ADC Program Examples for Products AT89C51CCxx, T89C51AC2, T89C5115

References

- Atmel 8051 Microcontrollers Hardware Manual

8051 Microcontrollers

Application Note

Rev. 4361C–80C51–11/04

1. Introduction

This Application Note provides to customers C and Assembler program examples for ADC.

These examples are developped for the different configuration modes of this feature. The Code example targets T89C51CC01, please replace the line # include ‘T89C51CC01.h’ with the corresponding line for AT89C51CC03, T89C51CC02, AT89C51AC3, T89C51AC2, T89C5115 products.

2. C Examples

2.1 8 bits ADC

```
/**
 * @file $RCSfile: Adc_8bits.c,v $
 *
 * Copyright (c) 2004 Atmel.
 *
 * Please read file license.txt for copyright notice.
 *
 * @brief This file is an example to use Adc.
 *
 * This file can be parsed by Doxygen for automatic documentation
 * generation.
 * Put here the functional description of this file within the software
 * architecture of your program.
 *
 * @version $Revision: 1.0 $ $Name:  $
 */

/* @section  I N C L U D E S */
#include "t89c51cc01.h"

unsigned char value_converted=0x00; /* converted value */
unsigned char value_AN6=0x00;       /* converted AN6 value */
unsigned char value_AN7=0x00;       /* converted AN7 value */
bit end_of_convertion=0;            /* software flag */



/**
 * FUNCTION_PURPOSE:this function setup Adc with channel 6 and 7 and start
 * 8bits convertion.
 * FUNCTION_INPUTS:void
 * FUNCTION_OUTPUTS:void
 */
void main(void)
{
/* configure channel P1.6(AN6) and P1.7(AN7) for ADC */
ADCF = 0xC0;

/* init prescaler for adc clock */
/* Fadc = Fperiph/(2*(32-PRS)), PRS -> ADCLK[4:0] */
ADCLK = 0x06;                       /* Fosc = 16 MHz, Fadsc = 153.8khz */

ADCON = 0x20;                       /* Enable the ADC */

EA = 1;                             /* enable interrupts */
EADC = 1;                           /* enable ADC interrupt */
```

```
while(1)
{
   ADCON &= ~0x07;                  /* Clear the channel field ADCON[2:0] */
   ADCON |= 0x06;                   /* Select channel 6 */
   ADCON &= ~0x40;                  /* standard mode */
   ADCON |= 0x08;                   /* Start conversion */

   while(!end_of_convertion);       /* wait end of convertion */
   end_of_convertion=0;             /* clear software flag */
   value_AN6=value_converted;       /* save converted value */


   ADCON &= ~0x07;                  /* Clear the channel field ADCON[2:0] */
   ADCON |= 0x07;                   /* Select channel 7 */
   ADCON &= ~0x40;                  /* standard mode */
   ADCON |= 0x08;                   /* Start conversion */

   while(!end_of_convertion);       /* wait end of convertion */
   end_of_convertion=0;             /* clear software flag */
   value_AN7=value_converted;       /* save converted value */

}
}


/**
 * FUNCTION_PURPOSE:Adc interrupt, save ADDH into an unsigned char
 * FUNCTION_INPUTS:void
 * FUNCTION_OUTPUTS:void
 */
void it_Adc(void) interrupt 8
{
ADCON &= ~0x10;                     /* Clear the End of conversion flag */
value_converted = ADDH;             /* save value */
end_of_convertion=1;                /* set flag */
}
```

