CH2_TX_LEN	0x14	W	0	DMA channel 2 transmit length. Maximal length is 64 bytes and 0 will send an zero length data packet. Cleared by H/W after transmission complete
CH3_TX_LEN	0x15	W	0	DMA channel 3 transmit length. Maximal length is 64 bytes and 0 will send an zero length data packet.. Cleared by H/W after transmission complete
CHX_STATUS(0x16)				
CH0_START	0x16.7	R/W	0	DMA channel 0 transfer start. Write 1 to start the transfer. Reset by H/W after transmission completed.
CH1_START	0x16.6	R/W	0	DMA channel 1 transfer start. Write 1 to start the transfer. Reset by H/W after transmission completed.
CH2_START	0x16.5	R/W	0	DMA channel 2 transfer start.

				Write 1 to start the transfer. Reset by H/W after transmission completed.
CH3_START	0x16.4	R/W	0	DMA channel 3 transfer start. Write 1 to start the transfer. Reset by H/W after transmission completed.
DMA_AUTO_FAIL	0x16.3	R	0	DMA channel 0-3 auto swap fail. F/W must write 1 to DMA_CHX_AUTO first, then read back this flag to make sure the auto swap mode is enabled. When fail, start the DMA by setting DMA_CHX_START. 0: enable 1: fail
	0x16.2~ 0x16.0			Reserved
UART				
UART_TX_DATA	0x17	W	0	UART transmit buffer. CPU can not write this register during DMA operation.
UART_RX_DATA	0x18	R	0	UART receive buffer
UART_BAUD	0x19~ 0x1A	W	0x FF FF	UART baud rate generator 48M/16/baud rate/2 EX: 300 bps 48M/16/300/2 = 1_0011_1000_1000 (0x19) = 1000_1000 (0x1A) = 0001_0011 all 0 = 3M bps all 1 = stop internal UART clock
UART_RX_TOUT_CFG	0x1B	W	0x FF	UART receive time out counter configuration Time out range 16~255 UART bit time. Time out counter start by received a start bit and cleared by valid data receiving.
UART_STATUS(0x1C)				
UART_DATA_CFG	0x1C.7	R/W	0	UART data bit configuration. 0: 8 bits 1: 7 bits
UART_PAR_CFG	0x1C.6 ~ 0x1C.5	R/W	0	UART parity bit configuration 00: no parity 01: odd parity 10: even parity
UART_STP_CFG	0x1C.4	R/W	0	UART stop bit configuration 0: 1 stop bit 1: 2 stop bit
UART_TX_START	0x1C.3	R/W	0	UART start to transmit. Write 1 to start the transmission. Reset by H/W after transmission completed.
UART_RX_ERR	0x1C.2 ~ 0x1C.1	R/W	0	UART receive error. Write 0 to clear. 00: no error 01: parity error 10: stop bit error
UART_RX_OV	0x1C.0	R/W	0	UART receive overflow. If no any DMA channel is assigned to receive the data from UART, and UART is received a correct data , then this flag will be set. Write 0 to clear.
SPI				
SPI_TX_DATA	0x1D	W	0	SPI transmit buffer
SPI_RX_DATA	0x1E	R	0	SPI receive buffer

