

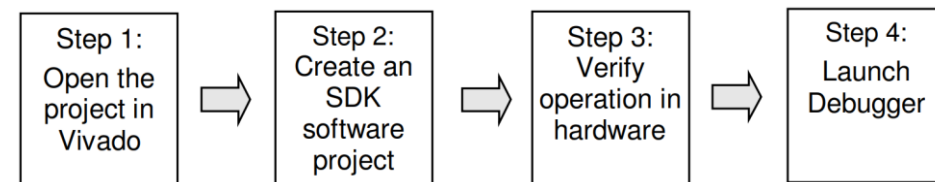
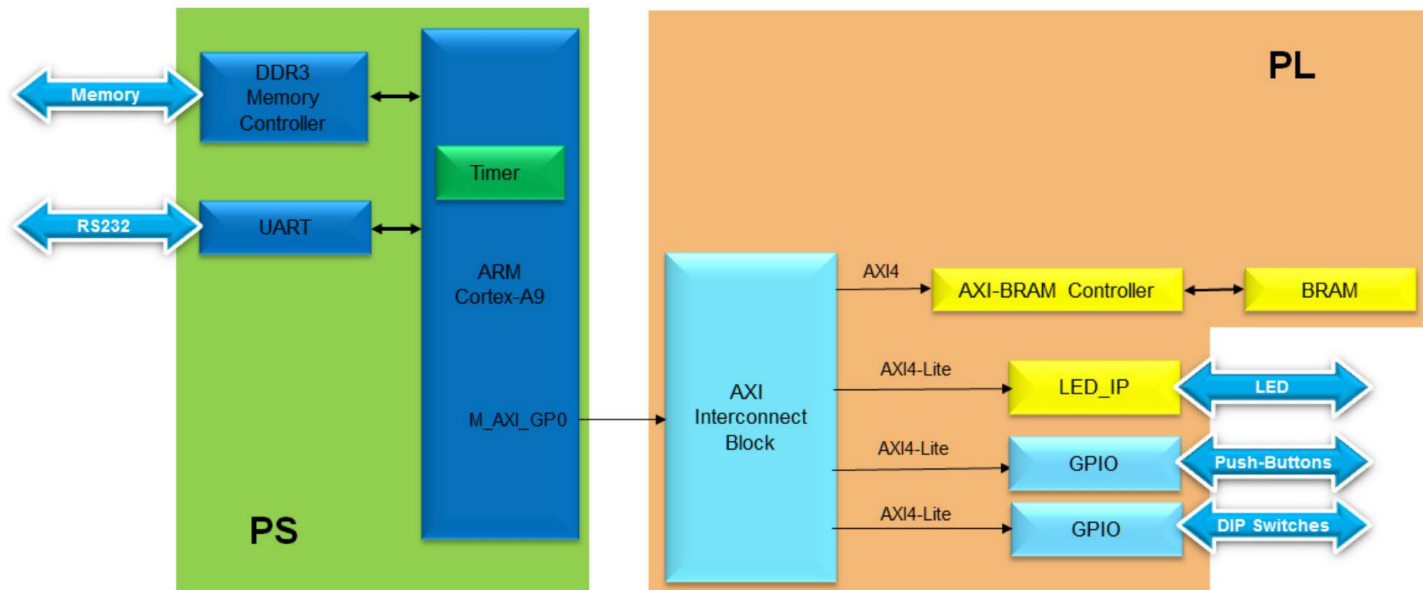
# FPGA embedded system design using AXI