2.2 10bits ADC

```
/**
 * @file $RCSfile: Adc_10bits.c,v $
 *
 * Copyright (c) 2004 Atmel.
 *
 * Please read file license.txt for copyright notice.
 *
 * @brief This file is an example to use Adc.
 *
 * This file can be parsed by Doxygen for automatic documentation
 * generation.
 * Put here the functional description of this file within the software
 * architecture of your program.
 *
 * @version $Revision: 1.0 $ $Name:  $
 */

/* @section  I N C L U D E S */
#include "t89c51cc01.h"

unsigned int value_converted=0x0000;/* converted value */
unsigned int value_AN6=0x0000;      /* converted AN6 value */
unsigned int value_AN7=0x0000;      /* converted AN7 value */
bit end_of_convertion=0;            /* software flag */




/**
 * FUNCTION_PURPOSE:this function setup Adc with channel 6 and 7 and start
 * 10bits convertion.
 * FUNCTION_INPUTS:void
 * FUNCTION_OUTPUTS:void
 */
void main(void)
{

/* configure channel P1.6(AN6) and P1.7(AN7) for ADC */
ADCF = 0xC0;

/* init prescaler for adc clock */
/* Fadc = Fperiph/(2*(32-PRS)), PRS -> ADCLK[4:0] */
  ADCLK = 0x06;                     /* Fosc = 16 MHz, Fadsc = 153.8khz */

ADCON = 0x20;                       /* Enable the ADC */


EA = 1;                             /* enable interrupts */
EADC = 1;                           /* enable ADC interrupt */
```

```
while(1)
{
   ADCON &= ~0x07;                  /* Clear the channel field ADCON[2:0] */
   ADCON |= 0x06;                   /* Select channel 6 */
   ADCON &= ~0x40;                  /* standard mode */
   ADCON |= 0x08;                   /* Start conversion */

   while(!end_of_convertion);       /* wait end of convertion */
   end_of_convertion=0;             /* clear software flag */
   value_AN6=value_converted;       /* save converted value */


   ADCON &= ~0x07;                  /* Clear the channel field ADCON[2:0] */
   ADCON |= 0x07;                   /* Select channel 7 */
   ADCON &= ~0x40;                  /* standard mode */
   ADCON |= 0x08;                   /* Start conversion */

   while(!end_of_convertion);       /* wait end of convertion */
   end_of_convertion=0;             /* clear software flag */
   value_AN7=value_converted;       /* save converted value */

}
}


/**
 * FUNCTION_PURPOSE:Adc interrupt, save ADDH and ADDL into an unsigned int
 * FUNCTION_INPUTS:void
 * FUNCTION_OUTPUTS:void
 */
void it_Adc(void) interrupt 8
{
ADCON &= ~0x10;                     /* Clear the End of conversion flag */
value_converted = ADDH<<2;          /* save 8 msb bits */
value_converted |= (ADDL & 0x03);   /* save 2 lsb bits */
end_of_convertion=1;                /* set flag */
}
```