SPI_CTRL(0x1F)				
SPI_CLK_SEL	0x1F.7~ 0x1F.6	R/W	0	SPI clock output select 00: 6 or 12 M = 6 or 12 Mbit/s 01: 6 or 12 M/2 = 3 or 6 Mbit/s 10: 6 or 12 M/8 = 0.75 or 1.5 Mbit/s 11: 6 or 12 M/64 = 0.09375 or 0.1875 Mbit/s
SPI_START	0x1F.5	R/W	0	SPI transmit/receive start. Write 1 to start the transfer. Reset by H/W after transmission completed.
	0x1F.4~ 0x1F.0			Reserved
USB				
EP0_RX_FIFO	0x20~ 0x27	R	0	EP0 receive data FIFO (8 bytes) (control transfers) 0x20 = data0, 0x27 = data7
EP0_TX_FIFO	0x28~ 0x2F	W	0	EP0 transmit data FIFO (8 bytes) (control transfers) 0x28 = data0 , 0x2F = data7
EP3_TX_FIFO	0x30~ 0x37	W	0	EP3 transmit data FIFO (8 bytes) 0x30 = data0 , 0x37 = data 7
USB_ADR	0x38	W	0	USB device address, cleared while chip reset or USB bus reset
EP0_STATUS(0x39)				
EP0_RX_LEN	0x39.7~ 0x39.4	R	0	EP0 receive data length (control transfers)
TOKEN	0x39.3~ 0x39.2	R	0	EP0 received token 00: SETUP token 01: IN token 10: OUT token
DATA_PKT	0x39.1	R	0	USB received DATA packet 0: DATA0 packet 1: DATA1 packet
DATA_ERR	0x39.0	R	0	USB received data error. 0: no error 1: error
EP0_3_LEN(0x3A)				
EP0_TX_LEN	0x3A.7~ 0x3A.4	W	0	EP0 transmit data length. 0 will send an zero length data packet. (control transfers)
EP3_TX_LEN	0x3A.3~ 0x3A.0	W	0	EP3 transmit data length. 0 will send an zero length data packet. (interrupt transfers)
USB_RDY_CTRL(0x3B)				
EP0_RX_RDY	0x3B.7	R/W	0	EP0 is ready to receive data. F/W should set this flag every time the device is ready to receive the data and H/W will reset this flag after transmission completed(EP0_OUT_I). If not ready, H/W will respond with NAK command once IN/OUT token received. 0: not ready 1: ready
EP0_TX_RDY	0x3B.6	R/W	0	EP0 is ready to transmit data. F/W should set this flag every time the device is ready to transmit the data and H/W will reset this flag after transmission completed(EP0_IN_I). If not ready, H/W will respond with NAK command once IN/OUT token received. 0: not ready 1: ready
EP1_RDY	0x3B.5	R/W	0	EP1 is ready to transmit data.

