การทดลองที่ 3

การใช้งานโมดูลแปลงสัญญาณอนาลอกเป็นดิจิตอล (ADC)

(Implementation of ADCs.)

วัตถุประสงค์

1. เพื่อให้สามารถใช้งานโมดูลการแปลงสัญญาณอนาลอกแบบต่างของ PSoC ได้
2. สามารถทำการเชื่อมต่อโมดูลต่างๆภายใน PSoC เข้าด้วยกันได้
3. เพื่อให้สามารถสร้าง Header file ได้

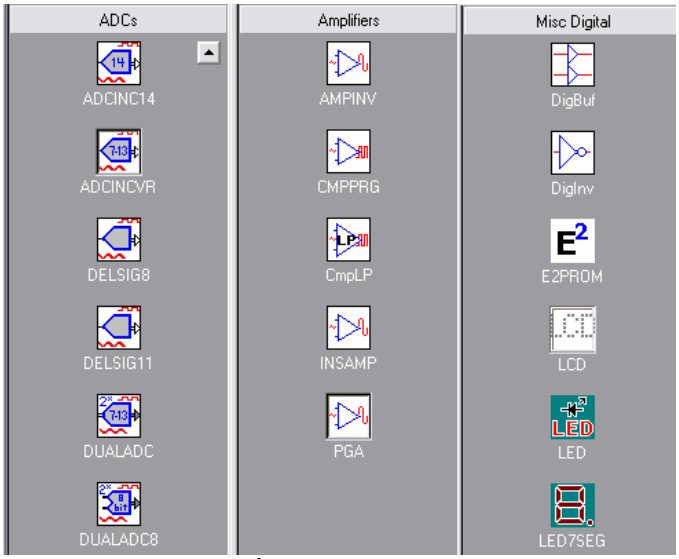
บทนำ

การทดลองนี้ ได้แสดงวิธีการใช้งานโมดูลการแปลงสัญญาณอนาลอกแบบ Incremental แบบ Successive Approximation Register (SAR) และแบบเดลต้าซิกมา (Delta Sigma) และนำผลการแปลงแสดงออกทางแอลซีดีแบบดอทแมทริก

การทดลองที่ 3.1 แสดงค่าการแปลงแรงดันด้วยโมดูลการแปลงสัญญาณอนาลอกแบบ Incremental แล้วนำผลแสดงทางแอลซีดี

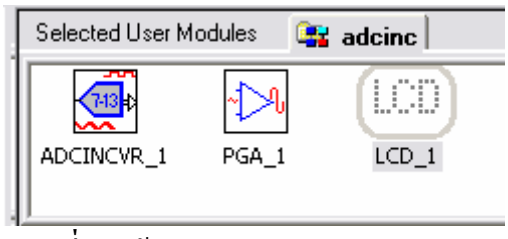
การทดลอง เนื่องจากขั้นตอนต่างๆได้กล่าวไว้โดยละเอียดแล้วในการทดลองที่ 1 และ 2 ดังนั้นในที่นี้จะกล่าวถึงเพียงย่อๆ หากนักศึกษายังไม่เข้าใจให้ย้อนกลับไปดูการทดลองที่ 1 สำหรับในการทดลองนี้จะได้อธิบายอย่างละเอียดเกี่ยวกับขั้นตอนการเรียกใช้โมดูลและการเชื่อมต่อโมดูลเข้าด้วยกัน

1. ให้รันโปรแกรม PSoC Designer แล้วทำการสร้างโปรเจคใหม่ โดยใช้ไมโครคอนโทรลเลอร์เบอร์ CY8C29466-24PXI ส่วน Main file ให้เลือกเป็นภาษา C



