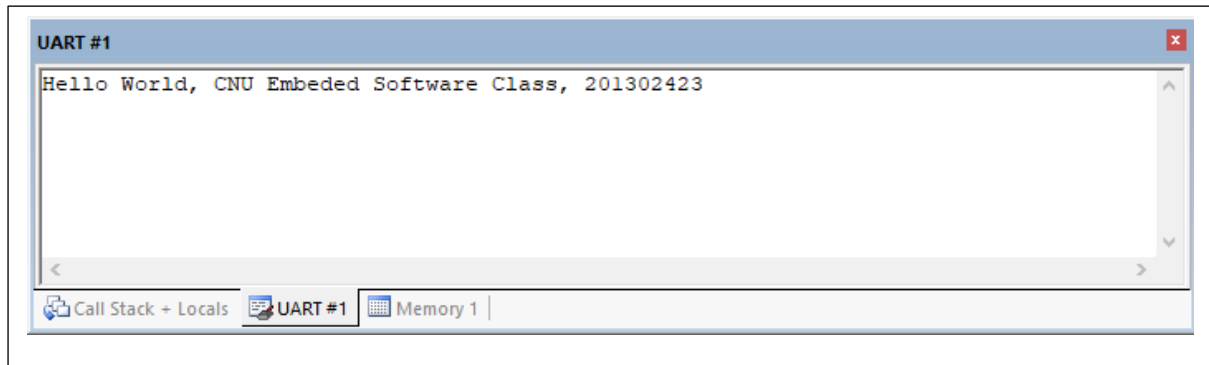


실습 1. 개발환경 익히기

문제 1. Hello.c파일을 열고 printf문을 수정하여 자신의 학번을 입력한 후 프로젝트를 실행하여, UART#1 창을 캡처하시오.



문제 2. 각 폴더의 역할이 무엇인지, 추측한 내용을 아래에 작성 하시오.

Startup Files:

프로그램이 실행될 때 메모리 초기화 하는 소스코드를 저장하는 폴더

System Files:

프로그램과 상호작용할 시스템에 맞게 설정하고 보레이트와 플로우 컨트롤 유무,어떤 시리얼 포트를 사용할지 선택하는 소스코드를 저장하는 폴더

Source Files:

실행하는 메인 C소스 코드 저장하는 폴더

Documentations:

해당 프로그램이 어떤 용도로 만들어 졌는지 설정값이 어떤지에 대한 설명한 텍스트파일을 저장하는 폴더

문제 3. 센서 측정 값을 확인하기 위해서는 Serial Port를 통해 baudrate를 맞춘 후에 값을 확인해야 한다. Serial.c파일의 SER_Init(void)함수를 참고하여 baudrate의 값을 쓰시오.

115200 baud

문제 4. Cortex-M3/M4의 프로세스 모드는 thread 모드와 handler 모드로 구성되어있다. thread 모드는 프로세서가 user application을 실행할 때의 동작 모드이고, handler 모드는 프로세서가 OS의 kernel을 실행하거나 인터럽트를 처리할 때의 동작 모드이다. Register 창을 참조하여 현재 프로세서의 모드가 무엇인지 쓰시오


thread 모드

문제 5. Cortex-M3/M4는 Privileged과 Unprivileged로 구성된 특권 단계를 갖는다. Privileged는 handler 모드와 thread 모드에서 프로세서가 취할 수 있는 상태이며, 모든 시스템 resource와 instruction을 사용할 수 있다. 또한 handler 모드와 thread 모드 사이의 모드 전환을 수행할 수 있다. Unprivileged는 thread 모드에서만 프로세서가 취할 수 있는 상태이며 여러 가지 제한을 가진다. 현재의 특권 수준을 쓰시오.

Privileged

문제 6. Cortex-M3/M4는 MSP(Main Stack Pointer)와 PSP(Process Stack Pointer)라는 2개의 stack 포인터를 갖는다. MSP는 default stack pointer이지만, 프로세서의 상태가 Privileged일 때만 사용 가능하다. PSP는 user application에 의해서 사용되기 때문에 thread 모드에서만 사용할 수 있다. 현재의 스택의 종류를 쓰시오.

MSP

문제 7. R15(PC)는 Program Counter로 다음에 수행이 되어야 할 명령어의 주소를 가리킨다. Debug 상태에서  (Reset)버튼을 누른 후 PC값이 얼마인지 쓰시오.

0x08000100

문제 8. View 메뉴의 Memory 윈도우 Memory 1을 열고, On-Chip IROM1 영역의 시작 주소를 0x 형태로 입력하면, 메모리 내용을 볼 수 있다. 0x8000100주소를 입력하여 데이터를 2바이트씩 4개의 데이터를 쓰시오. Little Endian이라는 점에 주의하시오.

주소: 0x8000100	데이터: 0x4806
주소: 0x8000102	데이터: 0x4780
주소: 0x8000104	데이터: 0x4806
주소: 0x8000106	데이터: 0x4700

문제 9. Hello 프로젝트를 실행하면, startup_stm32f10x_md.s 파일에서 프로그램이 시작해서 Hello.c 의 main 까지 프로그램이 진행된다. 프로그램의 시작부터 Hello 메시지를 출력하기 까지 프로그램의 실행과정을 조사해서 진행과정을 설명하시오.

startup_stm32f10x_md.s에서 SystemInit을 Import해서 System_stm32f10x.c 시스템파일로 넘어가서 SystemInit()함수가 실행된다.

SystemInit()함수는 실행되기전에 사용되는 메모리값들을 초기화하는 함수이다.

그다음 시스템시간을 설정하는 SetSysClockTo72함수가 실행

다음 다시 Hello.c의 Main의 SER_Init이 실행되어서 serial interface의 초기화가 시작되어서 baud레이트와 데이터 크기,플로우 컨트롤들을 설정한후

```
0x08000424 F7FFF8E BL.W SER_Init (0x08000344)
28: printf ("Hello World, CNU Embeded Software Class, 201302423\n");
29:
0x08000428 A002 ADR r0,{pc}+4 ; @0x08000434
0x0800042A F000F82F BL.W __Oprintf$bare (0x0800048C)
```

위 그림과 같은 명령에 의해 프린트문이 실행되고 Hello World가 출력된다.