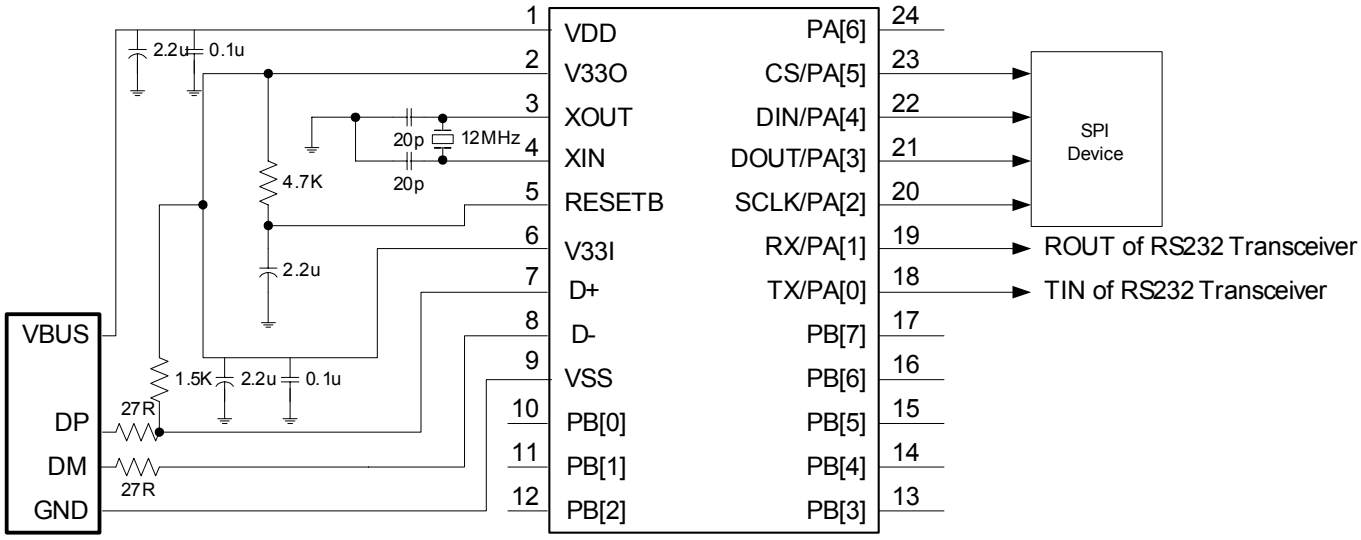
				F/W should set this flag every time the device is ready to transmit the data and H/W will reset this flag after transmission completed(EP1_IN_I). If not ready, H/W will respond with NAK command once IN/OUT token received. 0: not ready 1: ready
EP2_RDY	0x3B.4	R/W	0	EP2 is ready to receive data. F/W should set this flag every time the device is ready to receive the data and H/W will reset this flag after transmission completed(EP2_OUT_I). If not ready, H/W will respond with NAK command once IN/OUT token received. 0: not ready 1: ready
EP3_RDY	0x3B.3	R/W	0	EP3 is ready to transmit data. F/W should set this flag every time the device is ready to transmit the data and H/W will reset this flag after transmission completed(EP3_IN_I). If not ready, H/W will respond with NAK command once IN/OUT token received. 0: not ready 1: ready
SUSPND	0x3B.2	R/W	0	USB device suspend mode enable write “1” or “0” to enable or disable the suspend mode
	0x3B.1~ 0x3B.0			Reserved
USB_CTRL(0x3C)				
RESUME	0x3C.7	W	0	Force USB device send RESUME signal to USB host while in suspend mode. F/W should set this flag for 10~15ms to send RESUME signal.
EP1_CFG	0x3C.6	W	0	Indicate the EP1 is a configured end point. After SET_CNFIGURATION request received, F/W should set this flag or reset this flag when received SET_CONFIGURATION with a value of 0
EP2_CFG	0x3C.5	W	0	Indicate the EP2 is a configured end point. After SET_CNFIGURATION request received, F/W should set this flag or reset this flag when received SET_CONFIGURATION with a value of 0
EP3_CFG	0x3C.4	W	0	Indicate the EP3 is a configured end point. After SET_CNFIGURATION request received, F/W should set this flag or reset this flag when received SET_CONFIGURATION with a value of 0
	0x3C.3 ~ 0x3C.0			Reserved
USB_DATA_CTRL(0x3D)				
EP0_STALL	0x3D.7	R/W	0	EP0 stalled. This is used for received an unknown command from host or unable to transmit or receive data 0: not stall 1: stall
EP1_STALL	0x3D.6	R/W	0	EP1 stalled. This is used for received an unknown command from host or unable to transmit or receive data 0: not stall 1: stall