With Vivado

27<sup>th</sup> March 2023

Judong Park

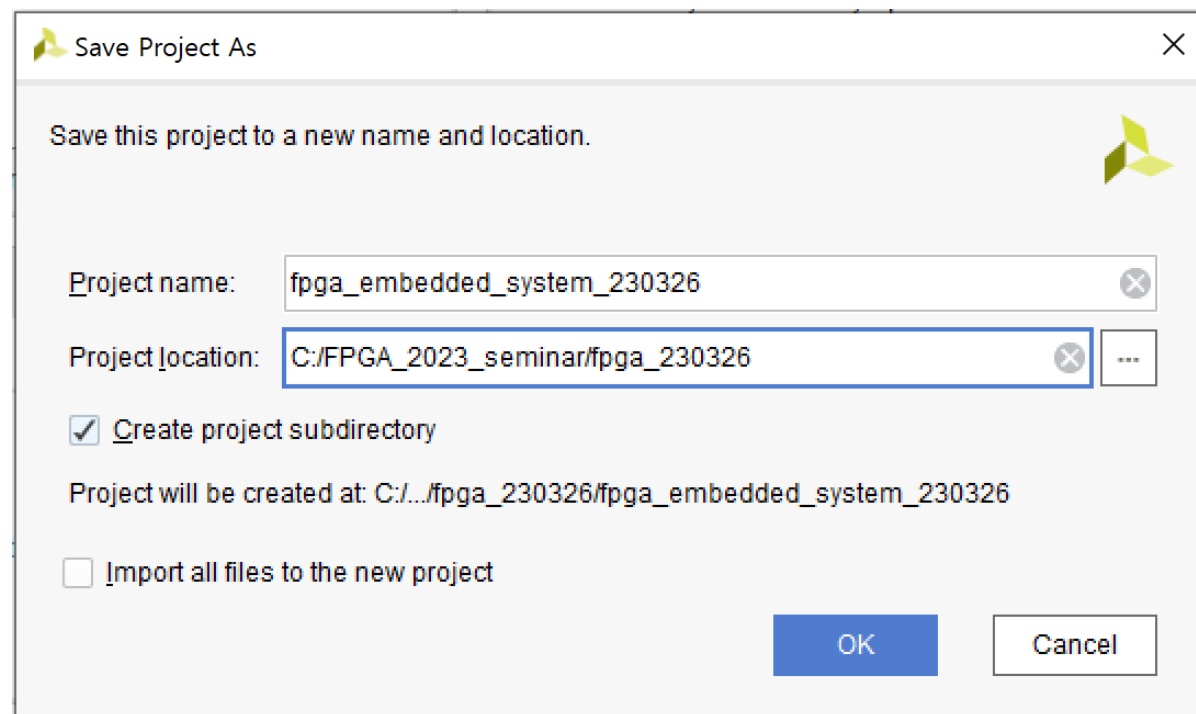
# 이번 시간 만들 design



- 개요: SDK를 활용한 ARM Cortex-A9 timer 코딩
- 학습 목표:
  - Timer 활용
  - SDK debugger 활용
    - ➔ 코드 결과에 따른 변수, 메모리 값 확인

