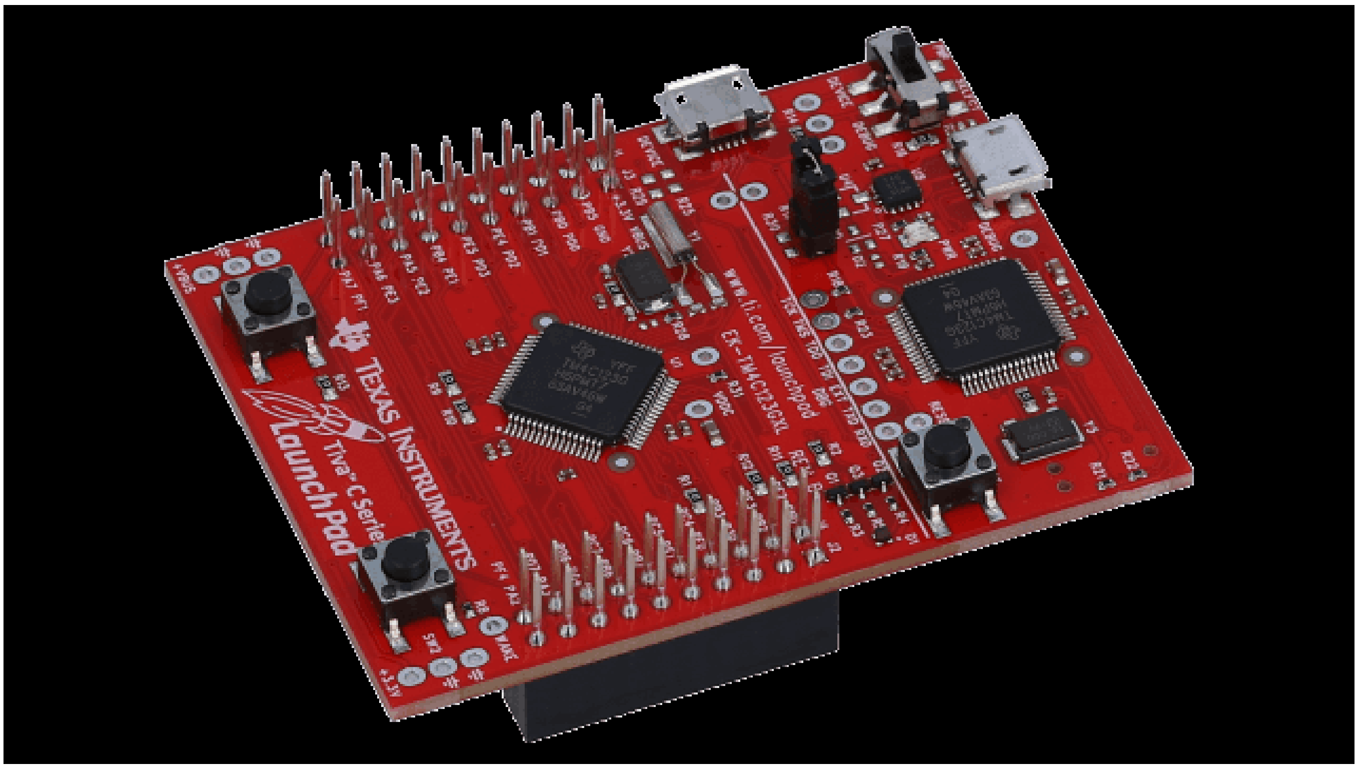
ORTA DOĞU TEKNİK ÜNİVERSİTESİ
MIDDLE EAST TECHNICAL UNIVERSITY

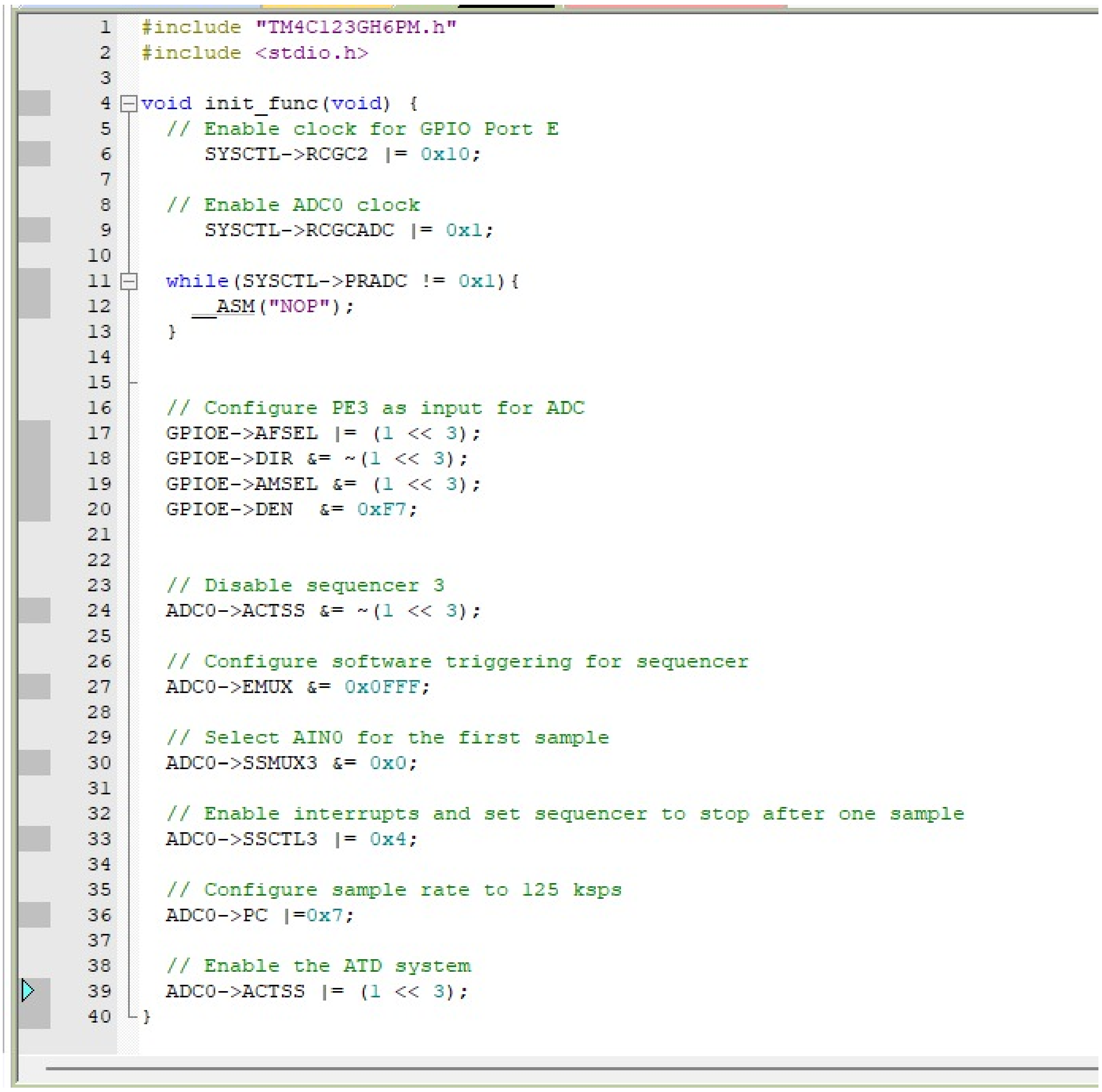
EE447- Introduction to Microprocessors Laboratory With Assembly Programming (Preliminary Work 5)