2.3 SFR Register Definition

```
/*H*************************************************************************
**
* NAME: T89C51CC01.h
*-------------------------------------------------------------------------
-
* PURPOSE: inlcude file for KEIL
*************************************************************************
*/
#ifndef _T89C51CC01_H_

#define _T89C51CC01_H_

#define Sfr(x, y)  sfr x = y
#define Sbit(x, y, z)   sbit x = y^z
#define Sfr16(x,y)sfr16 x = y

/*---------------------------------------*/
/* Include file for 8051 SFR Definitions  */
/*---------------------------------------*/

/*  BYTE Register  */
Sfr (P0 , 0x80);
Sfr (P1 , 0x90);

Sbit (P1_7, 0x90, 7);
Sbit (P1_6, 0x90, 6);
Sbit (P1_5, 0x90, 5);
Sbit (P1_4, 0x90, 4);
Sbit (P1_3, 0x90, 3);
Sbit (P1_2, 0x90, 2);
Sbit (P1_1, 0x90, 1);
Sbit (P1_0, 0x90, 0);

Sfr (P2 , 0xA0);

Sbit (P2_7 , 0xA0, 7);
Sbit (P2_6 , 0xA0, 6);
Sbit (P2_5 , 0xA0, 5);
Sbit (P2_4 , 0xA0, 4);
Sbit (P2_3 , 0xA0, 3);
Sbit (P2_2 , 0xA0, 2);
Sbit (P2_1 , 0xA0, 1);
Sbit (P2_0 , 0xA0, 0);


Sfr (P3 , 0xB0);

Sbit (P3_7 , 0xB0, 7);
Sbit (P3_6 , 0xB0, 6);
```

```
Sbit (P3_5 , 0xB0, 5);
Sbit (P3_4 , 0xB0, 4);
Sbit (P3_3 , 0xB0, 3);
Sbit (P3_2 , 0xB0, 2);
Sbit (P3_1 , 0xB0, 1);
Sbit (P3_0 , 0xB0, 0);

Sbit (RD , 0xB0, 7);
Sbit (WR , 0xB0, 6);
Sbit (T1 , 0xB0, 5);
Sbit (T0 , 0xB0, 4);
Sbit (INT1, 0xB0, 3);
Sbit (INT0, 0xB0, 2);
Sbit (TXD , 0xB0, 1);
Sbit (RXD , 0xB0, 0);

Sfr (P4 , 0xC0);

Sfr (PSW , 0xD0);

Sbit (CY  , 0xD0, 7);
Sbit (AC  , 0xD0, 6);
Sbit (F0  , 0xD0, 5);
Sbit (RS1 , 0xD0, 4);
Sbit (RS0 , 0xD0, 3);
Sbit (OV  , 0xD0, 2);
Sbit (UD  , 0xD0, 1);
Sbit (P   , 0xD0, 0);

Sfr (ACC , 0xE0);
Sfr (B , 0xF0);
Sfr (SP , 0x81);
Sfr (DPL , 0x82);
Sfr (DPH , 0x83);

Sfr (PCON , 0x87);
Sfr (CKCON , 0x8F);

/*------------------ TIMERS registers ----------------------*/
Sfr (TCON , 0x88);
Sbit (TF1 , 0x88, 7);
Sbit (TR1 , 0x88, 6);
Sbit (TF0 , 0x88, 5);
Sbit (TR0 , 0x88, 4);
Sbit (IE1 , 0x88, 3);
Sbit (IT1 , 0x88, 2);
Sbit (IE0 , 0x88, 1);
Sbit (IT0 , 0x88, 0);

Sfr (TMOD , 0x89);
```

```
Sfr  (T2CON , 0xC8);
Sbit (TF2   , 0xC8, 7);
Sbit (EXF2  , 0xC8, 6);
Sbit (RCLK  , 0xC8, 5);
Sbit (TCLK  , 0xC8, 4);
Sbit (EXEN2 , 0xC8, 3);
Sbit (TR2   , 0xC8, 2);
Sbit (C_T2  , 0xC8, 1);
Sbit (CP_RL2, 0xC8, 0);

Sfr (T2MOD , 0xC9);
Sfr (TL0 , 0x8A);
Sfr (TL1 , 0x8B);
Sfr (TL2 , 0xCC);
Sfr (TH0 , 0x8C);
Sfr (TH1 , 0x8D);
Sfr (TH2 , 0xCD);
Sfr (RCAP2L , 0xCA);
Sfr (RCAP2H , 0xCB);
Sfr (WDTRST , 0xA6);
Sfr (WDTPRG , 0xA7);


/*------------------- UART registers ------------------------*/
Sfr (SCON , 0x98);
Sbit (SM0  , 0x98, 7);
Sbit (FE   , 0x98, 7);
Sbit (SM1  , 0x98, 6);
Sbit (SM2  , 0x98, 5);
Sbit (REN  , 0x98, 4);
Sbit (TB8  , 0x98, 3);
Sbit (RB8  , 0x98, 2);
Sbit (TI   , 0x98, 1);
Sbit (RI   , 0x98, 0);

Sfr (SBUF , 0x99);
Sfr (SADEN , 0xB9);
Sfr (SADDR , 0xA9);

/*-------------------- ADC registers ----------------------*/
Sfr (ADCLK , 0xF2);
Sfr (ADCON , 0xF3);
#define MSK_ADCON_PSIDLE 0x40
#define MSK_ADCON_ADEN   0x20
#define MSK_ADCON_ADEOC  0x10
#define MSK_ADCON_ADSST  0x08
#define MSK_ADCON_SCH    0x07
Sfr (ADDL , 0xF4);
#define MSK_ADDL_UTILS 0x03
```

```
Sfr (ADDH , 0xF5);
Sfr (ADCF , 0xF6);

/*-------------------- FLASH EEPROM registers ------------*/
Sfr (FCON  , 0xD1);
#define MSK_FCON_FBUSY 0x01
#define MSK_FCON_FMOD  0x06
#define MSK_FCON_FPS   0x08
#define MSK_FCON_FPL   0xF0
Sfr (EECON  , 0xD2);
#define MSK_EECON_EEBUSY 0x01
#define MSK_EECON_EEE    0x02
#define MSK_EECON_EEPL   0xF0
Sfr (AUXR  , 0x8E);
#define MSK_AUXR_M0      0x20
Sfr (AUXR1  , 0xA2);
#define MSK_AUXR1_ENBOOT 0x20
/*-------------------- IT registers ----------------------*/
