EE3002/IM2002

Microprocessors

Tutorial 12

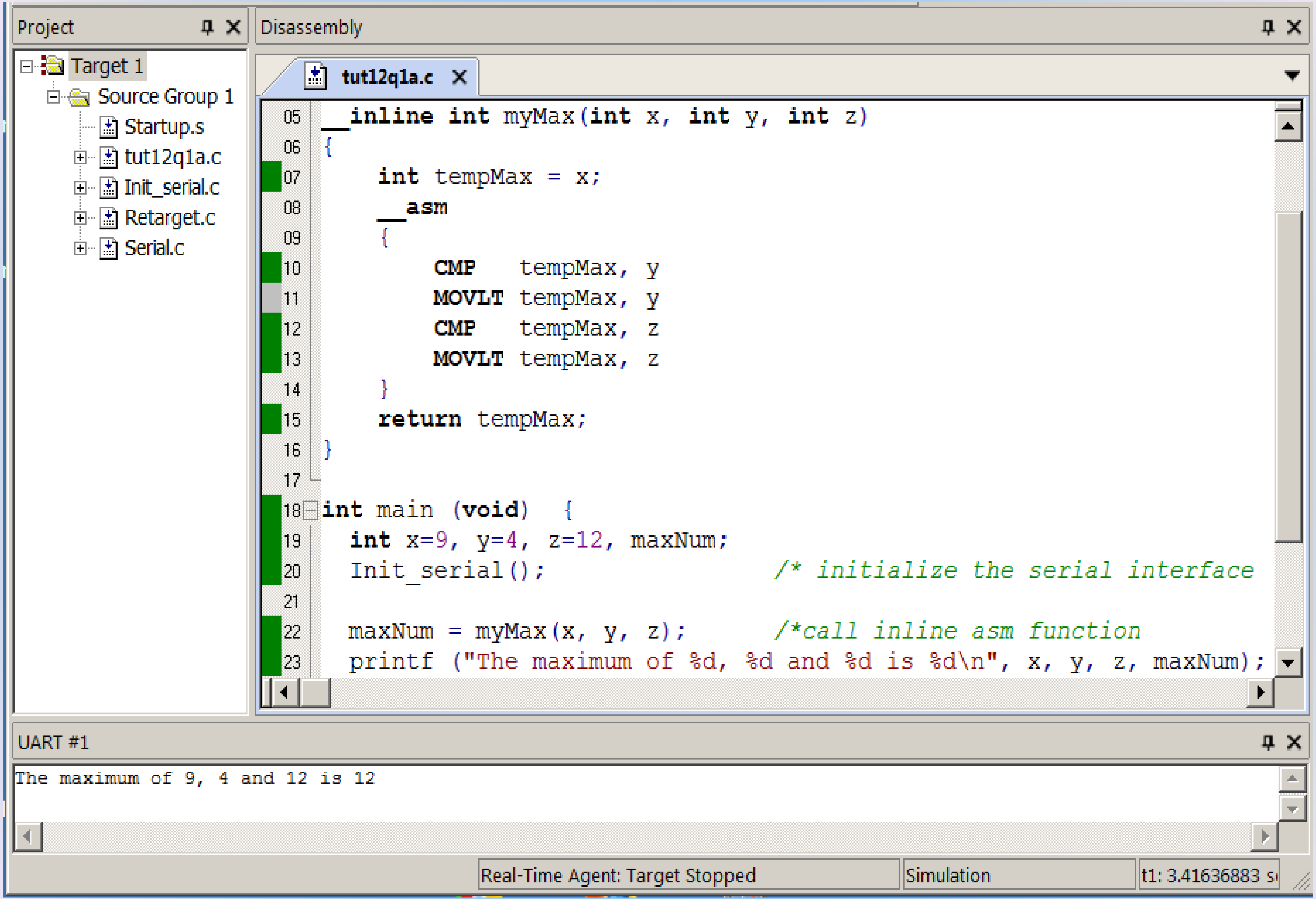
1. Write a short C program that declares 3 integers called x, y and z and determine the maximum value among them. Print out this maximum value to UART0. To test the program, let x = 9, y = 4 and z = 12.

   a) Write the maximum program as an inline assembly function.
   b) Write the maximum program as an embedded assembly function.

