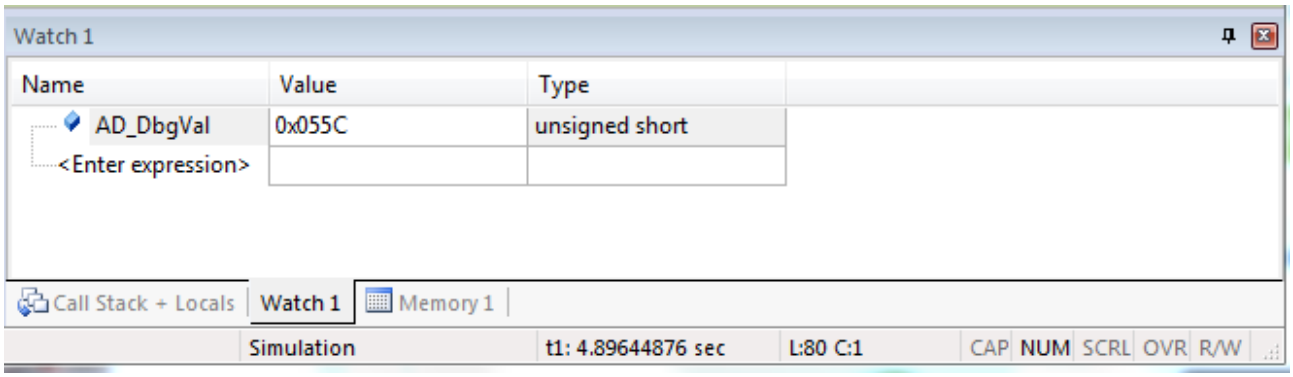
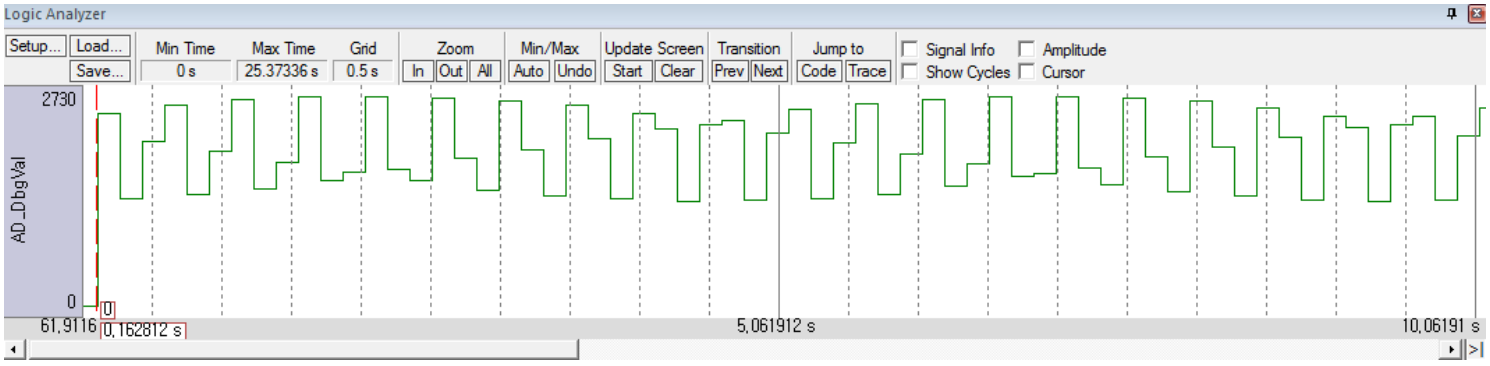
HW2
과목 : 임베디드 프로세서 응용(이종원 교수님)
제출일 : 2014.03.26 수요일
제출자 : 김주은 (21000172)

1. **P. 33에 있는 것과 같이 watch window를 동작시키고, watch window를 캡쳐하여 보고서에 포함 시키고, watch window가 무엇을 하는 것인지 설명하라.**



watch window : microcontroller ADC에 전원을 걸어주면 variable이 업데이팅 되면서 application code 로 변환된 값을 보여준다.