Sfr (IPL1 , 0xF8);
Sfr (IPH1 , 0xF7);
Sfr (IEN0  , 0xA8);
Sfr (IPL0 , 0xB8);
Sfr (IPH0 , 0xB7);
Sfr (IEN1  , 0xE8);

/*  IEN0  */
Sbit (EA   , 0xA8, 7);
Sbit (EC   , 0xA8, 6);
Sbit (ET2  , 0xA8, 5);
Sbit (ES   , 0xA8, 4);
Sbit (ET1  , 0xA8, 3);
Sbit (EX1  , 0xA8, 2);
Sbit (ET0  , 0xA8, 1);
Sbit (EX0  , 0xA8, 0);

/*  IEN1  */
Sbit (ETIM , 0xE8, 2);
Sbit (EADC , 0xE8, 1);
Sbit (ECAN , 0xE8, 0);

/*--------------------- PCA registers --------------------*/
Sfr (CCON , 0xD8);
Sbit(CF  , 0xD8, 7);
Sbit(CR  , 0xD8, 6);
Sbit(CCF4, 0xD8, 4);
Sbit(CCF3, 0xD8, 3);
Sbit(CCF2, 0xD8, 2);
Sbit(CCF1, 0xD8, 1);
Sbit(CCF0, 0xD8, 0);
```

```
Sfr (CMOD , 0xD9);
Sfr (CH , 0xF9);
Sfr (CL , 0xE9);
Sfr (CCAP0H  , 0xFA);
Sfr (CCAP0L  , 0xEA);
Sfr (CCAPM0  , 0xDA);
Sfr (CCAP1H  , 0xFB);
Sfr (CCAP1L  , 0xEB);
Sfr (CCAPM1  , 0xDB);
Sfr (CCAP2H  , 0xFC);
Sfr (CCAP2L  , 0xEC);
Sfr (CCAPM2  , 0xDC);
Sfr (CCAP3H  , 0xFD);
Sfr (CCAP3L  , 0xED);
Sfr (CCAPM3  , 0xDD);
Sfr (CCAP4H  , 0xFE);
Sfr (CCAP4L  , 0xEE);
Sfr (CCAPM4  , 0xDE);

/*------------------- CAN registers --------------------------*/
Sfr (CANGIT , 0x9B);
#define MSK_CANGIT_CANIT0x80
#define MSK_CANGIT_OVRTIM       0x20
#define MSK_CANGIT_OVRBUF0x10
#define MSK_CANGIT_SERG0x08
#define MSK_CANGIT_CERG0x04
#define MSK_CANGIT_FERG0x02
#define MSK_CANGIT_AERG0x01

Sfr (CANTEC , 0x9C);
Sfr (CANREC , 0x9D);
Sfr (CANTCON , 0xA1);
Sfr (CANMSG , 0xA3);
Sfr (CANTTCL , 0xA4);
Sfr (CANTTCH , 0xA5);
Sfr (CANGSTA , 0xAA);
#define MSK_CANGSTA_OVFG0x40
#define MSK_CANGSTA_TBSY0x10
#define MSK_CANGSTA_RBSY0x08
#define MSK_CANGSTA_ENFG0x04
#define MSK_CANGSTA_BOFF0x02
#define MSK_CANGSTA_ERRP0x01

Sfr (CANGCON , 0xAB);
#define MSK_CANGCON_ABRQ   0x80
#define MSK_CANGCON_OVRQ   0x40
#define MSK_CANGCON_TTC   0x20
#define MSK_CANGCON_SYNCTTC     0x10
#define TTC_EOF                 0x10
#define TTC_SOF        0x00
```

```
#define MSK_CANGCON_AUTBAUD     0x08
#define MSK_CANGCON_ENA  0x02
#define MSK_CANGCON_GRES 0x01


Sfr (CANTIML , 0xAC);
Sfr (CANTIMH , 0xAD);
Sfr (CANSTMPL , 0xAE);
Sfr (CANSTMPH , 0xAF);
Sfr (CANPAGE , 0xB1);
Sfr (CANSTCH , 0xB2);
#define MSK_CANSTCH_DLCW  0x80
#define MSK_CANSTCH_TxOk  0x40
#define MSK_CANSTCH_RxOk  0x20
#define MSK_CANSTCH_BERR  0x10
#define MSK_CANSTCH_SERR  0x08
#define MSK_CANSTCH_CERR  0x04
#define MSK_CANSTCH_FERR  0x02
#define MSK_CANSTCH_AERR  0x01

Sfr (CANCONCH , 0xB3);
#define MSK_CANCONCH_IDE  0x10
#define MSK_CANCONCH_DLC  0x0F
#define MSK_CANCONCH_CONF 0xC0
#define DLC_MAX    8
#define CH_DISABLE 0x00
#define CH_RxENA   0x80
#define CH_TxENA   0x40
#define CH_RxBENA  0xC0

Sfr (CANBT1 , 0xB4);
#define CAN_PRESCALER_MIN  0
#define CAN_PRESCALER_MAX  63

Sfr (CANBT2 , 0xB5);
#define MSK_CANBT2_SJW  0x60
#define MSK_CANBT2_PRS  0x0E
#define CAN_SJW_MIN  0
#define CAN_SJW_MAX  3
#define CAN_PRS_MIN  0
#define CAN_PRS_MAX  7

Sfr (CANBT3 , 0xB6);
#define MSK_CANBT3_PHS2 0x70
#define MSK_CANBT3_PHS1 0x0E
#define CAN_PHS2_MIN 0
#define CAN_PHS2_MAX 7
#define CAN_PHS1_MIN 0
#define CAN_PHS1_MAX 7
```

```
Sfr (CANSIT1 , 0xBA);
Sfr (CANSIT2 , 0xBB);
Sfr (CANIDT1 , 0xBC);
Sfr (CANIDT2 , 0xBD);
Sfr (CANIDT3 , 0xBE);
Sfr (CANIDT4 , 0xBF);
#define MSK_CANIDT4_RTRTAG 0x04

Sfr (CANGIE , 0xC1);
#define MSK_CANGIE_ENRX    0x20
#define MSK_CANGIE_ENTX    0x10
#define MSK_CANGIE_ENERCH  0x08
#define MSK_CANGIE_ENBUF   0x04
#define MSK_CANGIE_ENERG   0x02

Sfr (CANIE1 , 0xC2);
Sfr (CANIE2 , 0xC3);
Sfr (CANIDM1 , 0xC4);
Sfr (CANIDM2 , 0xC5);
Sfr (CANIDM3 , 0xC6);
Sfr (CANIDM4 , 0xC7);
#define MSK_CANIDM4_RTRMSK 0x04
#define MSK_CANIDM4_IDEMSK 0x01

Sfr (CANEN1 , 0xCE);
Sfr (CANEN2 , 0xCF);

#endif
```