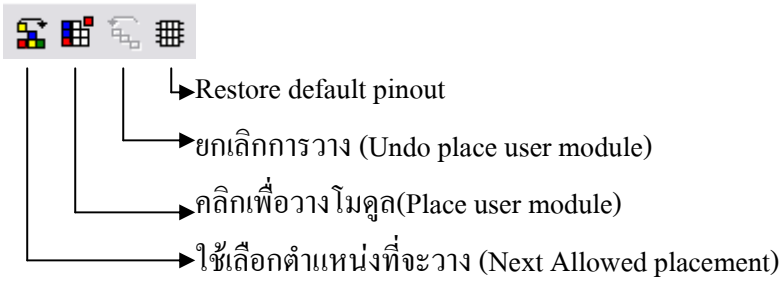
รูปที่ 1 โมดูลที่เลือกใช้

2. เมื่อเข้าสู่ PSoC designer ในหน้าต่าง User Module Selection View ให้เลือกใช้โมดูล ADCINCVR โมดูล PGA และโมดูล LCD ซึ่งอยู่ในกลุ่ม ADCs กลุ่มAmplifiers และกลุ่ม Misc Digital ตามลำดับ เมื่อเลือกแล้ว ในหน้าต่าง Selection User Modules จะปรากฏโมดูลทั้ง 3 ตามรูปที่ 2



รูปที่ 2 หน้าต่าง Selection User Modules

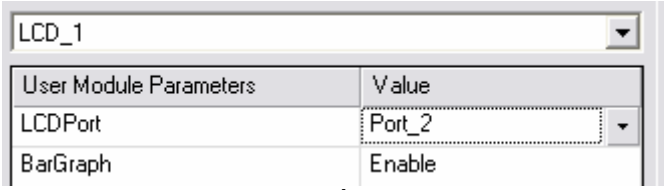
3. หลังจากเลือกโมดูลต่างๆแล้ว ให้คลิก  (Interconnect View) เพื่อทำการวางอุปกรณ์ ก่อนอื่นควรรู้จัก ไอคอนของคำสั่งเกี่ยวกับการวางโมดูลดังนี้



วิธีการวาง ให้คลิกเลือกโมดูลที่ต้องการวางใน Selected User Modules แล้วคลิกที่ไอคอนวางโมดูลหรือถ้าต้องการลบโมดูลก็ คลิกที่ไอคอนลบโมดูล หรือใช้ทางลัดโดยการวางเมาส์ที่ไอคอนโมดูลแล้วคลิกขวาเลือกคำสั่ง Place เพื่อวางโมดูล หรือคำสั่ง Delete เพื่อลบโมดูล

4. การกำหนดค่าพารามิเตอร์ให้กับโมดูล

การกำหนดค่าให้โมดูลต่างๆ ให้คลิกที่ไอคอนของโมดูล จะปรากฏหน้าต่างพารามิเตอร์ของโมดูลนั้นขึ้นมาให้กำหนดค่าต่างๆตามรูปที่ 3



รูปที่ 3

ADCINCVR_1

User Module Parameters	Value
Input	ACB00
ClockPhase	Norm
Clock	VC1
ADCResolution	12 Bit
CalcTime	10
DataFormat	Unsigned

PGA_1

User Module Parameters	Value
Gain	1.000
Input	AnalogColumn_InputMUX_0
Reference	VSS
AnalogBus	Disable

รูปที่ 3 (ต่อ)

ส่วนค่า Global Resource ให้กำหนดดังนี้

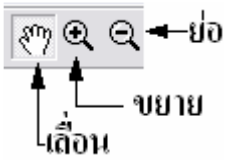
Global Resources	Value
Power Setting [Vcc / SysClk freq	5.0V / 24MHz
CPU_Clock	SysClk/2
32K_Select	Internal
PLL_Mode	Disable
Sleep_Timer	512_Hz
VC1= SysClk/N	6
VC2= VC1/N	1
VC3 Source	SysClk/1
VC3 Divider	1
SysClk Source	Internal
SysClk*2 Disable	No
Analog Power	SC On/Ref High
Ref Mux	(Vdd/2)+/-(Vdd/2)
AGndBypass	Disable
Op-Amp Bias	Low
A_Buff_Power	Low
SwitchModePump	OFF
Trip Voltage [LVD (SMP)]	4.81V (5.00V)
LVDThrottleBack	Disable
Watchdog Enable	Disable