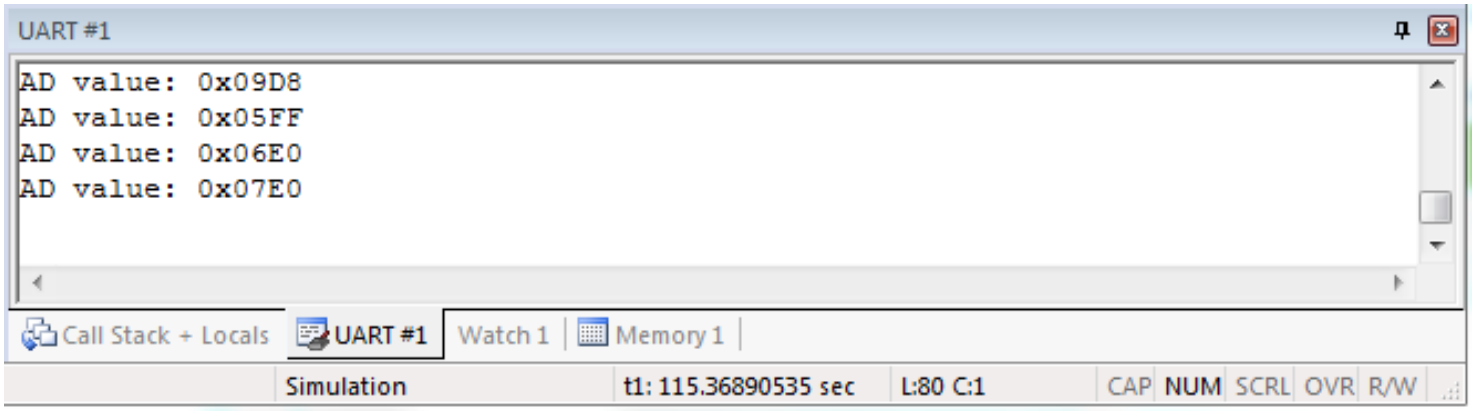
2. **P. 34에 나타난 것과 같은 AD_DbgVal에 대한 logic Analyzer window를 캡쳐하여 보고서 에 포함시켜라.**



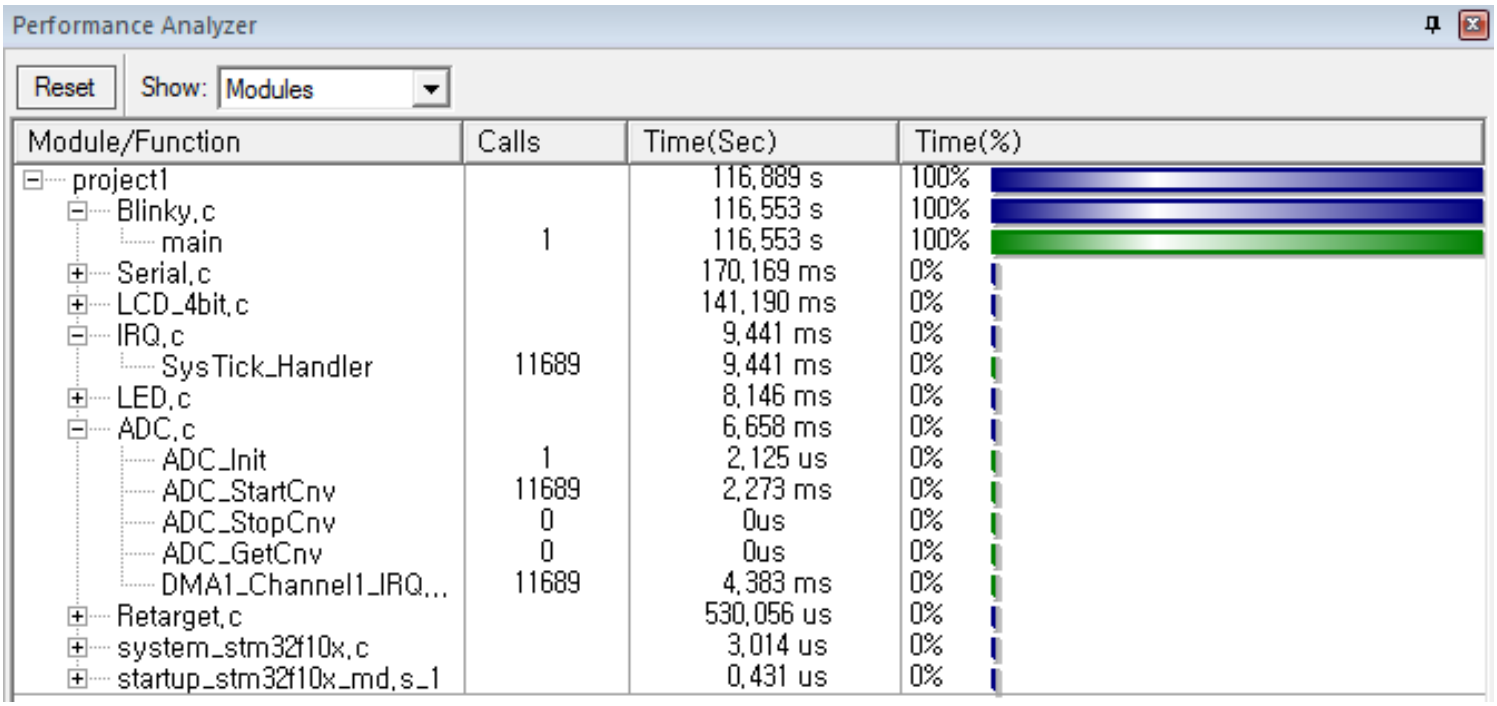
3. **35와 같이 Peripheral -> General Purpose I/O ->GPIOB 에 대한 window를 열고, window에서 Pins의 값이 어떻게 변하는 가를 기술하여라.**