EXPERIMENTAL WORK NO: 5
1st Group Member: Uğur SAMANCI - 2398915
2nd Group Member: Barış GÜZEL - 2304764



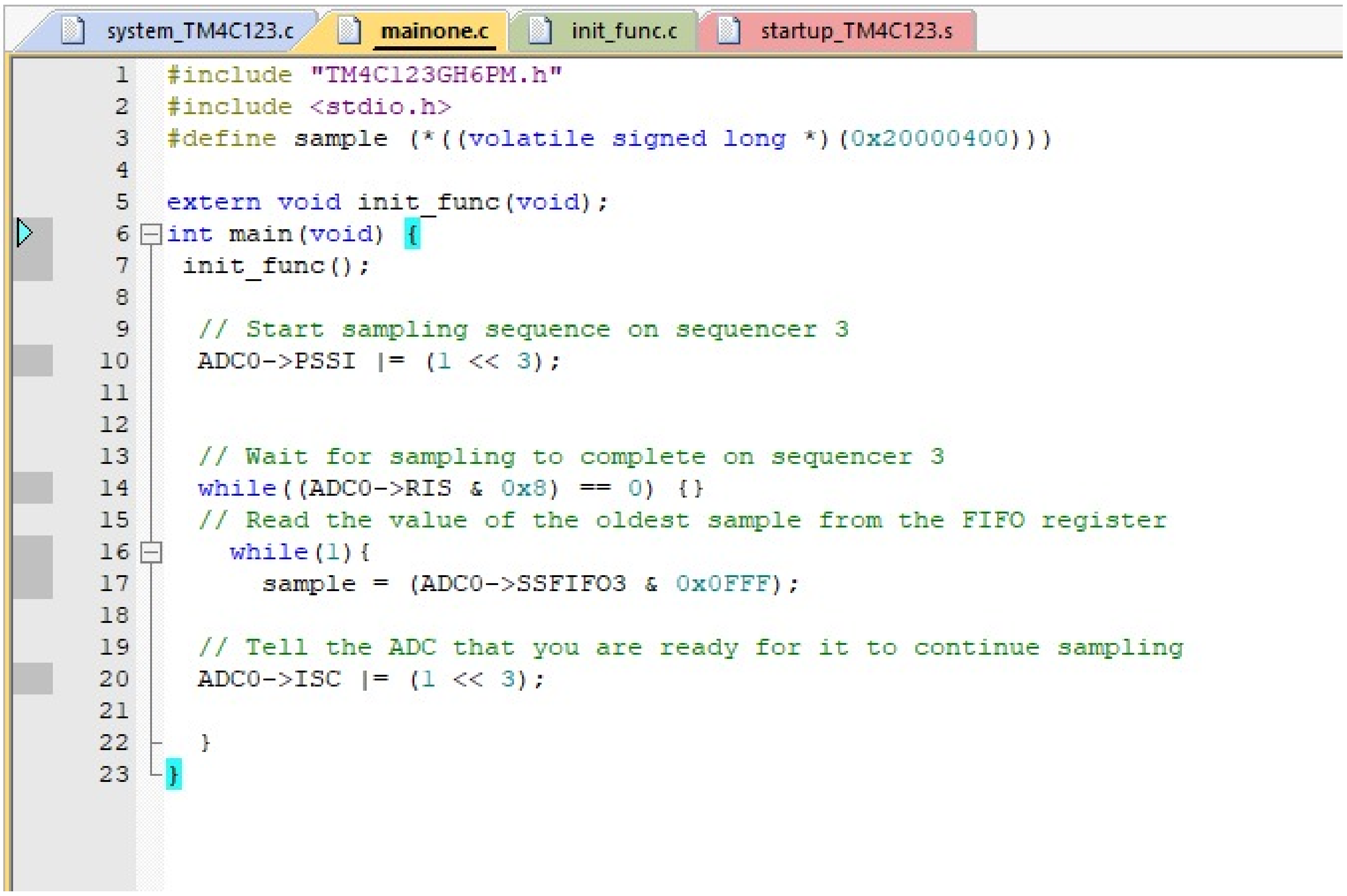
Question 1)

➢ In this part of the preliminary work, we created two c function one of them for initialization process and one for the main part one can see them in figure 1 and figure 2. Note that through question 5 initialization is same so that one figure only added for first four parts.