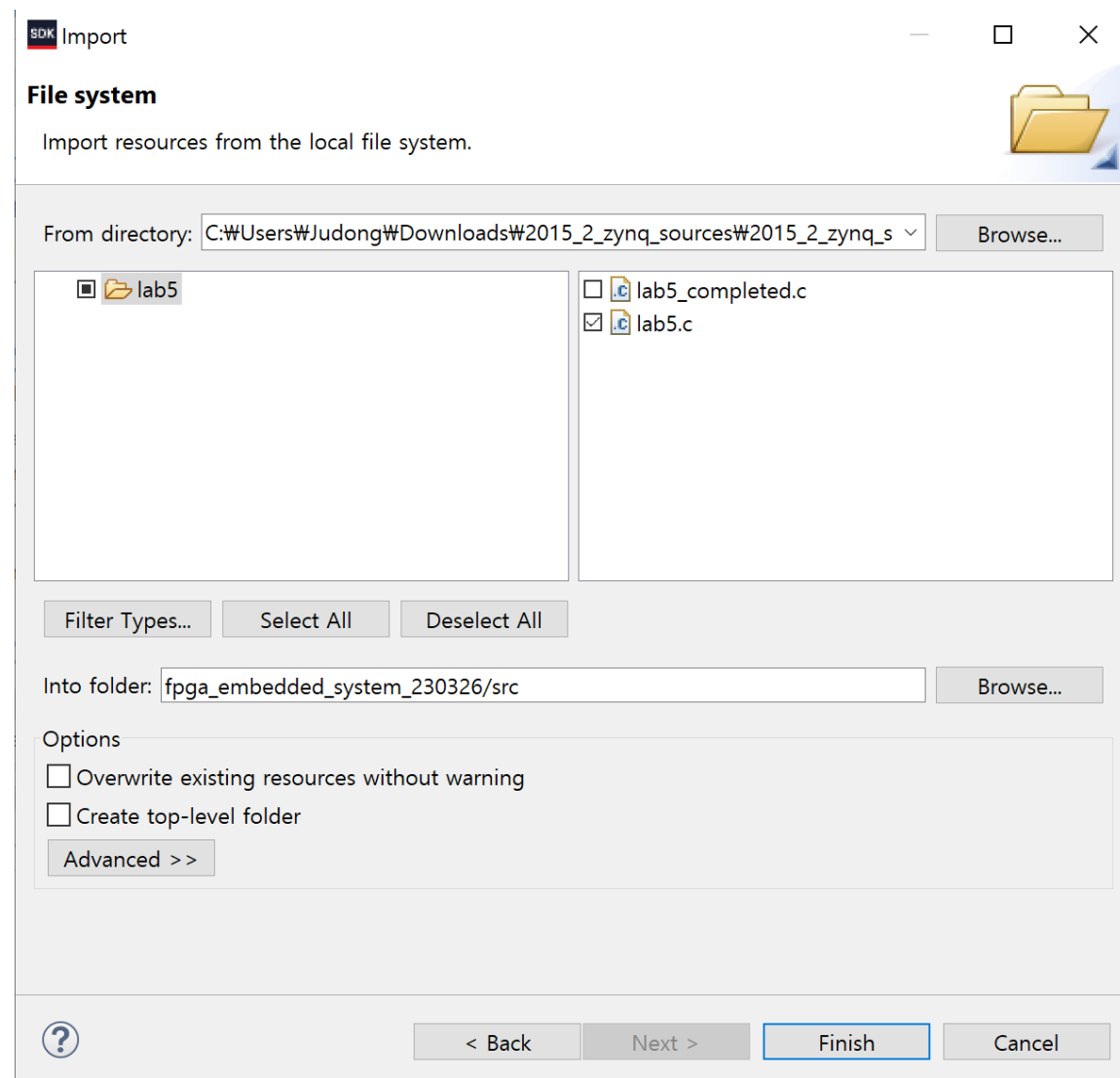
# Open the Project in Vivado

- 프로젝트 불러오기
  - 비바도 실행 및 이전 프로젝트 실행
    - Project: **fpga\_embedded\_system\_230313**
    - **fpga\_embedded\_system\_230327**
    - 저장 위치 확인
  - Sdk 실행
    - File – launch SDK



# Create an SDK Software Project

- 프로젝트 생성
  - File – new – application project
  - Project name: **fpga\_embedded\_system\_230327**
  - 템플릿: **empty application**
  - Finish 클릭
- 프로젝트 파일 추가
  - **fpga\_embedded\_system\_230327 – src – import**
    - General – file system 더블 클릭
    - Main.c 추가



# Refer to the Scutimer API documentation

- Scutimer API 문서 참조하기
    - fpga\_embedded\_system\_230327\_bsp-system.mss
      - File lists – xscutimer.h 클릭
        - XScuTimer\_LookupConfig( )
        - XScuTimer\_CfgInitialize( )
- ➔ Timer 사용 전에 미리 사용해야하는 API 함수

## Functions

XScuTimer_Config *	<b>XScuTimer_LookupConfig</b> (u16 DeviceId) Lookup the device configuration based on the unique device ID. <a href="#">More...</a>
s32	<b>XScuTimer_SelfTest</b> (XScuTimer *InstancePtr) Run a self-test on the timer. <a href="#">More...</a>
s32	<b>XScuTimer_CfgInitialize</b> (XScuTimer *InstancePtr, XScuTimer_Config *ConfigPtr, u32 EffectiveAddress) Initialize a specific timer instance/driver. <a href="#">More...</a>
void	<b>XScuTimer_Start</b> (XScuTimer *InstancePtr) Start the timer. <a href="#">More...</a>
void	<b>XScuTimer_Stop</b> (XScuTimer *InstancePtr) Stop the timer. <a href="#">More...</a>
void	<b>XScuTimer_SetPrescaler</b> (XScuTimer *InstancePtr, u8 PrescalerValue) This function sets the prescaler bits in the timer control register. <a href="#">More...</a>
u8	<b>XScuTimer_GetPrescaler</b> (XScuTimer *InstancePtr) This function returns the current prescaler value. <a href="#">More...</a>