รูปที่ 4

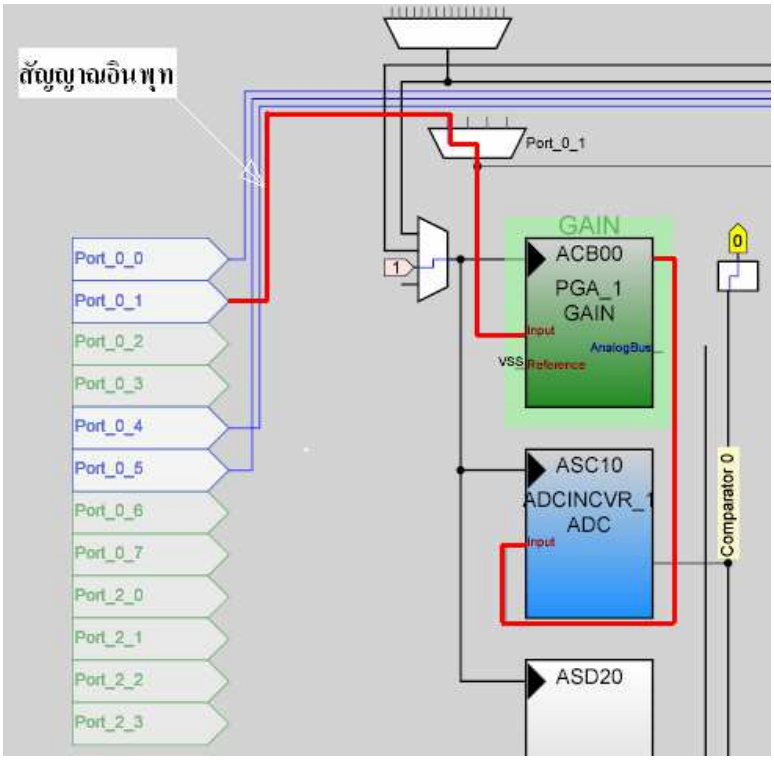
ส่วนขา Port_0_1 ให้เลือกเป็น AnalogInput

Name	Port	Select	Drive	Interru
Port_0_0	P0[0]	StdCPU	High Z Analog	Disabl
Port_0_1	P0[1]	AnalogInput	High Z Analog	Disabl
Port_0_2	P0[2]	StdCPU	High Z Analog	Disabl

จากการวางโมดูลและการกำหนดค่าพารามิเตอร์ของโมดูล เราสามารถดูไดอะแกรมของโมดูลต่างๆที่เชื่อมต่อกันได้จากหน้าต่างที่แสดงรายละเอียดการวาง โดยใช้ไอคอนคำสั่ง เลื่อน ย่อ ขยายภาพ เพื่อให้เห็นรายละเอียดต่างๆให้ชัดเจนขึ้น ตามรูปที่ 5