a) Using Inline assembly function

```
#include <stdio.h>
#include <LPC21xx.H>            /* LPC21xx definitions                */
extern void Init_serial(void);
__inline int myMax(int x, int y, int z)
{
        int tempMax = x;
        __asm
        {
                CMP     tempMax, y
                MOVLT tempMax, y
                CMP     tempMax, z
                MOVLT tempMax, z
        }
        return tempMax;
}
```

```
int main (void)  {
  int x=9, y=4, z=12, maxNum;

  Init_serial();                         /* initialize the serial interface          */

  maxNum = myMax(x, y, z);    /*call inline asm function                          */

  printf ("The maximum of %d, %d and %d is %d\n", x, y, z, maxNum);

  while (1) {                      /* An embedded program doesn't stop and */
    ;  /* ... */                     /* never returns. An endless loop is used       */
  }                                   /* Replace the dots (...) with your own code.*/
}
```



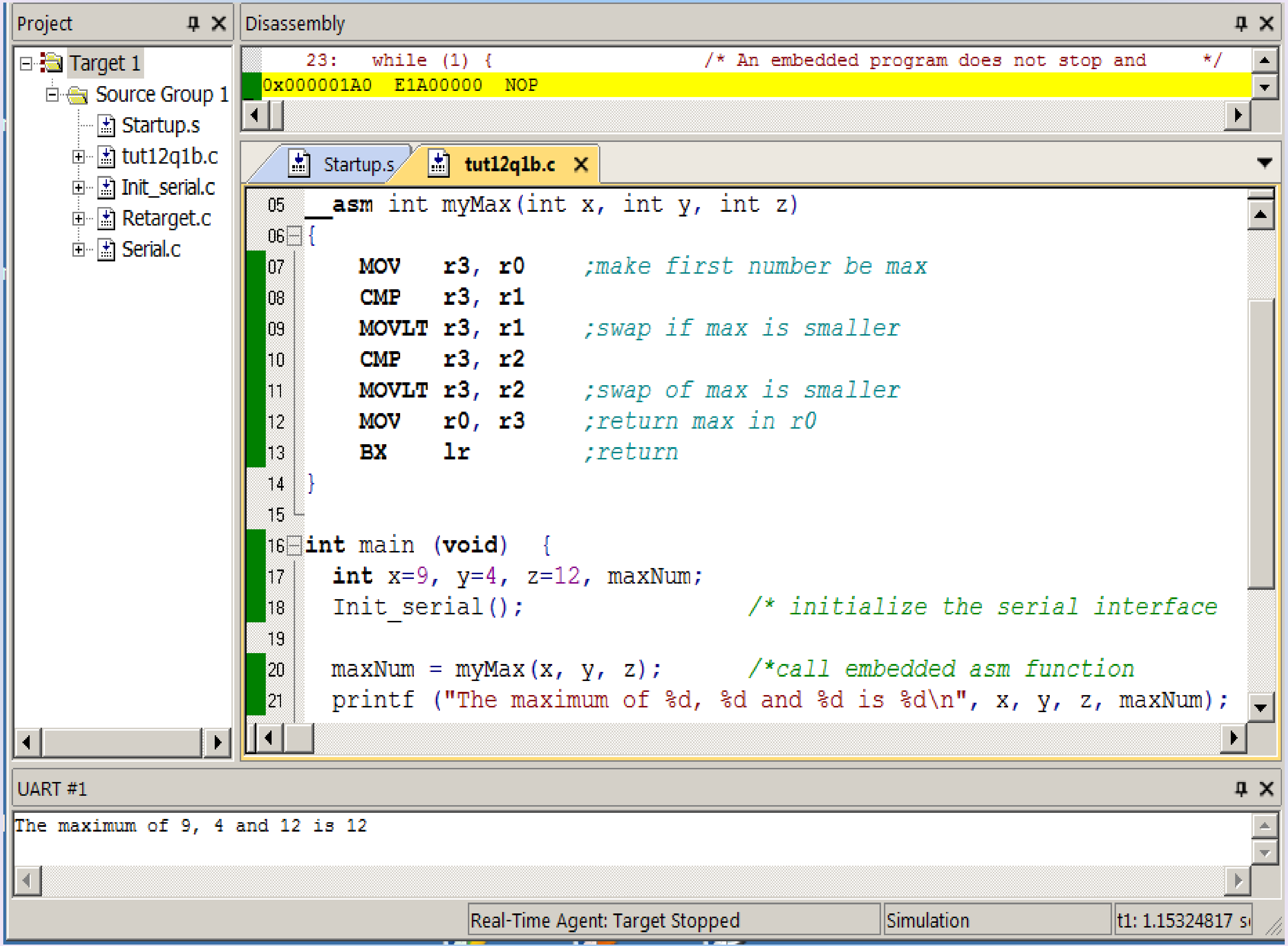
```
__inline int myMax(int x, int y, int z)
{
    int tempMax = x;
    __asm
    {
        CMP    tempMax, y
        MOVLT tempMax, y
        CMP    tempMax, z
        MOVLT tempMax, z
    }
    return tempMax;
}

int main (void)  {
  int x=9, y=4, z=12, maxNum;
  Init_serial();                  /* initialize the serial interface
                                   
  maxNum = myMax(x, y, z);        /*call inline asm function
  printf ("The maximum of %d, %d and %d is %d\n", x, y, z, maxNum);
```

```
The maximum of 9, 4 and 12 is 12
```