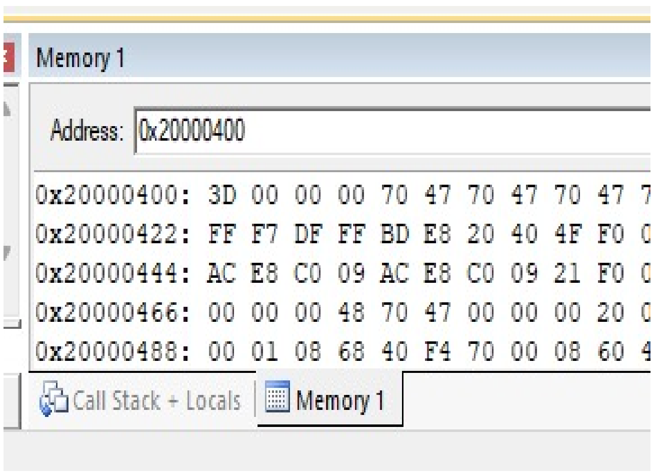
```c
#include "TM4C123GH6PM.h"
#include <stdio.h>

void init_func(void) {
  // Enable clock for GPIO Port E
     SYSCTL->RCGC2  |= 0x10;

  // Enable ADC0 clock
     SYSCTL->RCGCADC  |= 0x1;

  while(SYSCTL->PRADC != 0x1){
     __ASM("NOP");
  }


  // Configure PE3 as input for ADC
  GPIOE->AFSEL |= (1 << 3);
  GPIOE->DIR &= ~(1 << 3);
  GPIOE->AMSEL &= (1 << 3);
  GPIOE->DEN  &= 0xF7;


  // Disable sequencer 3
  ADC0->ACTSS &= ~(1 << 3);

  // Configure software triggering for sequencer
  ADC0->EMUX &= 0x0FFF;

  // Select AIN0 for the first sample
  ADC0->SSMUX3 &= 0x0;

  // Enable interrupts and set sequencer to stop after one sample
  ADC0->SSCTL3 |= 0x4;

  // Configure sample rate to 125 ksps
  ADC0->PC |=0x7;

  // Enable the ATD system
  ADC0->ACTSS |= (1 << 3);
}
```

Figure 1: Initialization function


system_TM4C123.c mainone.c init_func.c startup_TM4C123.s

```c
#include "TM4C123GH6PM.h"
#include <stdio.h>
#define sample (*((volatile signed long *)(0x20000400)))

extern void init_func(void);
int main(void) {
 init_func();

  // Start sampling sequence on sequencer 3
  ADC0->PSSI |= (1 << 3);


  // Wait for sampling to complete on sequencer 3
  while((ADC0->RIS & 0x8) == 0) {}
  // Read the value of the oldest sample from the FIFO register
    while(1){
      sample = (ADC0->SSFIFO3 & 0x0FFF);

  // Tell the ADC that you are ready for it to continue sampling
  ADC0->ISC |= (1 << 3);

  }
}
```

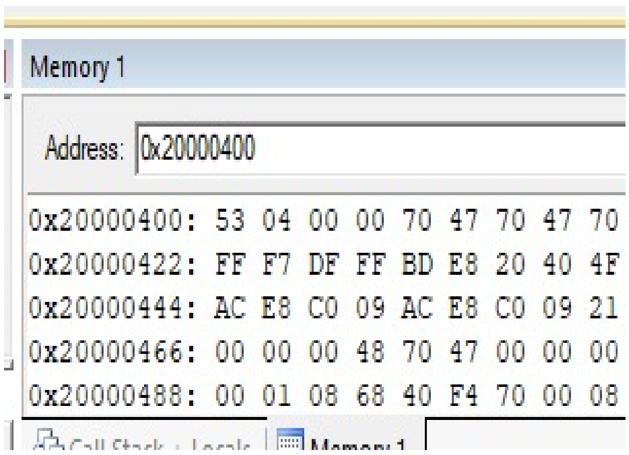
Figure 2: Main function for part one


Memory 1

Address: 0x20000400

```
0x20000400: 3D 00 00 00 70 47 70 47 70 47 7
0x20000422: FF F7 DF FF BD E8 20 40 4F F0 0
0x20000444: AC E8 C0 09 AC E8 C0 09 21 F0 0
0x20000466: 00 00 00 48 70 47 00 00 20 0
0x20000488: 00 01 08 68 40 F4 70 00 08 60 4
```

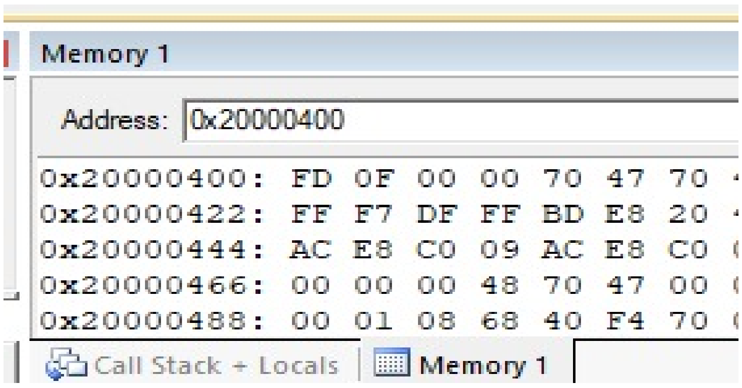
Figure 3:POT Low Value


Memory 1

Address: 0x20000400

```
0x20000400: 53 04 00 00 70 47 70 47 70
0x20000422: FF F7 DF FF BD E8 20 40 4F
0x20000444: AC E8 C0 09 AC E8 C0 09 21
0x20000466: 00 00 00 48 70 47 00 00 00
0x20000488: 00 01 08 68 40 F4 70 00 08
```