รูปที่ 5 ไอคอนคำสั่ง เลื่อน ย่อ ขยายภาพ

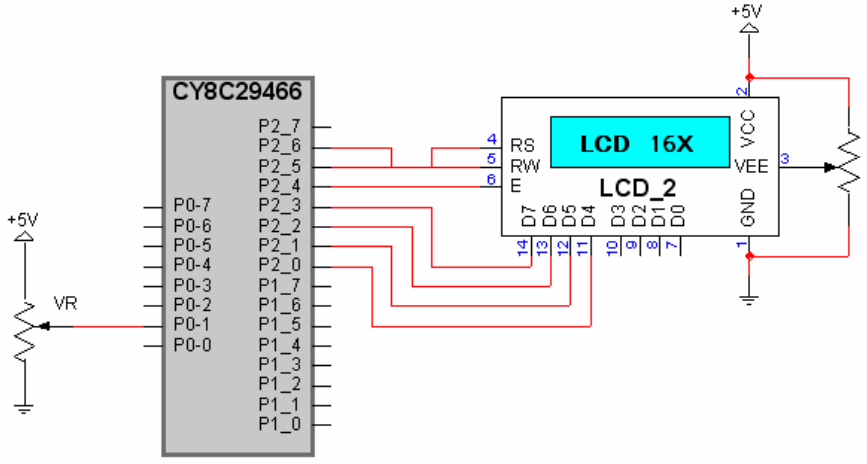


รูปที่ 6 แสดงการเชื่อมต่อสัญญาณอินพุทเข้า ADC โดยผ่านโมดูล PGA

5. เมื่อเสร็จแล้วให้กด (Generate Application) เพื่อสร้างซอร์ไฟล์ต่างๆ
6. ให้คลิกที่ปุ่ม (Application Editor) เพื่อเปิดหน้าต่าง Application เพื่อเขียน Source file ให้แก้ไขโปรแกรม main.c เป็น ดังนี้

```
//-------------------------------------------------------------------------
// C main line
//-------------------------------------------------------------------------

#include <m8c.h>                // part specific constants and macros
#include "PSoCAPI.h"            // PSoC API definitions for all User Modules
#include "stdlib.h"             // Add this header to use the ftoa function

int iData;                      // Variable that stores the ADC result
float fVolts;                   // Variable that stores the converted voltage value
float fScaleFactor;             // Variable that stores the volts/count scale factor
char *pResult;                  // Pointer used to store the result returned by ftoa function
int iStatus;                    // Status variable for the ftoa function
void main()
{
   PGA_1_Start(PGA_1_HIGHPOWER);            // Start PGA with Highpower
     LCD_1_Start();                         // Start LCD
     LCD_1_Position(0,0);                   // Set LCD position to row 0 column 0
     LCD_1_PrCString("MEASURED VOLTAGE");   // Print string "MEASURED VOLTAGE" on LCD
     M8C_EnableGInt;                        // Enable Global Interrupts
     ADCINCVR_1_Start(ADCINCVR_1_HIGHPOWER);      // Start ADC by powering SC block at High Power
     ADCINCVR_1_GetSamples(0);                    // Have ADC run continuously
     fScaleFactor = (float)5/(float)4096;         // Calculate Scale Factor.
     for(;;)                                      // Infinite loop
     {
               while(ADCINCVR_1_fIsDataAvailable() == 0);          // Loop until value ready
               iData=ADCINCVR_1_iGetData();     // Read ADC result
               ADCINCVR_1_ClearFlag();          // Clear ADC flag
               fVolts = fScaleFactor*(float)iData;  // Calculate voltage using ADC result and scale factor
               pResult = ftoa(fVolts,&iStatus );    // Convernt Float value of voltage into ASCII string
               LCD_1_Position(1,0);             // Set LCD position to row 1 column 0
               LCD_1_PrString(pResult);         // Print voltage value on LCD
               LCD_1_PrCString(" V");           // Print string " V" on LCD after voltage value
     }
}
```

7. ให้แปลโปรแกรมหากไม่มีข้อผิดพลาด จะได้ไฟล์ Hex ที่มีชื่อเหมือนกับชื่อของโปรเจค อยู่ในโฟลเดอร์ Output สำหรับใช้ดาว์นโหลดลงชิพไมโครคอนโทรลเลอร
8. ทำการติดตั้ง LCD เข้ากับบอร์ด และต่อสายจาก Port_0_1 เข้ากับขั้วต่อ VR ในบอร์ดทดลอง



รูปที่ 7 แสดงการเชื่อมต่อแรงดันดีซีจาก ความต้านทานปรับค่าได้ เข้า PSoC

9. ให้ต่อสาย USB แล้วดาว์นโหลดโปรแกรมลงชิพ **<u>ขั้นตอนนี้ขอให้ทำด้วยความระมัดระวังมิฉนั้นอาจเกิดความเสียหายคอมพิวเตอร์และบอร์ดทดลองได้</u>**
10. เมื่อดาว์นโหลดเสร็จ ให้ทำการทดสอบการทำงาน โดยทำการปรับ VR แล้วดูการเปลี่ยนแปลงที่ LCD
11. ใช้ดิจิตอลมัลติมีเตอร์วัดค่าแรงดันที่ VR และปรับค่า VR ให้ได้แรงดันตามตารางที่ 1 อ่านค่าแรงดันที่ได้จาก LCD บันทึกลงในตารางที่ 1 ในคอลัมน์ Vinc