3. Assembler 51 Examples

3.1 8 bits Adc

```
$INCLUDE   (t89c51cc01.INC)
value_converted DATA 10H;     /* converted value */
value_AN6 DATA 11H;           /* converted AN6 value */
value_AN7 DATA 12H;           /* converted AN7 value */
end_of_convertion BIT 20H;    /* software flag */

org 000h
ljmp begin

org 43h
ljmp adc_it

;/**
; * FUNCTION_PURPOSE:this function setup Adc with channel 6 and 7 and start
; * 8bits convertion.
; * FUNCTION_INPUTS:void
; * FUNCTION_OUTPUTS:void
; */
org 0100h
begin:

/* configure channel P1.6(AN6) and P1.7(AN7) for ADC */
MOV ADCF,#0C0h;

/* init prescaler for adc clock */
/* Fadc = Fperiph/(2*(32-PRS)), PRS -> ADCLK[4:0] */
MOV ADCLK,#06h;                  /* Fosc = 16 MHz, Fadsc = 153.8khz */

MOV ADCON,#20h;                  /* Enable the ADC */


SETB EA;                         /* enable interrupts */
SETB EADC;                       /* enable ADC interrupt */
loop:

   ANL ADCON,#~07h;              /* Clear the channel field ADCON[2:0] */
   ORL ADCON, #06h;              /* Select channel 6 */
   ANL ADCON,#~40h;              /* standard mode */
   ORL ADCON, #08h;              /* Start conversion */

   JNB end_of_convertion,$;      /* wait end of convertion */
   CLR end_of_convertion;        /* clear software flag */
   MOV value_AN6,value_converted;/* save converted value */


   ANL ADCON,#~07h;              /* Clear the channel field ADCON[2:0] */
```

```
   ORL ADCON, #07h;              /* Select channel 7 */
   ANL ADCON,#~40h;              /* standard mode */
   ORL ADCON, #08h;              /* Start conversion */

   JNB end_of_convertion,$;      /* wait end of convertion */
   CLR end_of_convertion;        /* clear software flag */
   MOV value_AN7,value_converted;/* save converted value */

JMP loop

;/**
; * FUNCTION_PURPOSE:Adc interrupt, save ADDH into an unsigned char
; * FUNCTION_INPUTS:void
; * FUNCTION_OUTPUTS:void
; */
adc_it:
ANL ADCON,#~10h;                 /* Clear the End of conversion flag */
MOV value_converted,ADDH;        /* save value */
SETB end_of_convertion;          /* set flag */
RETI

end
```