Figure 4:POT Mid Value


Memory 1

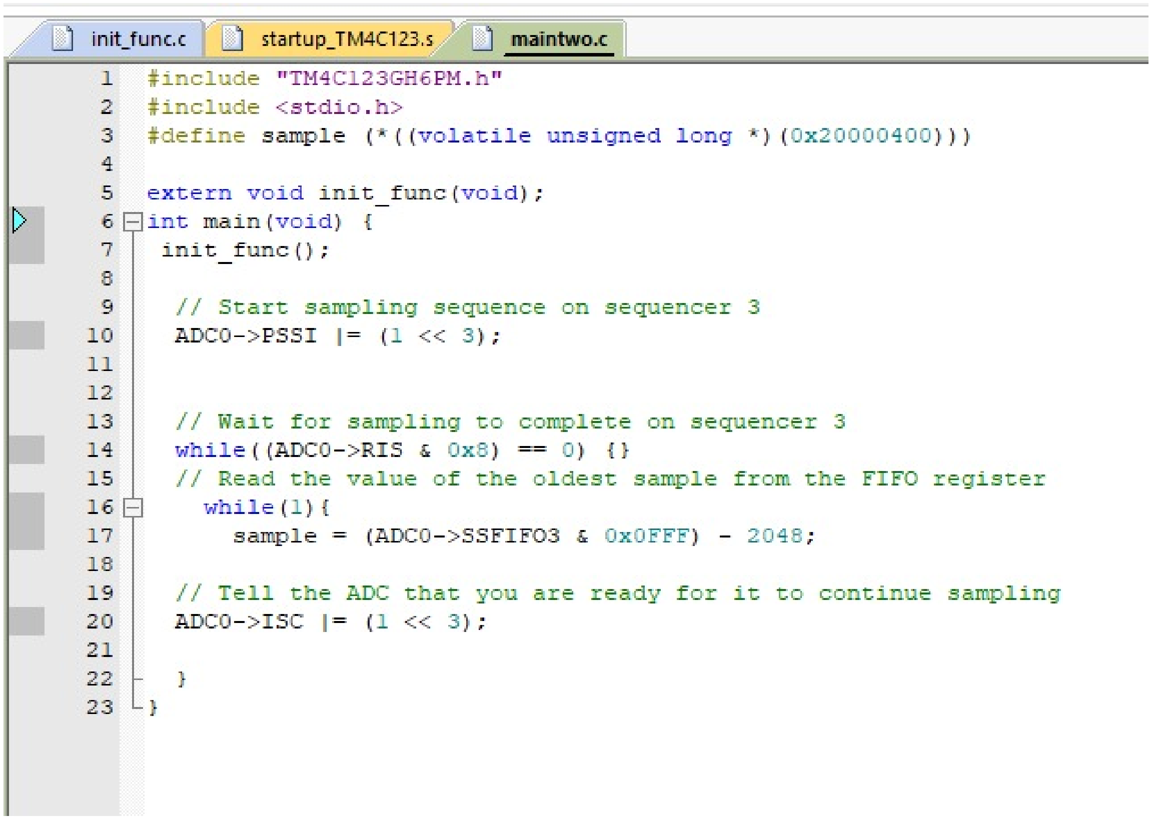
Address: 0x20000400

```
0x20000400: FD 0F 00 00 70 47 70
0x20000422: FF F7 DF FF BD E8 20
0x20000444: AC E8 C0 09 AC E8 C0
0x20000466: 00 00 00 48 70 47 00
0x20000488: 00 01 08 68 40 F4 70
```

Figure 5:POT Max Value

Question 2)

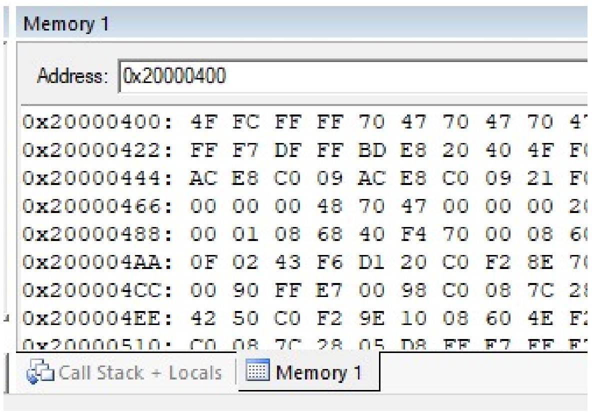
➢ In this part of the preliminary work, we used functions we already defined in question 1. We simply calculated step and we substract 2048 value which is equal to 1.65 V as a result our interval changed from 0-3.3v to -1.65v-1.65v one can see result from below figures.



init_func.c | startup_TM4C123.s | maintwo.c

```c
#include "TM4C123GH6PM.h"
#include <stdio.h>
#define sample (*((volatile unsigned long *)(0x20000400)))

extern void init_func(void);
int main(void) {
 init_func();

  // Start sampling sequence on sequencer 3
  ADC0->PSSI |= (1 << 3);


  // Wait for sampling to complete on sequencer 3
  while((ADC0->RIS & 0x8) == 0) {}
  // Read the value of the oldest sample from the FIFO register
    while(1){
      sample = (ADC0->SSFIFO3 & 0x0FFF) - 2048;

  // Tell the ADC that you are ready for it to continue sampling
  ADC0->ISC |= (1 << 3);

  }
}
```

Figure 6:Main for part two

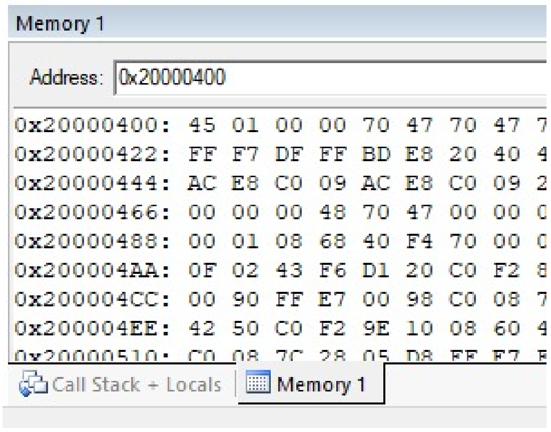


Memory 1

Address: 0x20000400

```
0x20000400: 4F FC FF FF 70 47 70 47 70 47
0x20000422: FF F7 DF FF BD E8 20 40 4F F
0x20000444: AC E8 C0 09 AC E8 C0 09 21 F
0x20000466: 00 00 00 48 70 47 00 00 00 2
0x20000488: 00 01 08 68 40 F4 70 00 08 6
0x200004AA: 0F 02 43 F6 D1 20 C0 F2 8E 7
0x200004CC: 00 90 FF E7 00 98 C0 08 7C 2
0x200004EE: 42 50 C0 F2 9E 10 08 60 4E F
0x20000510: C0 08 7C 28 05 D8 FF F7 FF F
```

Call Stack + Locals | Memory 1



Memory 1

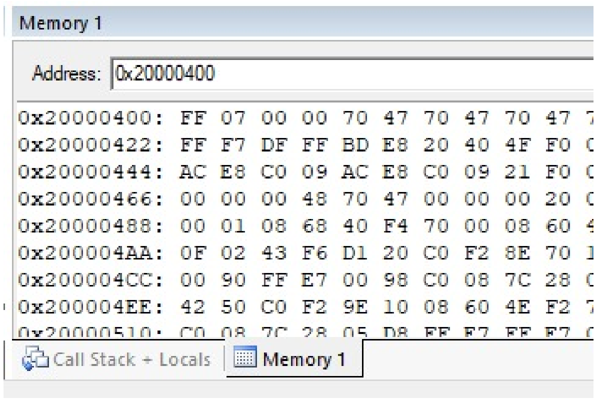
Address: 0x20000400

```
0x20000400: 45 01 00 00 70 47 70 47 7
0x20000422: FF F7 DF FF BD E8 20 40 4
0x20000444: AC E8 C0 09 AC E8 C0 09 2
0x20000466: 00 00 00 48 70 47 00 00 0
0x20000488: 00 01 08 68 40 F4 70 00 0
0x200004AA: 0F 02 43 F6 D1 20 C0 F2 8
0x200004CC: 00 90 FF E7 00 98 C0 08 7
0x200004EE: 42 50 C0 F2 9E 10 08 60 4
0x20000510: C0 08 7C 28 05 D8 FF F7 F
```