b) Using Embedded assembly function

```
#include <stdio.h>
#include <LPC21xx.H>            /* LPC21xx definitions              */
extern void Init_serial(void);

__asm int myMax(int x, int y, int z)
{
        MOV     r3, r0          ;make first number be max
        CMP     r3, r1
        MOVLT r3, r1            ;swap if max is smaller
        CMP     r3, r2
        MOVLT r3, r2            ;swap of max is smaller
        MOV     r0, r3          ;return max in r0
        BX      lr              ;return
}
```

```
int main (void)  {
  int x=9, y=4, z=12, maxNum;

  Init_serial();                          /* initialize the serial interface         */

  maxNum = myMax(x, y, z);    /*call embedded asm function          */

  printf ("The maximum of %d, %d and %d is %d\n", x, y, z, maxNum);

  while (1) {                        /* An embedded program doesn't stop and */
    ;  /* ... */                     /* never returns. An endless loop is used.     */
  }                                  /* Replace the dots (...) with your own code.*/
}
```



```
    23:     while (1) {                          /* An embedded program does not stop and     */
0x000001A0  E1A00000  NOP
```

```
__asm int myMax(int x, int y, int z)
{
    MOV    r3, r0     ;make first number be max
    CMP    r3, r1
    MOVLT  r3, r1     ;swap if max is smaller
    CMP    r3, r2
    MOVLT  r3, r2     ;swap of max is smaller
    MOV    r0, r3     ;return max in r0
    BX     lr         ;return
}

int main (void)  {
  int x=9, y=4, z=12, maxNum;
  Init_serial();                    /* initialize the serial interface

  maxNum = myMax(x, y, z);          /*call embedded asm function
  printf ("The maximum of %d, %d and %d is %d\n", x, y, z, maxNum);
```

```
The maximum of 9, 4 and 12 is 12
```

Real-Time Agent: Target Stopped | Simulation | t1: 1.15324817 s

2. Below is a main program written in C. It calls a subroutine to append two strings into one. Write the subroutine in a separate ARM assembly file. Test the program using the Keil µVision 4 simulator.