8bit에서 15bit쪽으로(오른쪽에서 왼쪽으로) 한칸씩 이동하면서 값이 변하는 것을 볼수있다. shift left 하면서 pin값이 증가하는 것을 볼 수 있다 .
(0x00000100 ->0x00000200 -> 0x00000400->0x00000800....)

4. **P. 36에서 나타난 바와 같이 View -> Serial Windows -> UART #1의 창을 열고 UART #1 창을 캡쳐하여 보고서에 포함시켜라.**



5. **P. 37에 나타난 바와 같이 View -> Analysis Windows -> Performance Analyzer window 를 열고 Performance Analyzer window를 캡쳐하고 그 내용을 분석하여 설명하라.**



Module/Function	Calls	Time(Sec)	Time(%)
project1		116,889 s	100%
Blinky.c		116,553 s	100%
main	1	116,553 s	100%
Serial.c		170,169 ms	0%
LCD_4bit.c		141,190 ms	0%
IRQ.c		9,441 ms	0%
SysTick_Handler	11689	9,441 ms	0%
LED.c		8,146 ms	0%
ADC.c		6,658 ms	0%
ADC_Init	1	2,125 us	0%
ADC_StartCnv	11689	2,273 ms	0%
ADC_StopCnv	0	0us	0%
ADC_GetCnv	0	0us	0%
DMA1_Channel1_IRQ...	11689	4,383 ms	0%
Retarget.c		530,056 us	0%
system_stm32f10x.c		3,014 us	0%
startup_stm32f10x_md.s_1		0,431 us	0%

설명 : Blinky.c 파일의 main() 함수에서 1번의 호출(call)이 일어났으며, Time 에서는 Cumulative Runtime을 나타낸다. Blinky.c에서는 116.553s의 cumulative runtime이 걸렸다. 즉 Call은 함수호출 횟수를, Time은 Cumulative runtime을 나타낸다.