Call Stack + Locals | Memory 1

Figure 7: POT Low value

Figure 8: POT Mid Value



Memory 1

Address: 0x20000400

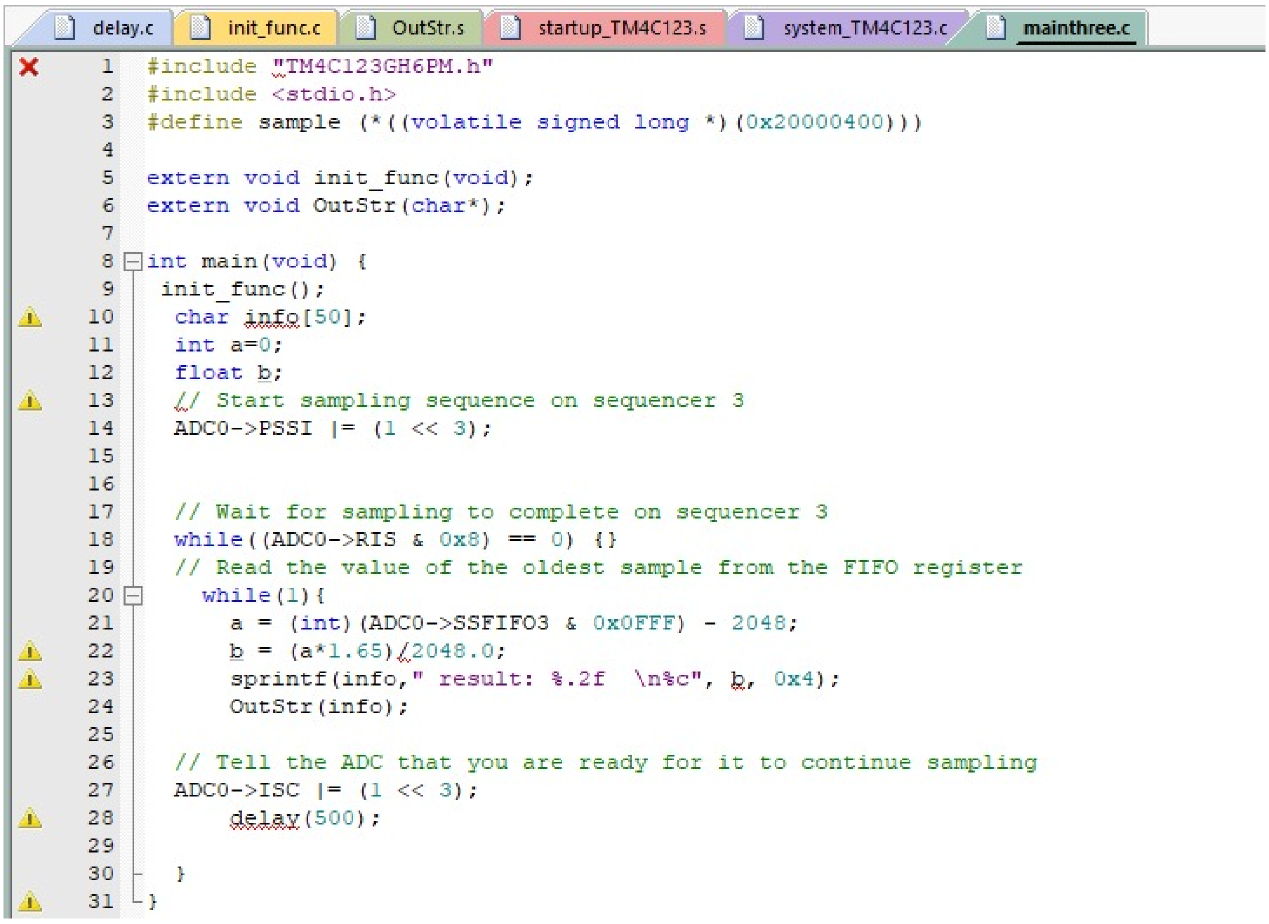
```
0x20000400: FF 07 00 00 70 47 70 47 70 47 7
0x20000422: FF F7 DF FF BD E8 20 40 4F F0 C
0x20000444: AC E8 C0 09 AC E8 C0 09 21 F0 C
0x20000466: 00 00 00 48 70 47 00 00 00 20 C
0x20000488: 00 01 08 68 40 F4 70 00 08 60 4
0x200004AA: 0F 02 43 F6 D1 20 C0 F2 8E 70 1
0x200004CC: 00 90 FF E7 00 98 C0 08 7C 28 C
0x200004EE: 42 50 C0 F2 9E 10 08 60 4E F2 7
0x20000510: C0 08 7C 28 05 D8 FF F7 FF F7 C
```

Call Stack + Locals | Memory 1

Figure 9: POT Max Value

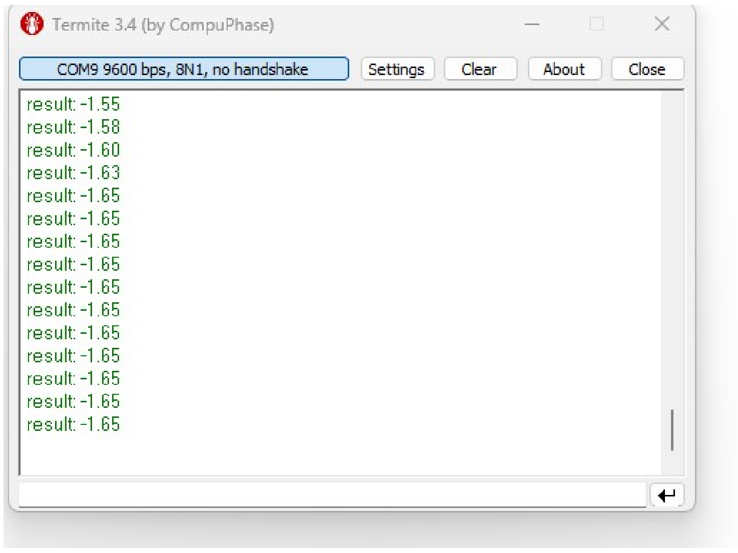
Question 3)

➢ In this part of the preliminary work, we used functions we already defined in question 2. We multiplied the value with 1.65 and divided by 2048 to get values as -1.65 – 1.65 in decimal format and we used termite to show that our code is work. Results are shown in figures below.