EP2_STALL	0x3D.5	R/W	0	EP2 stalled. This is used for received an unknown command from host or unable to transmit or receive data 0: not stall 1: stall
EP3_STALL	0x3D.4	R/W	0	EP3 stalled. This is used for received an unknown command from host or unable to transmit or receive data 0: not stall 1: stall
EP0_TGL	0x3D.3	R/W	0	Read this flag will read out the received toggle bit of EP0. Write this flag to select DATA0 or DATA1 packet to transmit via endpoint 0
EP1_TGL_CLR	0x3D.2	R/W	0	EP1 data toggle bit cleared to DATA0. When CLEAR FEATURE command received, write “1” to clear the toggle bit
EP2_TGL_CLR	0x3D.1	R/W	0	EP2 data toggle bit cleared to DATA0. When CLEAR FEATURE command received, write “1” to clear the toggle bit
EP3_TGL_CLR	0x3D.0	R/W	0	EP3 data toggle bit cleared to DATA0. When CLEAR FEATURE command received, write “1” to clear the toggle bit
GPIO				
PA_DATA	0x3E	R/W	0	Port A [6:0] data read/write
PA_DIR	0x3F	R/W	0	Port A [6:0] input/output direction. 0: input 1: output
PA_PUL_UP	0x40	R/W		Port A [6:0] pull-up resistor enable 0: enable 1: disable
PB_DATA	0x41	R/W	0	Port B [7:0] data read/write
PB_DIR	0x42	R/W	0	Port B [7:0] input/output direction 0: input 1: output
PB_PUL_UP	0x43	R/W	0	Port B [7:0] pull-up resistor enable 0: enable 1: disable
PA_INT_E	0x44	R/W	0	Port A [6:0] interrupt enable
PA_INT_EDGE	0x45	R/W	0	Port A [6:0] interrupt edge select 0: rising edge 1: falling edge
PB_INT_E	0x46	R/W	0	Port B [7:0] interrupt enable
PB_INT_EDGE	0x47	R/W	0	Port B [7:0] interrupt edge select 0: rising edge 1: falling edge
GPIO_SEL(0x48)				
GPIO_SPI_SEL	0x48.7	R/W	0	GPIO SPI select 0: GPIO 1: SPI
GPIO_UART_SEL	0x48.6	R/W	0	GPIO UART select 0: GPIO 1: UART
RST_PIN	0x48.5	R	0	When RESET pin is configured as normal function input pin, Read it from this flag.

PA6_4M_SEL	0x48.4~ 0x48.3	R/W	0	Port A[6] GPIO or fixed 4/8Mhz output select 00 : GPIO 01 : fixed 4 Mhz output 1X : fixed 8 Mhz output
	0x48.2~ 0x48.0			Reserved
Timer 0/1				
TM0_CTRL(0x49)				
TM0_DIV	0x49.7~ 0x49.5	R/W	0	8 bits Timer 0 or 16 bits cascaded timer divider 000: 8 bits timer 0 or cascaded 16 bits timer stop 001: 6 or 12 M 010: 6 or 12 M/8 011: 6 or 12 M/16 100: 6 or 12 M/32 101: 6 or 12 M/64 110: 6 or 12 M/128 111: 6 or 12 M/256
TM0_AUTO	0x49.4	R/W	0	8 bits timer 0 or 16 bits cascaded timer auto reload select 0: auto-reload 1: single shot
TM0_EN	0x49.3	R/W	0	8 bits timer 0 or 16 bits cascaded timer enable. After divider and auto-reload setting finished, enable this flag to start the timer. 0: disable 1: enable
TM_CASCADE	0x49.2	R/W	0	Timer 0 ,1 cascade select 0: individual 8 bits timer 0 and timer 1 1: cascaded 16 bits timer
	0x49.1~ 0x49.0			Reserved
TM1_CTRL(0x4A)				
TM1_DIV	0x4A.7~ 0x4A.5	R/W	0	8 bits Timer 1 timer divider 000 : 8 bits timer 0 or cascaded 16 bits timer stop 001 : 6 or 12 M 010 : 6 or 12 M/8 011 : 6 or 12 M/16 100 : 6 or 12 M/32 101 : 6 or 12 M/64 110 : 6 or 12 M/128 111 : 6 or 12 M/256
TM1_AUTO	0x4A.4	R/W	0	8 bits timer 1 auto reload select 0: auto-reload 1: single shot
TM1_EN	0x4A.3	R/W	0	8 bits timer 1 enable. After divider and auto-reload setting finished, enable this flag to start the timer. 0: disable 1: enable
	0x4A.2~ 0x4A.0			
TM0_PREL	0x4B	W	0	Timer 0 pre-load register
TM0_CNT	0x4C	R/W	0	Timer 0 counter value. Only “0” can be written to clear the counter.
TM1_PREL	0x4D	W	0	Timer 1 pre-load register
TM1_CNT	0x4E	R/W	0	Timer 1 counter value. Only “0” can be written to clear the

