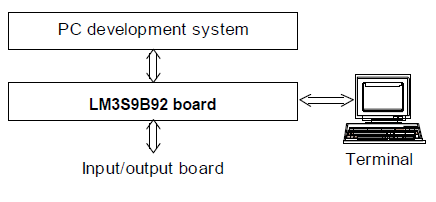


**Introduction Of the Experiments:**

The assignment for this lab mainly consists of two parts. Both parts demand the creation of a C program: the first one – for an external D/A converter, the second one – for an internal A/D converter. The Figure shows the appropriate PC development system and LM3S9B92 board for the implementation of the created code.



In the lab the appropriate connection between the components of the digital voltmeter and the input/output board should be created. Furthermore, the code that was written as a part of the preparation for the lab should be implemented via using the editor, compiler, and linker of the PC system. That’s how the control software for the programs is developed. Finally, the control software should be transferred to the LM3S9B92 board for test.

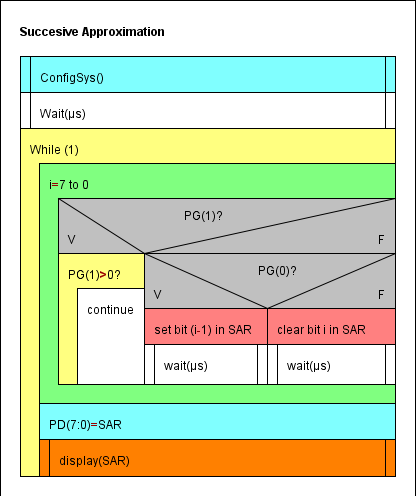
**2.Experiments**

2.1 External D/A converter :

A digital-to-analog converter (D/A) is a device which is used to handle the conversion of a digital value to an analog signal. The digital value is usually represented by a binary code, while an analog signal can be current, voltage, or electric charge.

The transfer characteristics of the internal D/A converter were given in the assignment. The corresponding circuit diagram is shown on Figure 2. The method that was used in order to develop a C program for such a converter is called weighting conversion method. This method is based on the following: first, an output is a binary number with only MSB set. Then the comparison of the corresponding output voltage Uout with the input voltage Ue is done. If Uout is less than Ue, the voltage level at PG(0) is set to high level. If Uout is more than Ue, the comparator puts out a low level. So, the decision whether the MSB should be cleared or not is determined. The sequence of these steps hold for all remaining bits and as a result we get a binary value which corresponds to the input voltage Ue. Afterwards, the voltage is being output. Port(J) and port(E) are used for displaying three digits of this output. When the change of the digital input occurs, it takes some time for a comparator to come to a stable mode. In our program we should consider the delay for the comparator which is equal to 30µs.

**Nassi-Shneiderman diagram:**



------------------------------------------------------------------------------------------------------------------------------

**C code (2.1):**

#define PART\_LM3S9B92

#include "lm3s9b92.h"

#include "stdio.h"

void wait(unsigned short usec)

{

TIMER0\_TAILR\_R=((16000/8)\* (usec)); // ILR= 16M/122\*0.5-1

TIMER0\_CTL\_R |= 0x0001; // enable Timer 0

while((TIMER0\_RIS\_R&(1<<0))==0);

TIMER0\_ICR\_R=0x01;

TIMER0\_CTL\_R &= ~0x0001; // disabling the Timer \*/

}

void configSys(void)

{

int i=0;

SYSCTL\_RCGC2\_R |= (1<<3)| (1<<6)| (1<<8)| (1<<4); // PD-PG-PJ-PE

SYSCTL\_RCC\_R =((SYSCTL\_RCC\_R |0x00000540)&~ 0x000002B1);

//--------------------------------------------------------------------

//PORTD

GPIO\_PORTD\_DEN\_R = 0xFF;

GPIO\_PORTD\_DIR\_R = 0xFF;

//--------------------------------------------------------------------

//PORTE

GPIO\_PORTE\_DEN\_R |=0x07;

GPIO\_PORTE\_DIR\_R |=0x07;

//--------------------------------------------------------------------

//PORTG

GPIO\_PORTG\_DEN\_R = 0x03;

GPIO\_PORTG\_DIR\_R = 0x00;

//--------------------------------------------------------------------------------

//PORTJ

GPIO\_PORTJ\_DEN\_R = 0xFF;

GPIO\_PORTJ\_DIR\_R = 0xFF;

//--------------------------------------------------------------------------------

//Timer Configuration

SYSCTL\_RCGC1\_R |= (1<<16);

i++;

TIMER0\_CTL\_R &= ~0x0001; // disable Timer 0

TIMER0\_CFG\_R = 0x04; // 2 x 16-bit mode

TIMER0\_TAMR\_R = 0x22; // periodic mode + match enable

TIMER0\_TAPR\_R = 7; // pre-scaler PR= ceil((16M/2^16)\*0.03) -1}

}

void display(int SAR)