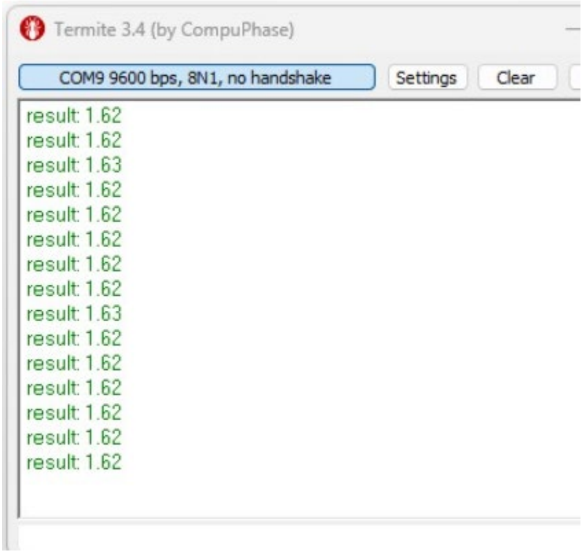
```c
#include "TM4C123GH6PM.h"
#include <stdio.h>
#define sample (*((volatile signed long *)(0x20000400)))

extern void init_func(void);
extern void OutStr(char*);

int main(void) {
 init_func();
  char info[50];
  int a=0;
  float b;
  // Start sampling sequence on sequencer 3
  ADC0->PSSI |= (1 << 3);


  // Wait for sampling to complete on sequencer 3
  while((ADC0->RIS & 0x8) == 0) {}
  // Read the value of the oldest sample from the FIFO register
    while(1){
      a = (int)(ADC0->SSFIFO3 & 0x0FFF) - 2048;
      b = (a*1.65)/2048.0;
      sprintf(info," result: %.2f  \n%c", b, 0x4);
      OutStr(info);

  // Tell the ADC that you are ready for it to continue sampling
  ADC0->ISC |= (1 << 3);
      delay(500);

  }
}
```

Figure 10: Main for part three



```
result: -1.55
result: -1.58
result: -1.60
result: -1.63
result: -1.65
result: -1.65
result: -1.65
result: -1.65
result: -1.65
result: -1.65
result: -1.65
result: -1.65
result: -1.65
result: -1.65
result: -1.65
```

Figure 11: Negative Values on termite

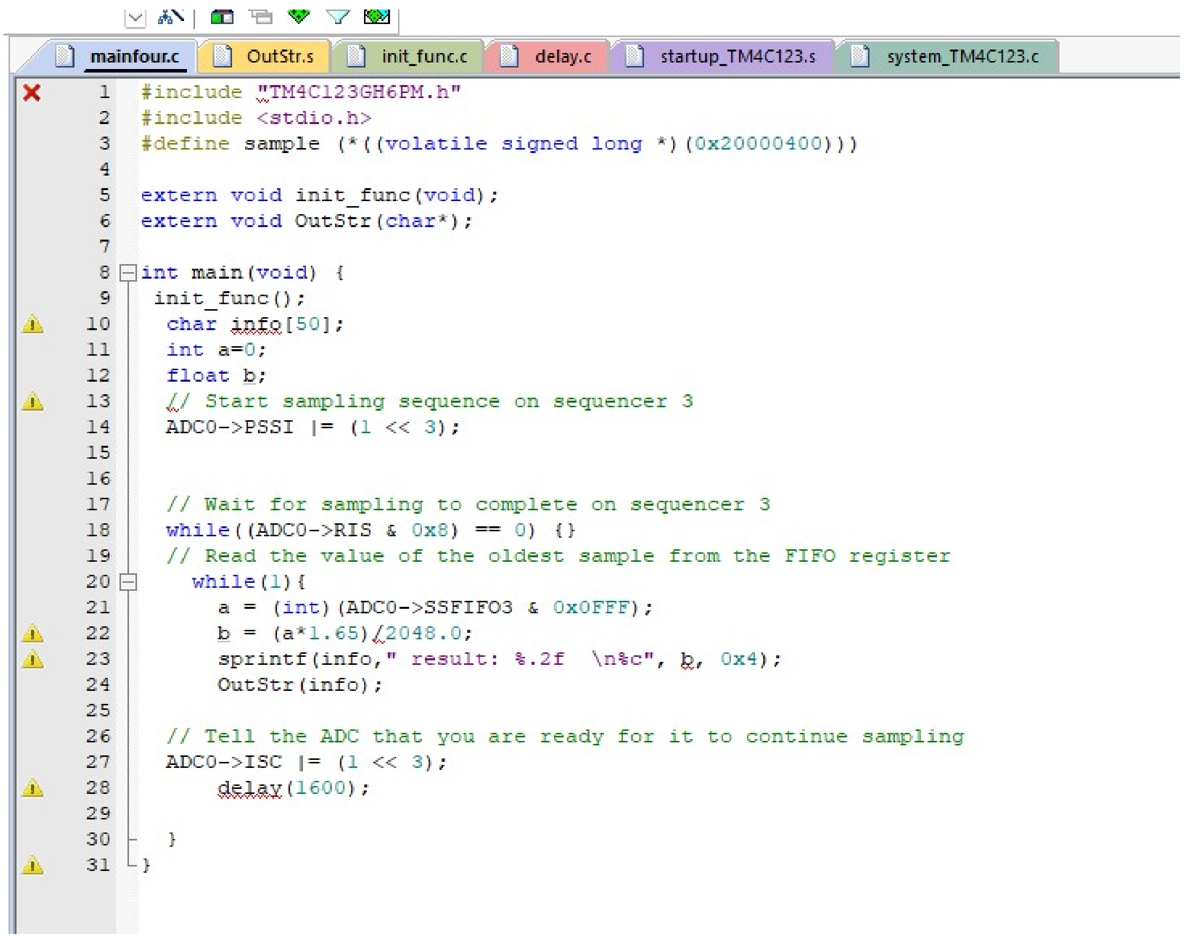


```
result: 1.62
result: 1.62
result: 1.63
result: 1.62
result: 1.62
result: 1.62
result: 1.62
result: 1.62
result: 1.63
result: 1.62
result: 1.62
result: 1.62
result: 1.62
result: 1.62
result: 1.62
```