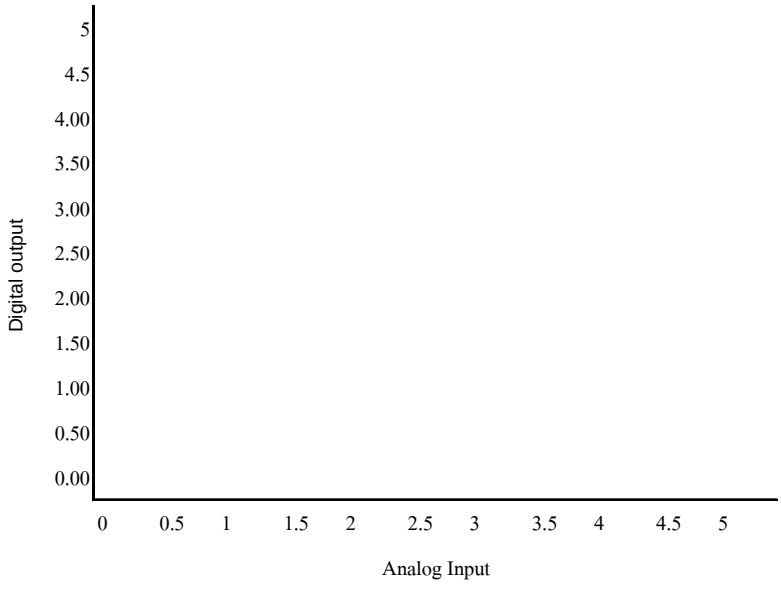
ตารางที่ 1

VR (โวลท์)	Vinc	Vsar	Vdel	Einc	Esar	Edel
0.50						
1.00						
1.50						
2.00						
2.50						
3.00						
3.50						
4.00						
4.50						

ให้คำนวณหาค่าความผิดพลาด

$$Einc = VR - Vinc$$

และพร็อตกราฟ Vinc (Digital output) เทียบกับค่า VR (Analog input) ลงในรูปที่ 8



รูปที่ 8

การทดลองที่ 3.2 แสดงค่าการแปลงแรงดันด้วยโมดูลการแปลงสัญญาณอนาลอกแบบ Successive Approximation Register (SAR) แล้วนำผลแสดงทางแอลซีดี

การทดลองนี้ใช้ โมดูล SAR6 โมดูล PGA และโมดูล LCD โดยกำหนดพารามิเตอร์ ตามรูปที่ 9

Global Resources	Value
Power Setting [Vcc / SysClk freq	5.0V / 24MHz
CPU_Clock	SysClk/8
32K_Select	Internal
PLL_Mode	Disable
Sleep_Timer	512_Hz
VC1= SysClk/N	16
VC2= VC1/N	1
VC3 Source	SysClk/1
VC3 Divider	1
SysClk Source	Internal
SysClk*2 Disable	No
Analog Power	SC On/Ref Low
Ref Mux	(Vdd/2)+/-(Vdd/2)
AGndBypass	Disable
Op-Amp Bias	Low
A_Buff_Power	Low
SwitchModePump	OFF
Trip Voltage [LVD (SMP)]	4.81V (5.00V)
LVDThrottleBack	Disable
Watchdog Enable	Disable

SAR6_1

User Module Parameters	Value
SignalSource	ACB00

PGA_1

User Module Parameters	Value
Gain	1.000
Input	AnalogColumn_Input
Reference	VSS
AnalogBus	Disable

รูปที่ 9 การกำหนดพารามิเตอร์ของ Global resources SAR6 และ PGA

สำหรับพารามิเตอร์ของ LCD และขา Port_0_1 ให้เหมือนการทดลองที่ 3.1 และทดลองเหมือนข้อ 11 ของการทดลองที่ 3.1 ให้บันทึกค่าและพร็อตกราฟ ลงใน ตารางที่ 1 และรูปที่ 8 ตามลำดับ โปรแกรมทดสอบใช้ดังนี้