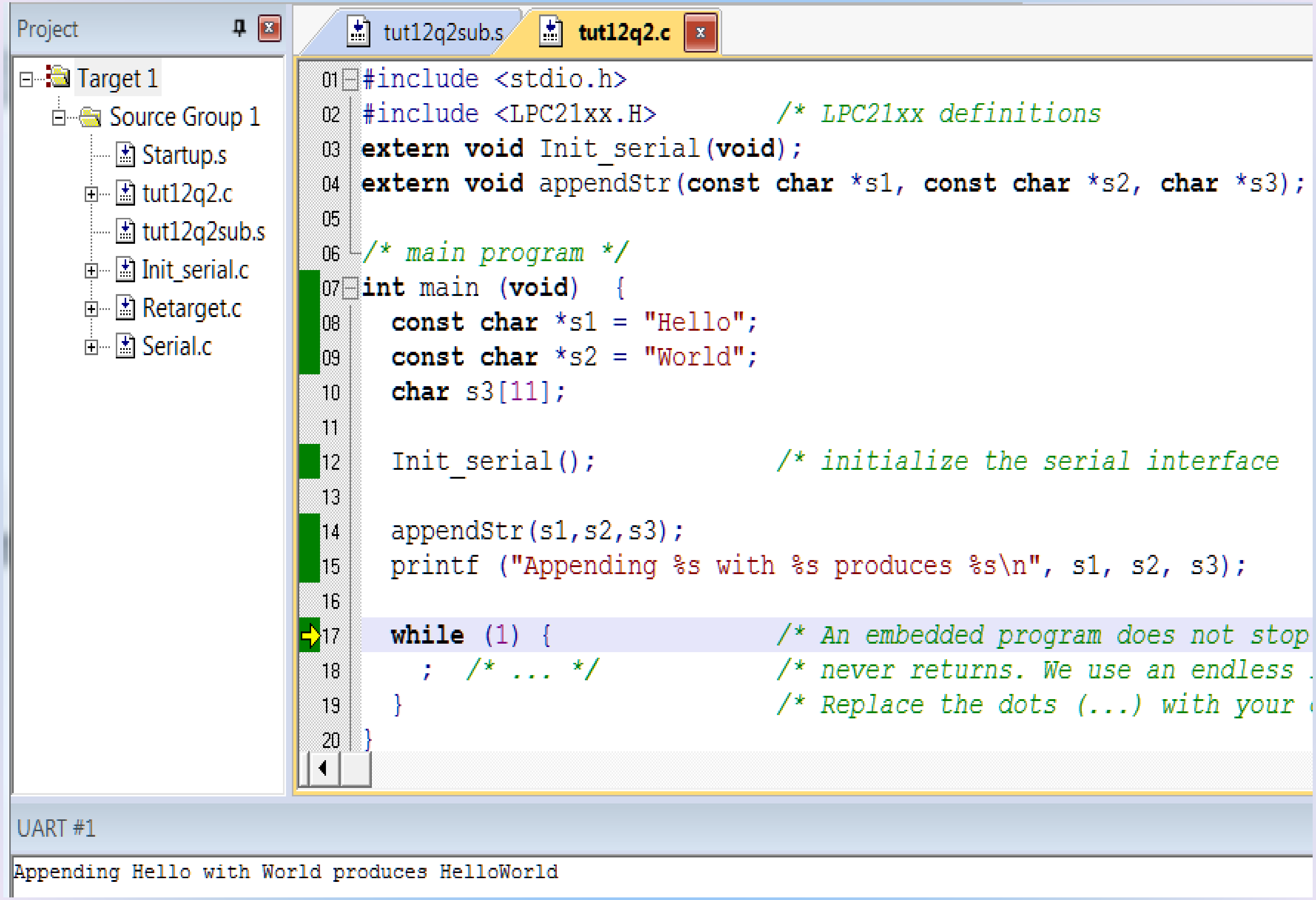
```c
#include <stdio.h>
#include <LPC21xx.H>            /* LPC21xx definitions */
extern void Init_serial(void);
extern void appendStr(const char *s1, const char *s2, char *s3);
```

```
/* main program */
int main (void)  {
  const char *s1 = "Hello";
  const char *s2 = "World";
  char s3[30];                          /*s3 should be “HelloWorld” eventually*/

  Init_serial();                                  /* initialize the serial interface  */
  appendStr(s1,s2,s3);                            /* call the asm subroutine          */
  printf ("Appending %s with %s produces %s\n", s1, s2, s3);

  while (1) {        /* An embedded program does not stop              */
     /* ... */       /* and never returns. We use an endless loop.     */
  }                  /* Replace the dots (...) with your own code.     */
}
```