Figure 12:Positive Valeus on Termite

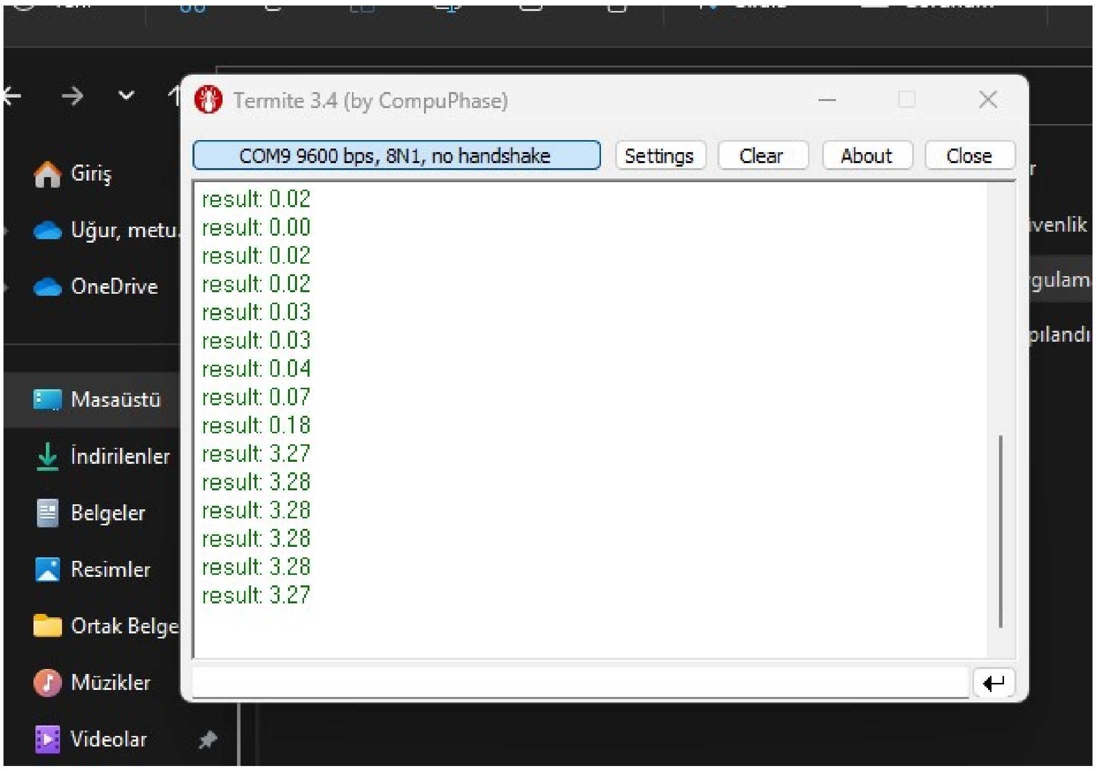
Question 4)

➢ In this part of the preliminary work, we used functions we already defined in question 3. We removed the offset we add in question three so that our range become 0.00 to 3.30 volts. Results are shown in figures below.



```c
#include "TM4C123GH6PM.h"
#include <stdio.h>
#define sample (*((volatile signed long *)(0x20000400)))

extern void init_func(void);
extern void OutStr(char*);

int main(void) {
 init_func();
  char info[50];
  int a=0;
  float b;
  // Start sampling sequence on sequencer 3
  ADC0->PSSI |= (1 << 3);


  // Wait for sampling to complete on sequencer 3
  while((ADC0->RIS & 0x8) == 0) {}
  // Read the value of the oldest sample from the FIFO register
    while(1){
      a = (int)(ADC0->SSFIFO3 & 0x0FFF);
      b = (a*1.65)/2048.0;
      sprintf(info," result: %.2f  \n%c", b, 0x4);
      OutStr(info);

  // Tell the ADC that you are ready for it to continue sampling
  ADC0->ISC |= (1 << 3);
      delay(1600);

  }
}
```

Figure 13: Main for part four

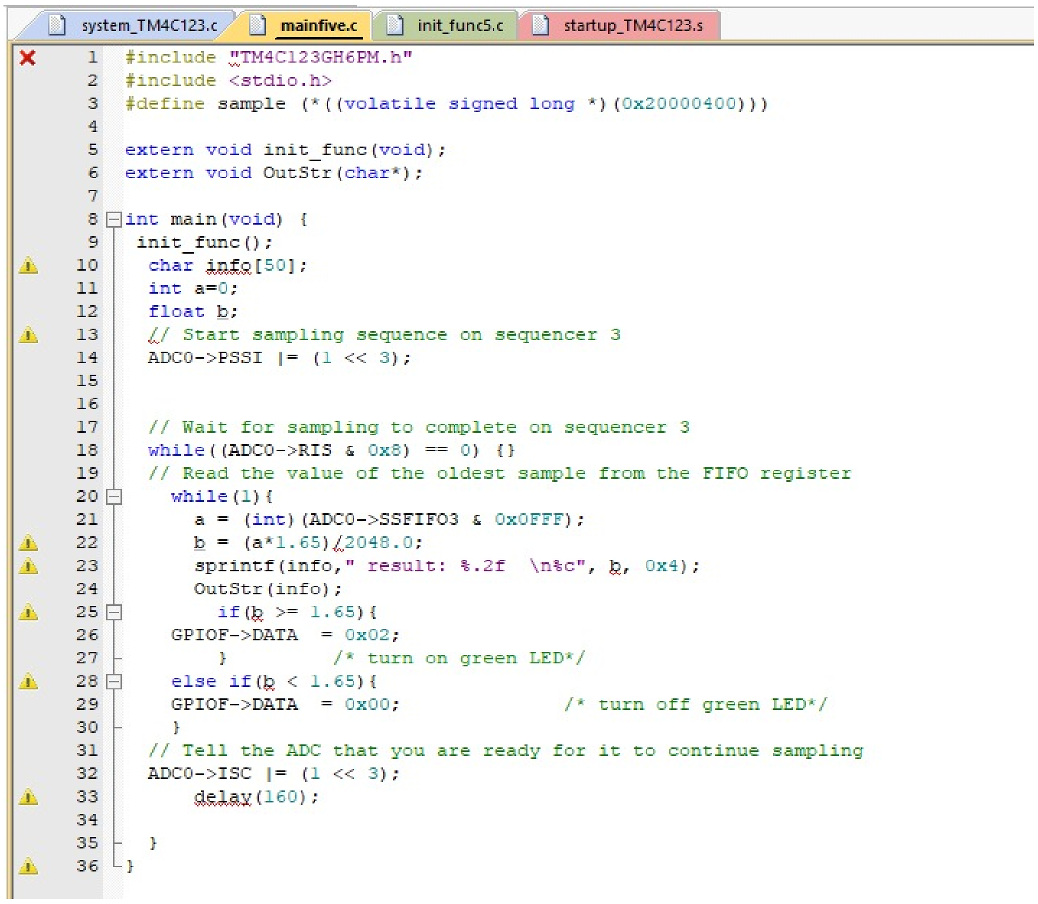


```
result: 0.02
result: 0.00
result: 0.02
result: 0.02
result: 0.03
result: 0.03
result: 0.04
result: 0.07
result: 0.18
result: 3.27
result: 3.28
result: 3.28
result: 3.28
result: 3.28
result: 3.27
```