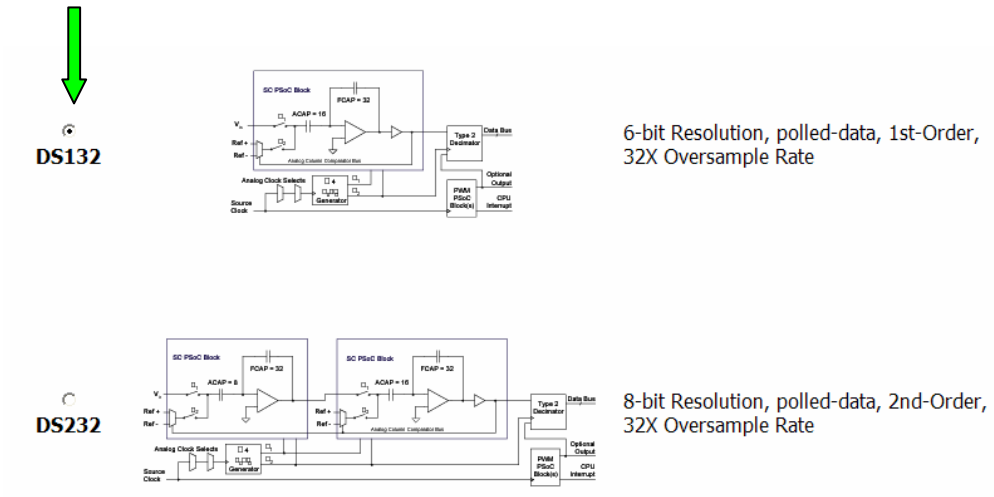
```
//-------------------------------------------------------------------------
// C main line
//-------------------------------------------------------------------------

#include <m8c.h>                    // part specific constants and macros
#include "PSoCAPI.h"                // PSoC API definitions for all User Modules
#include "stdlib.h"                 // Add this header to use the ftoa function

char iData;                         // Variable that stores the ADC result
float fVolts;                       // Variable that stores the converted voltage value
float fScaleFactor;                 // Variable that stores the volts/count scale factor
char *pResult;                      // Pointer used to store the result returned by ftoa function
int iStatus;                        // Status variable for the ftoa function

void main()
```

```
{
   PGA_1_Start(PGA_1_HIGHPOWER);                // Start PGA with Highpower
        LCD_1_Start();                          // Start LCD
        LCD_1_Position(0,0);                    // Set LCD position to row 0 column 0
        LCD_1_PrCString("MEASURED VOLTAGE");    // Print string "MEASURED VOLTAGE" on LCD
        M8C_EnableGInt;                         // Enable Global Interrupts

        SAR6_1_Start(SAR6_1_HIGHPOWER);         // Start ADC by powering SC block at High Power
        fScaleFactor = (float)5/(float)62;      // Calculate Scale Factor.
        for(;;)                                 // Infinite loop
        {
        iData = SAR6_1_cGetSample();            // Read ADC result
                if(iData>=32)                   //Convert signed 6bit value to unsiged 8bit.
                {
                        fVolts = 32.0-(256.0-(float)iData);
                }
                else
                {
                        fVolts = 32.0+(float)iData;
                }
                fVolts = fScaleFactor*(float)fVolts;  // Calculate voltage using ADC result and scale factor

                pResult = ftoa(fVolts,&iStatus );     // Convernt Float value of voltage into ASCII string

                LCD_1_Position(1,0);            // Set LCD position to row 1 column 0
                LCD_1_PrString(pResult);        // Print voltage value on LCD
                LCD_1_PrCString(" V      ");    // Print string " V" on LCD after voltage value
        }
}
```

การทดลองที่ 3.3 แสดงค่าการแปลงแรงดันด้วยโมดูลการแปลงสัญญาณอนาลอกแบบเดลต้าซิกมา แล้วนำผลแสดงทางแอลซีดี

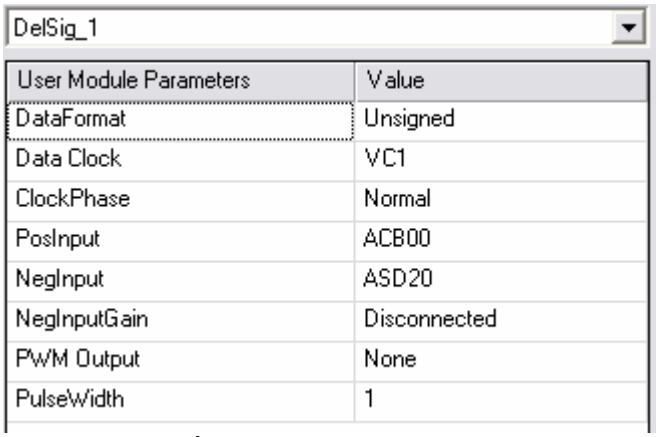
การทดลองนี้ใช้ โมดูล DelSig โมดูล PGA และโมดูล LCD โดยพารามิเตอร์ของ PGA LCD Global Resource และ ขาสัญญาณ กำหนดเหมือนกับ การทดลองที่ 3.2 ส่วน โมดูล ADC แบบ DelSig ให้ทำดังนี้