				counter.
MISC				
SPEED_MOD	0x4F.3	R	1	CPU speed mode status 0:6MHz 1:12MHz
	0x4F.2~ 0x4F.0			Reserved
BANK_SET(0x50)				
TAB_BANK	0x50.7~ 0x50.4	W	0	Table read instruction bank select. When use TABRDH/TABRDL instruction to read the data from ROM, assign these 4 bits flags first to address the corresponding address of ROM code. 0000: 0x000 ~ 0x0FF 0001: 0x100 ~ 0x1FF 0010: 0x200 ~ 0x2FF 0011: 0x300 ~ 0x3FF 0100: 0x400 ~ 0x4FF 0101: 0x500 ~ 0x5FF 0110: 0x600 ~ 0x6FF 0111: 0x700 ~ 0x7FF 1000: 0x800 ~ 0x8FF 1001: 0x900 ~ 0x9FF 1010: 0xA00 ~ 0xAFF 1011: 0xB00 ~ 0xBFF 1100: 0xC00 ~ 0xCFF 1101: 0xD00 ~ 0xDFF 1110: 0xE00 ~ 0xEFF 1111: 0xF00 ~ 0xFFF
AUTO_BANK	0x50.3	W	0	0: no auto bank 1: auto bank
	0x50.2~ 0x50.0			Reserved
USR_REG	0x51	R/W	0	User register.

8. APPLICATION CIRCUIT



<Note> If choose internal reset function, the RESETB pin must connect to V33I.

9. INSTRUCTION SET

JUMP INSTRUCTION				
LCALL I	Call subroutine. However, LCALL can addressing 4K address	2	None	011i iiii iiii iiii
LGOTO I	Go branch to any address	2	None	010i iiii iiii iiii
JZ I	If z=1 then jump to any address	2	None	001i iiii iiii iiii
JC I	If c=1 then jump to any address	2	None	000i iiii iiii iiii
LOGIC				
AND M, a	(M). (acc) → (acc)	1	Z	1010 1000 MMMM MMMM
AND M, m	(M). (acc) → (M)	1	Z	1010 1001 MMMM MMMM
ANDLA I	Immediate. (acc) → (acc)	1	Z	1111 1000 iiii iiii
COM M, a	~(M) → (acc)	1	Z	1010 0100 MMMM MMMM
COM M, m	~(M) → (M)	1	Z	1010 0101 MMMM MMMM
IOR M, a	(M) or (acc) → (acc)	1	Z	1011 1110 MMMM MMMM
IOR M, m	(M) or (acc) → (M)	1	Z	1011 1111 MMMM MMMM
IORLA I	Immediate or (acc) → (acc)	1	Z	1111 0010 iiii iiii
RL M, a	Rotate left from m to acc m[6:0]→acc[7:1] m[7]→ acc[0]	1	None	1110 0000 MMMM MMMM
RL M, m	Rotate left from m to itself m[6:0]→m[7:1] m[7]→ m[0]	1	None	1110 0001 MMMM MMMM
RR M, a	Rotate right from m to acc m[0]→acc[7] m[7:1]→acc[6:0]	1	None	1110 1000 MMMM MMMM
RR M, m	Rotate right from m to itself m[0]→m[7] m[7:1]→m[6:0]	1	None	1110 1001 MMMM MMMM
SWAP M, a	m[7:4] → acc[3:0] m[3:0] → acc[7:4]	1	None	1011 1100 MMMM MMMM
SWAP M, m	m[7:4] ←→ m[3:0]	1	None	1011 1101 MMMM MMMM
XOR M, a	(M) xor (acc) → (acc)	1	Z	1011 0110 MMMM MMMM
XOR M, m	(M) xor (acc) → (M)	1	Z	1011 0111 MMMM MMMM
XORLA I	Immediate xor (acc) → (acc)	1	Z	1111 1001 iiii iiii
MATHEMATICS				
ADD M, a	(M)+(acc) → (acc)	1	C, DC, Z	1010 1010 MMMM MMMM
ADD M, m	(M)+(acc) → (M)	1	C, DC, Z	1010 1011 MMMM MMMM
ADDLA I	Immediate + (acc) → (acc)	1	C, DC, Z	1111 1010 MMMM MMMM
BC M, bn	Clear bit n of (M)	1	None	1001 1bbb MMMM MMMM
BS M, bn	Set bit n of (M)	1	None	1001 0bbb MMMM MMMM
CLRA	Clear accumulator	1	Z	1010 0010 0000 0000
CLR M	Clear memory M	1	Z	1010 0011 MMMM MMMM
TABRDL M	Read low byte ROM table (ROM bank)	2	None	1101 1000 MMMM MMMM
TABRDH M	Read high byte ROM table (ROM bank)	2	None	1101 1001 MMMM MMMM