3.2 10 bits ADC

```
$INCLUDE   (t89c51cc01.INC)
msb_value_converted DATA 10H;    /* converted msb value */
lsb_value_converted DATA 11H;    /* converted lsb value */
msb_value_AN6 DATA 12H;          /* converted msb AN6 value */
lsb_value_AN6 DATA 13H;          /* converted lsb AN6 value */
msb_value_AN7 DATA 14H;          /* converted msb AN7 value */
lsb_value_AN7 DATA 15H;          /* converted lsb AN7 value */
end_of_convertion BIT 20H;       /* software flag */
CLR end_of_convertion

org 000h
ljmp begin

org 43h
ljmp adc_it

;/**
; * FUNCTION_PURPOSE:this function setup Adc with channel 6 and 7 and start
; * 8bits convertion.
; * FUNCTION_INPUTS:void
; * FUNCTION_OUTPUTS:void
; */
org 0100h
begin:

/* configure channel P1.6(AN6) and P1.7(AN7) for ADC */
MOV ADCF,#0C0h;

;/* init prescaler for adc clock */
;/* Fadc = Fperiph/(2*(32-PRS)), PRS -> ADCLK[4:0] */
MOV ADCLK,#06h;                  /* Fosc = 16 MHz, Fadsc = 153.8khz */

MOV ADCON,#20h;                  /* Enable the ADC */


SETB EA;                         /* enable interrupts */
SETB EADC;                       /* enable ADC interrupt */
loop:

   ANL ADCON,#~07h;              /* Clear the channel field ADCON[2:0] */
   ORL ADCON, #06h;              /* Select channel 6 */
   ANL ADCON,#~40h;              /* standard mode */
   ORL ADCON, #08h;              /* Start conversion */

   JNB end_of_convertion,$;      /* wait end of convertion */
   CLR end_of_convertion;        /* clear software flag */
   MOV msb_value_AN6,msb_value_converted;/* save converted msb value */
   MOV lsb_value_AN6,lsb_value_converted;/* save converted lsb value */
```

```
   ANL ADCON,#~07h;              /* Clear the channel field ADCON[2:0] */
   ORL ADCON, #07h;              /* Select channel 7 */
   ANL ADCON,#~40h;              /* standard mode */
   ORL ADCON, #08h;              /* Start conversion */

   JNB end_of_convertion,$;      /* wait end of convertion */
   CLR end_of_convertion;        /* clear software flag */
   MOV msb_value_AN7,msb_value_converted;/* save converted msb value */
   MOV lsb_value_AN7,lsb_value_converted;/* save converted lsb value */

JMP loop

;/**
; * FUNCTION_PURPOSE:Adc interrupt, save ADDH and ADDL into an unsigned int
; * FUNCTION_INPUTS:void
; * FUNCTION_OUTPUTS:void
; */
adc_it:
ANL ADCON,#~10h;                 /* Clear the End of conversion flag */
;/* copy ADDH[7:6] into msb_value_converted[1:0] */
MOV A,ADDH
SWAP A
RR A
RR A
ANL A,#~0FCh
MOV msb_value_converted,A
;/* copy ADDH[5:0] into lsb_value_coverted[7:2]
MOV A,ADDH
RL A
RL A
ANL A,#~03h
MOV lsb_value_converted,A
;/* copy ADDL[1:0] into lsb_value_coverted[1:0]
MOV A,ADDL
ANL A,#~0FCh
ORL lsb_value_converted,A

SETB end_of_convertion;          /* set flag */
RETI

end
```