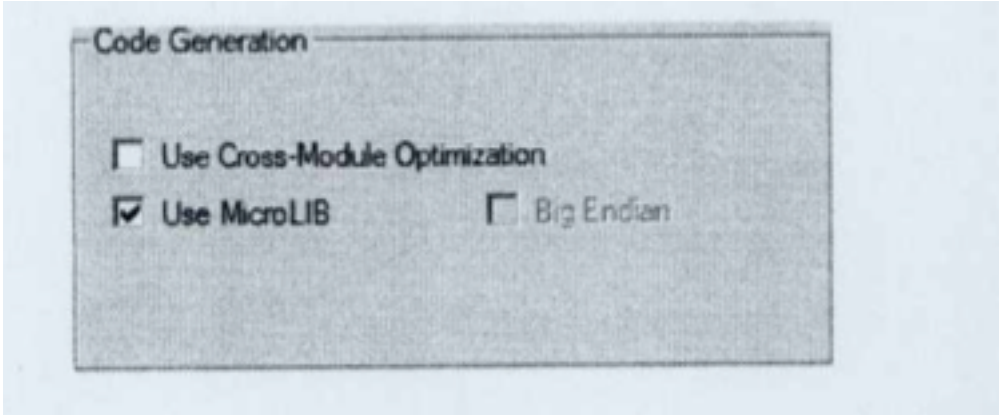
6. **Keil MDK-ARM compiler가 제공하는 ANSI library 종류는 어떠한 것이 있으며,**

각각의 특징은 무엇인가? 각 library를 사용하기 위해서는 어떻게 하여야 하는가 설명하라.

Keil MDK-ARM에는 두가지 ANSI library set이 있다.
첫번째는 **standard library**로, 현재 ANSI 기준에 완전히 부합하며
microcontroller 사용을 위한 큰 용량의 코드 공간을 가진다.

두번째로 **Keil MicroLIB**이다. 이것은 초기 ANSI 기준의 C99으로 쓰여졌다.
이 버전의 ANSI 기준은 microcontroller 사용자의 요구에 충족하도록 맞춰있다.
ARM compiler library 보다 50% 이상의 코드 공간을 절약할 수 있어
가능하다면 사용해 보는 것도 좋지만, 모든 함수를 제공하지 않으며 정확도도 낮다.

library 사용하기
Library 선택은 project-options for Target 'Target1'을 선택하여 사용할 수 있다.



7. **pp. 42-44에서 동작을 참조하여, 본 프로젝트에서 RAM에서 HEAP과 STACK의 메모리 영역을 기술하시고, 사용하는 total ROM 크기와 RAM 크기를 표시하시오.**

Total ROM Size (Code + RO Data + RW Data) 4244 (4.14kB)
RAM 사이즈(RW Data + ZI Data) : 1696 (1.66 kB)

Execution Region ER_RW (Base: 0x20000000, Size: 0x00000034, Max: 0xffffffff, ABSOLUTE)

Base Addr	Size	Type	Attr	Idx	E Section Name	Object
0x20000000	0x00000004	Data	RW	13	.data	adc.o
0x20000004	0x00000002	Data	RW	61	.data	blinky.o
0x20000006	0x00000002	PAD				
0x20000008	0x0000000d	Data	RW	103	.data	irq.o
0x20000015	0x00000003	PAD				
0x20000018	0x00000008	Data	RW	174	.data	retarget.o
0x20000020	0x00000014	Data	RW	218	.data	system_stm32f10x.o

Execution Region ER_ZI (Base: 0x20000034, Size: 0x0000066c, Max: 0xffffffff, ABSOLUTE)

Base Addr	Size	Type	Attr	Idx	E Section Name	Object
0x20000034	0x0000000a	Zero	RW	60	.bss	blinky.o
0x2000003e	0x00000002	PAD				
0x20000040	0x00000060	Zero	RW	354	.bss	c_w.l(libspace.o)
0x200000a0	0x00000200	Zero	RW	2	HEAP	startup_stm32f10x_md.o
0x200002a0	0x00000400	Zero	RW	1	STACK	startup_stm32f10x_md.o

```
================================================================================

   Total RO  Size (Code + RO Data)                 4192 (   4.09kB)
   Total RW  Size (RW Data + ZI Data)              1696 (   1.66kB)
   Total ROM Size (Code + RO Data + RW Data)       4244 (   4.14kB)

================================================================================
```

8. Scatter file이란 무엇인가?

-scatter file이란, contoller의 메모리 레이아웃을 어떻게 구성할것인가 결정한다.
구체적인 메모리 공간에 대상을 할당하고,
실행할 영영에 대치시킬 load region을 결정한다.

(The scatter file determines how the memory layout of your controller is organized.
In essence, you can allocate objects to specific memory regions, determine the mapping of load regions to execution regions etc)