{

int total, temp1, dig1,dig2;

int buyTime=0;

GPIO\_PORTE\_DATA\_R= 0x01;

buyTime++;

total=(int)(SAR \* 19.53125);

temp1=total% 100;

GPIO\_PORTJ\_DATA\_R= (int)(((temp1/10)<<4)|(temp1%10));

buyTime++;

total/= 100;

GPIO\_PORTE\_DATA\_R= 0x02;

buyTime++;

GPIO\_PORTJ\_DATA\_R= (int)(((total/10)<<4)|(total%10));

}

void main(void)

{

int i, SAR;

configSys();

while(1)

{

SAR= 0x80;

GPIO\_PORTE\_DATA\_R |=0x04;

GPIO\_PORTE\_DATA\_R &=0x04;

for(i=7; i>0;i--)

{

if(GPIO\_PORTG\_DATA\_R & 0x02)

{

while(GPIO\_PORTG\_DATA\_R>1);

}

else

{

GPIO\_PORTD\_DATA\_R=SAR; //giving the value of SAR for PORTD

wait(3); //giving enough time to Port D to get updated

if(GPIO\_PORTG\_DATA\_R & 0x01) //Input comparison by //comparator

SAR|=(1<<i-1); //set the i-1 th bit to 1

else

{

SAR &=~(1<<i); //clear the i-th bit

SAR |=(1<<i-1);//set the i-1 th bit to 1

}

}

}

display(SAR); //display the value

}

}

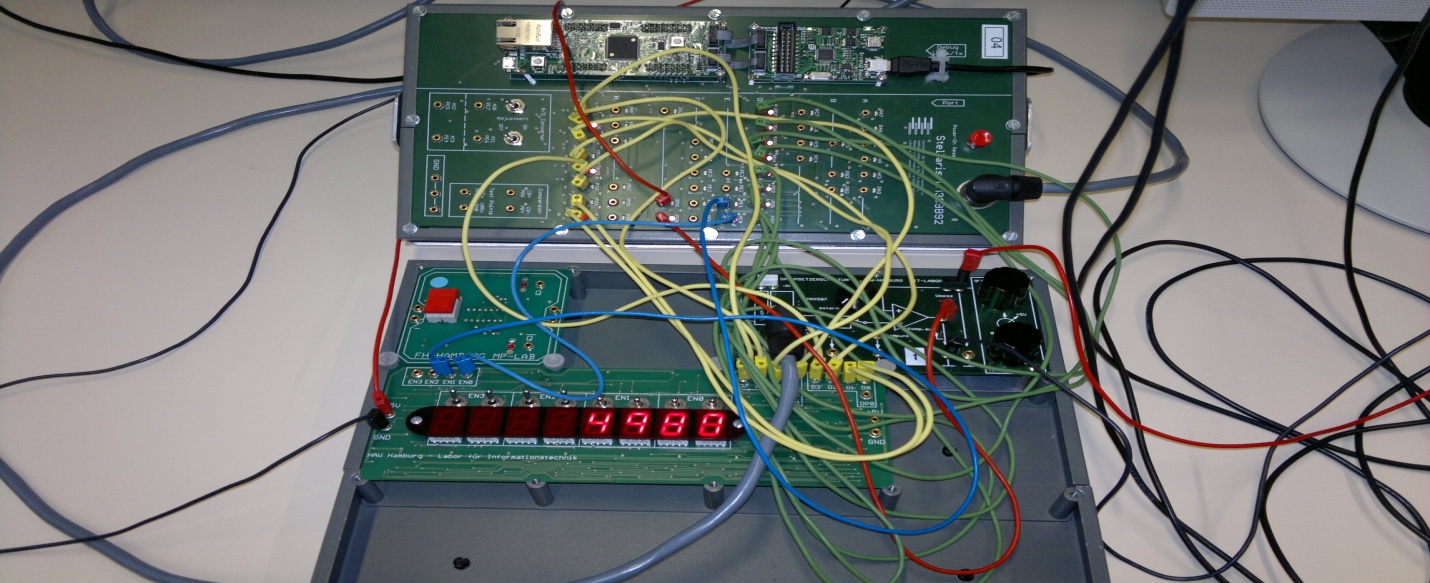
//-----------------------------------------------------------------------------------------------------------------------------