1. ให้เลือก DelSig จากกลุ่ม ADCs โดยดับเบิลคลิกที่ไอคอน DelSig จะปรากฏหน้าต่าง Multi UserModule Selection ซึ่งเป็นหน้าต่างสำหรับเลือกชนิดของ ADC ให้เลือก DS132 เป็น ADC ขนาด 6 บิต 1st-Order ตามรูป



รูปที่ 10 การเลือก ADC DelSig แบบ DS132

สำหรับ พารามิเตอร์ของ DelSig ให้กำหนดดังนี้



DelSig_1

User Module Parameters	Value
DataFormat	Unsigned
Data Clock	VC1
ClockPhase	Normal
PosInput	ACB00
NegInput	ASD20
NegInputGain	Disconnected
PWM Output	None
PulseWidth	1

รูปที่ 11 พารามิเตอร์ของ DelSig

ให้ทำการทดลองเหมือนข้อ 11 ของการทดลองที่ 3.1 และจดบันทึกค่าพร้อมเขียนกราฟ ลงใน ตารางที่ 1 และรูปที่ 8 ตามลำดับ โปรแกรมทดสอบใช้ดังนี้

```
//-------------------------------------------------------------------------
// C main line
//-------------------------------------------------------------------------

#include <m8c.h>                    // part specific constants and macros
#include "PSoCAPI.h"                // PSoC API definitions for all User Modules
#include "stdlib.h"                 // Add this header to use the ftoa function

int iData;                          // Variable that stores the ADC result
float fVolts;                       // Variable that stores the converted voltage value
float fScaleFactor;                 // Variable that stores the volts/count scale factor
char *pResult;                      // Pointer used to store the result returned by ftoa function
int iStatus;                        // Status variable for the ftoa function
void main()
{
        PGA_1_Start(PGA_1_HIGHPOWER);                   // Start PGA with Highpower
        LCD_1_Start();                                  // Start LCD
        LCD_1_Position(0,0);                            // Set LCD position to row 0 column 0
        LCD_1_PrCString("MEASURED VOLTAGE");            // Print string "MEASURED VOLTAGE" on LCD
        M8C_EnableGInt;                                 // Enable Global Interrupts
        DelSig_1_Start(DelSig_1_HIGHPOWER);             // Start ADC by powering SC block at High Power
        DelSig_1_StartAD();                             // Have ADC run continuously
        fScaleFactor = (float)5/(float)64;              // Calculate Scale Factor.
        for(;;)                                         // Infinite loop
        {
                while(DelSig_1_fIsDataAvailable() == 0);     // Loop until value ready
                iData=DelSig_1_bGetData();              // Read ADC result
                DelSig_1_ClearFlag();                   // Clear ADC flag
                fVolts = fScaleFactor*(float)iData;     // Calculate voltage using ADC result and scale factor
                pResult = ftoa(fVolts,&iStatus );       // Convernt Float value of voltage into ASCII string
                LCD_1_Position(1,0);                    // Set LCD position to row 1 column 0
                LCD_1_PrString(pResult);                // Print voltage value on LCD
                LCD_1_PrCString(" V      ");            // Print string " V" on LCD after voltage value
        }
}
```

คำถาม

1. ให้วิจารณ์ผลการทดลอง
2. ให้เขียนโปรแกรมวัดค่าแรงดัน VR แสดงผลที่ LCD ในรูปแบบตัวหนังสือและกราฟ
3. ให้อธิบายความหมายของพารามิเตอร์ของ DelSig ต่างๆ โดยศึกษาจาก Delta Sigma ADC Data Sheet