3.3 SFR Register Definition

```
;*INC*************************************************************************
**
; NAME: 89C51CC01.inc
;-------------------------------------------------------------------------------
-
; PURPOSE: for Keil
;*************************************************************************
**

;----------------------------------------
; Include file for 8051 SFR Definitions
;----------------------------------------

;  BYTE Register
P0      DATA    80H
P1      DATA    90H
P2      DATA    0A0H

P3      DATA    0B0H
RD      BIT     0B7H
WR      BIT     0B6H
T1      BIT     0B5H
T0      BIT     0B4H
INT1    BIT     0B3H
INT0    BIT     0B2H
TXD     BIT     0B1H
RXD     BIT     0B0H

P4      DATA    0C0H

PSW     DATA    0D0H
CY      BIT     0D7H
AC      BIT     0D6H
F0      BIT     0D5H
RS1     BIT     0D4H
RS0     BIT     0D3H
OV      BIT     0D2H
P       BIT     0D0H

ACC     DATA    0E0H
B       DATA    0F0H
SP      DATA    81H
DPL     DATA    82H
DPH     DATA    83H
PCON    DATA    87H
CKCON   DATA    8FH

;------------------ TIMERS registers ----------------------
TCON    DATA    88H
```

```
TF1     BIT     8FH
TR1     BIT     8EH
TF0     BIT     8DH
TR0     BIT     8CH
IE1     BIT     8BH
IT1     BIT     8AH
IE0     BIT     89H
IT0     BIT     88H

TMOD    DATA    89H

T2CON   DATA    0C8H
TF2     BIT 0CFH
EXF2    BIT 0CEH
RCLK    BIT 0CDH
TCLK    BIT 0CCH
EXEN2   BIT 0CBH
TR2     BIT 0CAH
C_T2    BIT 0C9H
CP_RL2  BIT 0C8H

T2MOD DATA0C9H
TL0     DATA    8AH
TL1     DATA    8BH
TL2 DATA0CCH
TH0     DATA    8CH
TH1     DATA    8DH
TH2 DATA0CDH
RCAP2L DATA0CAH
RCAP2H DATA0CBH
WDTRST  DATA0A6H
WDTPRG  DATA0A7H

;------------------- UART registers ------------------------
SCON    DATA    98H
SM0  BIT 9FH
FE   BIT9FH
SM1  BIT9EH
SM2  BIT9DH
REN  BIT9CH
TB8  BIT9BH
RB8  BIT9AH
TI   BIT99H
RI   BIT98H

SBUF    DATA    99H
SADEN DATA0B9H
SADDR DATA0A9H

;-------------------- ADC registers ----------------------
```

```
ADCLK DATA0F2H
ADCON DATA0F3H
ADDL DATA0F4H
ADDH DATA0F5H
ADCF DATA0F6H

;-------------------- FLASH EEPROM registers ------------
FPGACON  DATA0F1H
FCON  DATA0D1H
EECON  DATA0D2H
AUXR  DATA8EH
AUXR1  DATA0A2H

;-------------------- IT registers ----------------------
IPL1 DATA0F8H
IPH1 DATA0F7H
IEN0  DATA0A8H
IPL0 DATA0B8H
IPH0 DATA0B7H
IEN1  DATA0E8H

;  IEN0
EA   BIT 0AFH
EC   BIT 0AEH
ET2  BIT 0ADH
ES   BIT 0ACH
ET1  BIT 0ABH
EX1  BIT 0AAH
ET0  BIT 0A9H
EX0  BIT 0A8H

;  IEN1
ETIM BIT 0EAH
EADC BIT  0E9H
ECAN BIT  0E8H

;---------------------- PCA registers --------------------
CCON DATA0D8H
CF  BIT 0DFH
CR  BIT 0DEH
CCF4BIT0D4H
CCF3BIT0D3H
CCF2BIT0D2H
CCF1BIT0D1H
CCF0BIT0D0H

CMOD DATA0D9H
CH DATA0F9H
CL DATA0E9H
CCAP0H  DATA0FAH
```

```
CCAP0L  DATA0EAH
CCAPM0  DATA0DAH
CCAP1H  DATA0FBH
CCAP1L  DATA0EBH
CCAPM1  DATA0DBH
CCAP2H  DATA0FCH
CCAP2L  DATA0ECH
CCAPM2  DATA0DCH
CCAP3H  DATA0FDH
CCAP3L  DATA0EDH
CCAPM3  DATA0DDH
CCAP4H  DATA0FEH
CCAP4L  DATA0EEH
CCAPM4  DATA0DEH

;------------------- CAN registers --------------------------
CANGIT DATA 09BH
CANTEC DATA 09CH
CANREC DATA 09DH
CANTCON DATA 0A1H
CANMSG DATA 0A3H
CANTTCL DATA 0A4H
CANTTCH DATA 0A5H
CANGSTA DATA 0AAH
CANGCON DATA 0ABH
CANTIML DATA 0ACH
CANTIMH DATA 0ADH
CANSTMPL DATA 0AEH
CANSTMPH DATA 0AFH
CANPAGE DATA 0B1H
CANSTCH DATA 0B2H
CANCONCH DATA 0B3H
CANBT1 DATA 0B4H
CANBT2 DATA 0B5H
CANBT3 DATA 0B6H
CANSIT1 DATA 0BAH
CANSIT2 DATA 0BBH
CANIDT1 DATA 0BCH
CANIDT2 DATA 0BDH
CANIDT3 DATA 0BEH
CANIDT4 DATA 0BFH
CANGIE DATA 0C1H
CANIE1 DATA 0C2H
CANIE2 DATA 0C3H
CANIDM1 DATA 0C4H
CANIDM2 DATA 0C5H
CANIDM3 DATA 0C6H
CANIDM4 DATA 0C7H
CANEN1  DATA 0CEH
CANEN2  DATA 0CFH
```