**Experimental setup :**

\*Measuring UE with a digital voltmeter,

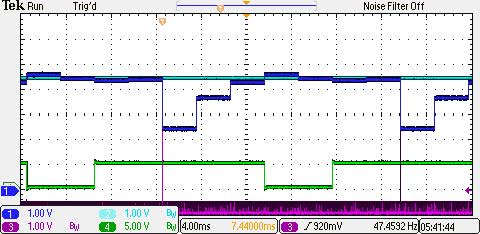
On voltmeter : 5 V

On Display : 4.988 V



**Display on Oscilloscope :**

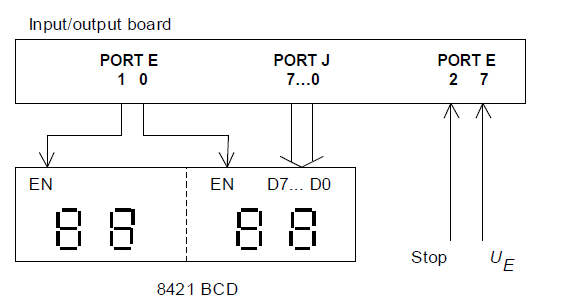
(Ch1:Vout )(Ch2:Analog input voltage)(Ch3:trigger input)(Ch4:output comparator)



**2.3 Internal A/D converter :**

The A/D conversion transforms the analog input voltage *UE* at pin PE(7) (= AI0) into an according binary value.

The resulting binary number is proportional to *UE*. The voltage has to be output with four digits using ports E and J .



**C code :**

#define PART\_LM3S9B92

#include "lm3s9b92.h"

#include "stdio.h"

void wait (unsigned short usec)

{

int i;

TIMER0\_TAILR\_R=((16000 /8)\* (usec)); // ILR= 16M/122\*0.5-1

TIMER0\_CTL\_R |= 0x0001; // enable Timer 0

while((TIMER0\_RIS\_R&(1<<0))==0);

TIMER0\_ICR\_R=0x01;

TIMER0\_CTL\_R &= ~0x0001; // disable Timer

}

void configSys(void)

{

int i;

SYSCTL\_RCGC2\_R |= (1<<3)| (1<<6)| (1<<8)| (1<<4); // PD-PG-PJ-PE

SYSCTL\_RCGC0\_R |= (1<<16); //ADC0

SYSCTL\_RCC\_R =((SYSCTL\_RCC\_R |0x00000540)&~ 0x000002B1);

//PORTE

GPIO\_PORTE\_DIR\_R |=0x07;

GPIO\_PORTE\_DIR\_R &=~0x04;

// configure AIN0 (=PE(7)) as analog inputs

GPIO\_PORTE\_DEN\_R |=0x07;

GPIO\_PORTE\_AFSEL\_R |= 0x80;// Alternating

GPIO\_PORTE\_DEN\_R &= ~0x80; // PE7 disable digital IO

GPIO\_PORTE\_DIR\_R &= ~0x80; // Setting the direction

GPIO\_PORTE\_AMSEL\_R |= 0x80; // PE7 enable analog function

//PORTJ

GPIO\_PORTJ\_DEN\_R = 0xFF;

GPIO\_PORTJ\_DIR\_R = 0xFF;

// ADC0\_SS0 configuration

ADC0\_ACTSS\_R &= ~0x0F; // disable all 4 sequencers of ADC0

ADC0\_SSMUX0\_R |= 0x00000000; // sequencer 0, channel AIN2,AIN0,AIN1

//ADC0\_SSCTL0\_R |= 0x00000200; // END2 set, sequence length = 3

ADC0\_ACTSS\_R |=0x01;

//Timer Configuration

SYSCTL\_RCGC1\_R |= (1<<16);

i++;

TIMER0\_CTL\_R &= ~0x0001; // disable Timer 0

TIMER0\_CFG\_R = 0x04; // 2 x 16-bit mode

TIMER0\_TAMR\_R = 0x22; // periodic mode + match enable

TIMER0\_TAPR\_R =7 ; // pre-scaler PR= ceil((16M/2^16)\*0.03) -1

}

void display()

{

int total, temp1, dig1,dig2;

while(ADC0\_SSFSTAT0\_R & (1<<8));

total=(int)(ADC0\_SSFIFO0\_R \*5000/1023);

GPIO\_PORTE\_DATA\_R= 0x01;

temp1=total% 100;

dig1= temp1%10;

dig2=(int)(temp1/10);

GPIO\_PORTJ\_DATA\_R= (int)((dig2<<4)|dig1);

total/= 100;

GPIO\_PORTE\_DATA\_R= 0x02;

wait(3);

dig1= total%10;

dig2=(int)(total/10);

GPIO\_PORTJ\_DATA\_R=(int)((dig2<<4)|dig1);

}

void main(void)

{

configSys();

while(1)

{

GPIO\_PORTE\_DATA\_R=0x04;

GPIO\_PORTE\_DATA\_R=0x00;

display();

}

}

//------------------------------------------------------------------------------------------------------------------------

**4.Conclusion**

Result and Evaluation:

The result of our program fulfills the need for this task. The corresponding value is shown on the 7-segment display.

This experiment allows us to see how an A/D converter works.

It takes an input voltage (Analog value)( Voltage could only display from 0 to 5V), and converts it into a digital

value. In order to write the code we need the transfer characteristics of the A/D converter. We have to write a

code that allows to take an input analog data and then convert it into a set of binary digits. We had to use different

calculation in our source code.