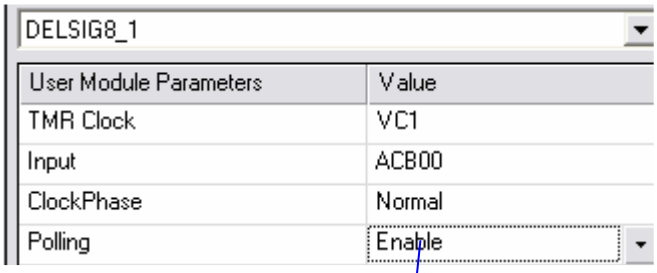
เอกสารอ้างอิง

1. Measure_Display_ 0 to 5V on an LCD, http://www.cypress.com/?rID=37772 (2009)
2. Simple ADC Example, http://www.cypress.com/?rID=38629 (2009)
3. Analog – Adc Selection – AN2239 , www.cypress.com/?docID=17396 (2009)
4. an2093.pdf, “Keypad scan using adc (sar6)”, http://www.psocdeveloper.com/docs/appnotes/an-mode/detail/an-pointer/an2093.html

กรณีชิพไม่มี DelSig ต้องใช้ DelSig8 ซึ่งมีจุดเด่น

- 8-bit resolution
- Data format available in 2's complement
- Sample rate up to 32 ksps
- 64X over-sampling with $sinc^2$ filter reduces anti-alias requirements
- Input range defined by internal and external reference options
- Internal or external clock
- Second-order modulator available for the CY8C29/27/24/22xxx family of PSoC devices

ต้องกำหนดพารามิเตอร์ดังนี้



ต้อง Enable เพื่อให้ใช้ API ต่อไปนี้ได้

```
#if ( DELSIG8_1_POLL_ENABLE )
    extern BYTE DELSIG8_1_fIsDataAvailable(void);      // RAM use class 4
    extern CHAR DELSIG8_1_cGetData(void);              // RAM use class 4
    extern CHAR DELSIG8_1_cGetDataClearFlag(void);     // RAM use class 4
    extern void DELSIG8_1_ClearFlag(void);             // RAM use class 4
#endif
```

และเขียนโปรแกรมดังนี้

```
//--------------------------------------------------------------------------
// C main line
//--------------------------------------------------------------------------

#include <m8c.h>                    // part specific constants and macros
#include "PSoCAPI.h"                // PSoC API definitions for all User Modules
#include "stdlib.h"                 // Add this header to use the ftoa function

char iData;                         // Variable that stores the ADC result
float fVolts;                       // Variable that stores the converted voltage value
float fScaleFactor;                 // Variable that stores the volts/count scale factor
char *pResult;                      // Pointer used to store the result returned by ftoa function
int iStatus;                        // Status variable for the ftoa function
void main()
{
        PGA_1_Start(PGA_1_HIGHPOWER);          // Start PGA with Highpower
        LCD_1_Start();                         // Start LCD
        LCD_1_Position(0,0);                   // Set LCD position to row 0 column 0
        LCD_1_PrCString("MEASURED VOLTAGE");   // Print string "MEASURED VOLTAGE" on LCD
        M8C_EnableGInt;                        // Enable Global Interrupts
        DELSIG8_1_Start(DELSIG8_1_HIGHPOWER);// Start ADC by powering SC block at High Power
        DELSIG8_1_StartAD();                   // Have ADC run continuously
```

```
        fScaleFactor = (float)5/(float)256;             // Calculate Scale Factor.
        for(;;)                                         // Infinite loop
        {
                while(DELSIG8_1_fIsDataAvailable() == 0);  // Loop until value ready
                iData=DELSIG8_1_cGetData();             // Read ADC result
                DELSIG8_1_ClearFlag();                  // Clear ADC flag
                if(iData>=128)                          //Convert signed 8 bits value to unsiged 8 bits.
                {
                        fVolts = 128.0-(256.0-(float)iData);
                }
                else
                {
                        fVolts = 128.0+(float)iData;
                }
                fVolts = fScaleFactor*(float)fVolts;  // Calculate voltage using ADC result and scale factor
                pResult = ftoa(fVolts,&iStatus );     // Convernt Float value of voltage into ASCII string
                LCD_1_Position(1,0);                  // Set LCD position to row 1 column 0
                LCD_1_PrString(pResult);              // Print voltage value on LCD
                LCD_1_PrCString(" V      ");          // Print string " V" on LCD after voltage value
        }
}
```