Atmel Corporation

2325 Orchard Parkway
San Jose, CA 95131
Tel: 1(408) 441-0311
Fax: 1(408) 487-2600

Regional Headquarters

Europe
Atmel Sarl
Route des Arsenaux 41
Case Postale 80
CH-1705 Fribourg
Switzerland
Tel: (41) 26-426-5555
Fax: (41) 26-426-5500

Asia
Room 1219
Chinachem Golden Plaza
77 Mody Road Tsimshatsui
East Kowloon
Hong Kong
Tel: (852) 2721-9778
Fax: (852) 2722-1369

Japan
9F, Tonetsu Shinkawa Bldg.
1-24-8 Shinkawa
Chuo-ku, Tokyo 104-0033
Japan
Tel: (81) 3-3523-3551
Fax: (81) 3-3523-7581

Atmel Operations

Memory
2325 Orchard Parkway
San Jose, CA 95131
Tel: 1(408) 441-0311
Fax: 1(408) 436-4314

Microcontrollers
2325 Orchard Parkway
San Jose, CA 95131
Tel: 1(408) 441-0311
Fax: 1(408) 436-4314

La Chantrerie
BP 70602
44306 Nantes Cedex 3, France
Tel: (33) 2-40-18-18-18
Fax: (33) 2-40-18-19-60

ASIC/ASSP/Smart Cards
Zone Industrielle
13106 Rousset Cedex, France
Tel: (33) 4-42-53-60-00
Fax: (33) 4-42-53-60-01

1150 East Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906
Tel: 1(719) 576-3300
Fax: 1(719) 540-1759

Scottish Enterprise Technology Park
Maxwell Building
East Kilbride G75 0QR, Scotland
Tel: (44) 1355-803-000
Fax: (44) 1355-242-743

RF/Automotive
Theresienstrasse 2
Postfach 3535
74025 Heilbronn, Germany
Tel: (49) 71-31-67-0
Fax: (49) 71-31-67-2340

1150 East Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906
Tel: 1(719) 576-3300
Fax: 1(719) 540-1759

Biometrics/Imaging/Hi-Rel MPU/
High Speed Converters/RF Datacom
Avenue de Rochepleine
BP 123
38521 Saint-Egreve Cedex, France
Tel: (33) 4-76-58-30-00
Fax: (33) 4-76-58-34-80

e-mail
literature@atmel.com

Web Site
http://www.atmel.com

4361C–80C51–11/04 /xM