Figure 14:Termite screen for part four

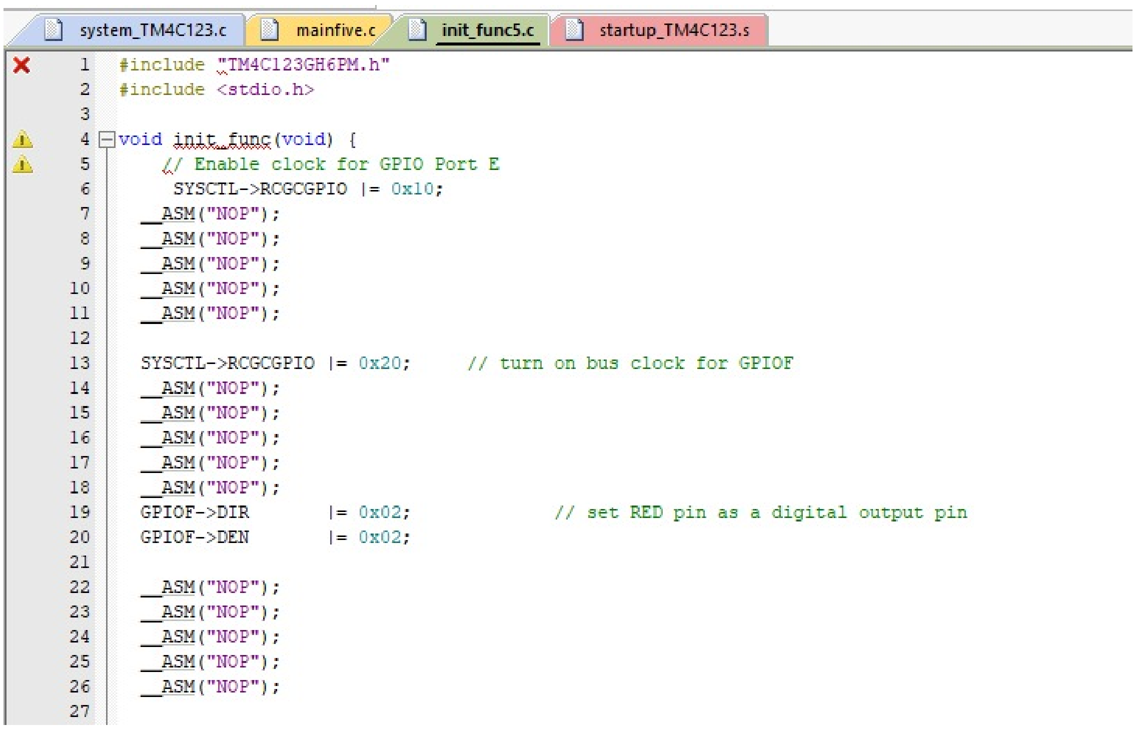
Question 5)

➢ In this part of the preliminary work, we used functions we already defined in question 4. We opened clock for GPIOF and we defined PF1 as output than in our main function for values bigger than 1.65V we opened the led and for values between 0-1.65 we closed the led.



```c
#include "TM4C123GH6PM.h"
#include <stdio.h>
#define sample (*((volatile signed long *)(0x20000400)))

extern void init_func(void);
extern void OutStr(char*);

int main(void) {
 init_func();
  char info[50];
  int a=0;
  float b;
  // Start sampling sequence on sequencer 3
  ADC0->PSSI |= (1 << 3);


  // Wait for sampling to complete on sequencer 3
  while((ADC0->RIS & 0x8) == 0) {}
  // Read the value of the oldest sample from the FIFO register
    while(1){
      a = (int)(ADC0->SSFIFO3 & 0x0FFF);
      b = (a*1.65)/2048.0;
      sprintf(info," result: %.2f  \n%c", b, 0x4);
      OutStr(info);
        if(b >= 1.65){
    GPIOF->DATA  = 0x02;
        }          /* turn on green LED*/
    else if(b < 1.65){
    GPIOF->DATA  = 0x00;              /* turn off green LED*/
    }
  // Tell the ADC that you are ready for it to continue sampling
  ADC0->ISC |= (1 << 3);
      delay(160);

  }
}
```

Figure 15:Main for part five



```c
#include "TM4C123GH6PM.h"
#include <stdio.h>

void init_func(void) {
    // Enable clock for GPIO Port E
     SYSCTL->RCGCGPIO |= 0x10;
  __ASM("NOP");
  __ASM("NOP");
  __ASM("NOP");
  __ASM("NOP");
  __ASM("NOP");

  SYSCTL->RCGCGPIO |= 0x20;     // turn on bus clock for GPIOF
  __ASM("NOP");
  __ASM("NOP");
  __ASM("NOP");
  __ASM("NOP");
  __ASM("NOP");
  GPIOF->DIR       |= 0x02;             // set RED pin as a digital output pin
  GPIOF->DEN       |= 0x02;

  __ASM("NOP");
  __ASM("NOP");
  __ASM("NOP");
  __ASM("NOP");
  __ASM("NOP");

```

Figure 16:Init function changed area for part five