Asm Subroutine

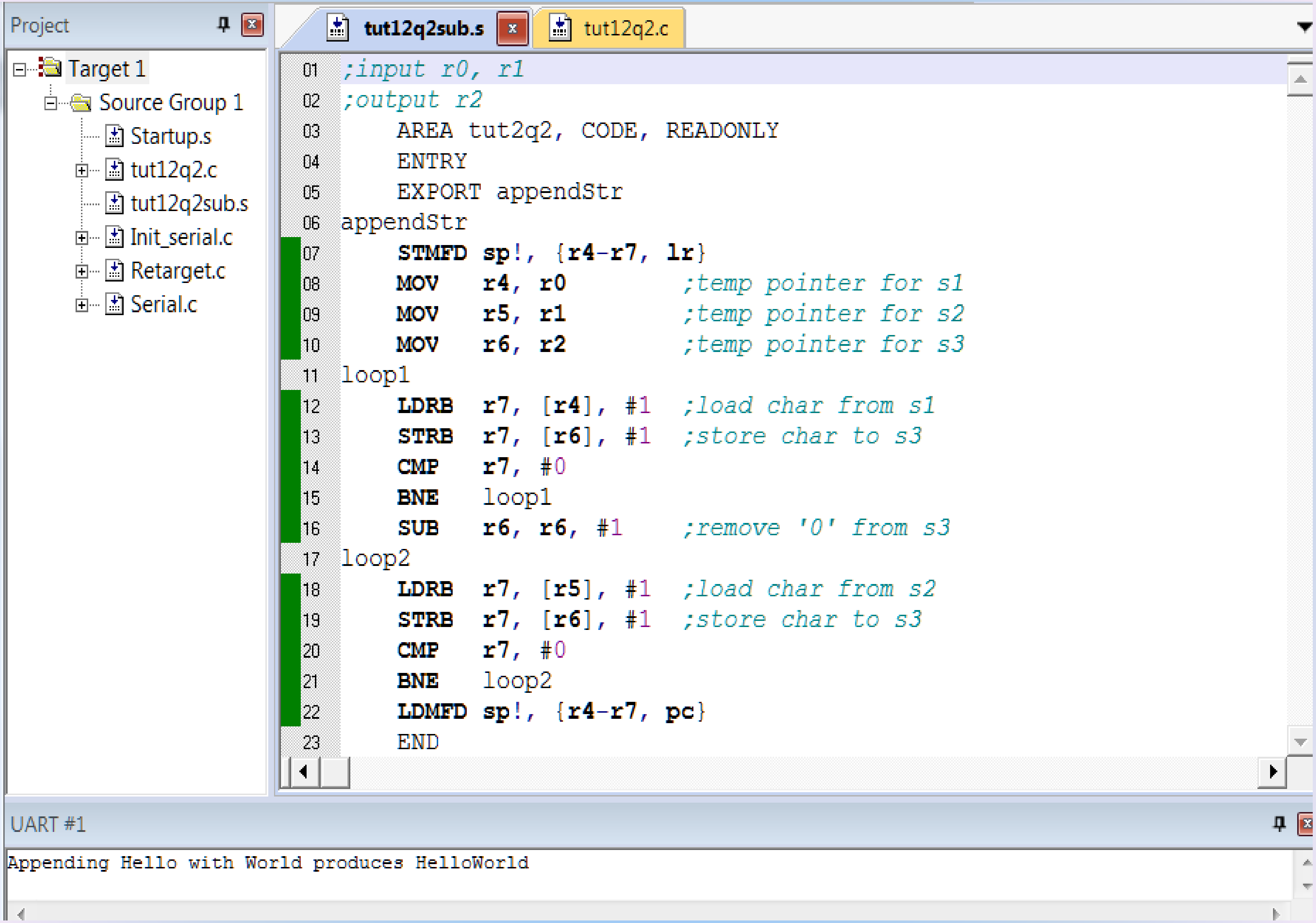
```
;input r0, r1
;output r2
        AREA tut2q2, CODE, READONLY
        ENTRY
        EXPORT appendStr
appendStr
        STMFD sp!, {r4-r7, lr}
        MOV   r4, r0            ;temp pointer for s1
        MOV   r5, r1            ;temp pointer for s2
        MOV   r6, r2            ;temp pointer for s3
```

```
loop1
        LDRB  r7, [r4], #1    ;load char from s1
        STRB  r7, [r6], #1    ;store char to s3
        CMP   r7, #0
        BNE   loop1
        SUB   r6, r6, #1      ;remove ‘0’ from s3
loop2
        LDRB  r7, [r5], #1    ;load char from s2
        STRB  r7, [r6], #1    ;store char to s3
        CMP   r7, #0
        BNE   loop2
        LDMFD sp!, {r4-r7, pc}
        END
```



```
#include <stdio.h>
#include <LPC21xx.H>          /* LPC21xx definitions
extern void Init_serial(void);
extern void appendStr(const char *s1, const char *s2, char *s3);

/* main program */
int main (void)  {
  const char *s1 = "Hello";
  const char *s2 = "World";
  char s3[11];

  Init_serial();              /* initialize the serial interface

  appendStr(s1,s2,s3);
  printf ("Appending %s with %s produces %s\n", s1, s2, s3);

  while (1) {                 /* An embedded program does not stop
    ;  /* ... */              /* never returns. We use an endless
  }                           /* Replace the dots (...) with your
}
```

```
Appending Hello with World produces HelloWorld
```



```
;input r0, r1
;output r2
    AREA tut2q2, CODE, READONLY
    ENTRY
    EXPORT appendStr
appendStr
    STMFD sp!, {r4-r7, lr}
    MOV   r4, r0        ;temp pointer for s1
    MOV   r5, r1        ;temp pointer for s2
    MOV   r6, r2        ;temp pointer for s3
loop1
    LDRB  r7, [r4], #1  ;load char from s1
    STRB  r7, [r6], #1  ;store char to s3
    CMP   r7, #0
    BNE   loop1
    SUB   r6, r6, #1    ;remove '0' from s3
loop2
    LDRB  r7, [r5], #1  ;load char from s2
    STRB  r7, [r6], #1  ;store char to s3
    CMP   r7, #0
    BNE   loop2
    LDMFD sp!, {r4-r7, pc}
    END
```

```
Appending Hello with World produces HelloWorld
```