DA M, a	Decimal Adjust M to ACC If ACC[3:0] > 9 or DC=1 Then ACC[3:0]←ACC[3:0]+6, DC1=DC else ACC[3:0] ←ACC[3:0], DC1=0 If ACC[7:4]+DC1 > 9 or C=1 Then ACC[7:4]←ACC[7:4]+6+DC1, C=1 else ACC[7:4] ←ACC[7:4]+DC1, C=C	1	C	1101 0110 MMMM MMMM
DA M, m	Decimal Adjust M to memory If ACC[3:0] > 9 or DC=1 Then M[3:0]←ACC[3:0]+6, DC1=DC else M[3:0] ←ACC[3:0], DC1=0 If ACC[7:4]+DC1 > 9 or C=1 Then M[7:4]←ACC[7:4]+6+DC1, C=1 else M[7:4] ←ACC[7:4]+DC1, C=C	1	C	1101 0111 MMMM MMMM
DEC M, a	(M) - 1 → (acc)	1	Z	1010 1100 MMMM MMMM
DEC M, m	(M) - 1 → (M)	1	Z	1010 1101 MMMM MMMM
INC M, a	(M) + 1 → (acc)	1	Z	1011 0000 MMMM MMMM
INC M, m	(M) + 1 → (M)	1	Z	1011 0001 MMMM MMMM
MOVAM m	(acc) → (M)	1	None	1010 0001 MMMM MMMM
MOV M, a	(M) → (acc)	1	Z	1010 0110 MMMM MMMM
MOV M, m	(M) → (M)	1	Z	1010 0111 MMMM MMMM
MOVLA I	Immediate data → acc	1	None	1111 0000 iiii iiii
SUBLA I	(immediate data)-(Acc)→(Acc)	1	C, DC, Z	1111 0100 iiii iiii
SUB M, m	(M)–(acc) → (M)	1	C, DC, Z	1011 0101 MMMM MMMM
SUB M, a	(M) –(acc) → (acc)	1	C, DC, Z	1011 0100 MMMM MMMM
OTHER OPERATION				
NOP	No operation	1	None	1111 1111 1111 1111
CLRWDT	Clear watch-dog register	1	$\overline{TO}$, $\overline{PD}$	1111 1111 1111 0000
RET	Return (for lcall instruction)	2	None	1111 1111 1111 0001
IRETI	Return and enable INTM (for IRQ)	2	None	1111 1111 1111 0010
IRET	Return (for IRQ)	2	None	1111 1111 1111 0011
SLEEP	Enter sleep (saving) mode, Have to need add 2 NOP	1	$\overline{TO}$, $\overline{PD}$	1111 1111 1111 100
CONDITION OPERATION				
BTSC M, bn	If bit n of(M)=0, skip next instruction	1 or 2	None	1000 1bbb MMMM MMMM

BTSS M, bn	If bit n of (M)=1, skip next instruction	1 or 2	None	1000 0bbb MMMM MMMM
TMSS A	If (acc) =0, skip next instruction	1 or 2	None	1011 1000 XXXX XXXX
TMSC M	If (M) = 0, skip next instruction	1 or 2	None	1011 1001 MMMM MMMM