## Macros

#define	<b>XScuTimer_IsExpired</b> (InstancePtr) Check if the timer has expired. <a href="#">More...</a>
#define	<b>XScuTimer_RestartTimer</b> (InstancePtr) Re-start the timer. <a href="#">More...</a>
#define	<b>XScuTimer_LoadTimer</b> (InstancePtr, Value) Write to the timer load register. <a href="#">More...</a>
#define	<b>XScuTimer_GetCounterValue</b> (InstancePtr) Returns the current timer counter register value. <a href="#">More...</a>
#define	<b>XScuTimer_EnableAutoReload</b> (InstancePtr) Enable auto-reload mode. <a href="#">More...</a>
#define	<b>XScuTimer_DisableAutoReload</b> (InstancePtr) Disable auto-reload mode. <a href="#">More...</a>
#define	<b>XScuTimer_EnableInterrupt</b> (InstancePtr) Enable the Timer interrupt. <a href="#">More...</a>
#define	<b>XScuTimer_DisableInterrupt</b> (InstancePtr) Disable the Timer interrupt. <a href="#">More...</a>
#define	<b>XScuTimer_GetInterruptStatus</b> (InstancePtr) This function reads the interrupt status. <a href="#">More...</a>
#define	<b>XScuTimer_ClearInterruptStatus</b> (InstancePtr) This function clears the interrupt status. <a href="#">More...</a>

# Correct the errors

- 코드 에러 확인 및 디버깅
  - Problems – Errors – 오류 내용 확인
  - 에러 디버깅: Timer 관련 드라이버 명령어 관련 오류  
→ **#include "xscutimer.h"** 추가

- 코드 추가: timer 초기화

- 코드:

```
ConfigPtr = XScuTimer_LookupConfig  
(XPAR_PS7_SCUTIMER_0_DEVICE_ID);  
Status = XScuTimer_CfgInitialize (TimerInstancePtr,  
ConfigPtr, ConfigPtr->BaseAddr);  
if(Status != XST_SUCCESS){  
    xil_printf("Timer init() failed\r\n");  
    return XST_FAILURE;  
}
```

```
// PS Timer related definitions
```

```
XScuTimer_Config *ConfigPtr;  
XScuTimer *TimerInstancePtr = &Timer;
```

<에러 부분>

```
// Initialize the timer
```

```
ConfigPtr = XScuTimer_LookupConfig (XPAR_PS7_SCUTIMER_0_DEVICE_ID);  
Status = XScuTimer_CfgInitialize (TimerInstancePtr, ConfigPtr, ConfigPtr->BaseAddr);  
if(Status != XST_SUCCESS){  
    xil_printf("Timer init() failed\r\n");  
    return XST_FAILURE;  
}
```

<timer 초기화>

# Correct the errors

- 코드 추가: timer 동작

```
// Load timer with delay in multiple of ONE_TENTH
XScuTimer_LoadTimer(TimerInstancePtr,
ONE_TENTH*dip_check_prev);
// Set AutoLoad mode
XScuTimer_EnableAutoReload(TimerInstancePtr);
// Start the timer
XScuTimer_Start (TimerInstancePtr);
```

- 코드 추가: led 동작

```
// clear status bit
XScuTimer_ClearInterruptStatus(TimerInstancePtr);
// output the count to LED and increment the count
LED_IP_mWriteReg(XPAR_LED_IP_0_S_AXI_BASEADDR, 0, count);
count++;
```

```
// Load timer with delay in multiple of ONE_TENTH
XScuTimer_LoadTimer(TimerInstancePtr, ONE_TENTH*dip_check_prev);
// Set AutoLoad mode
XScuTimer_EnableAutoReload(TimerInstancePtr);
// Start the timer
XScuTimer_Start (TimerInstancePtr);
```

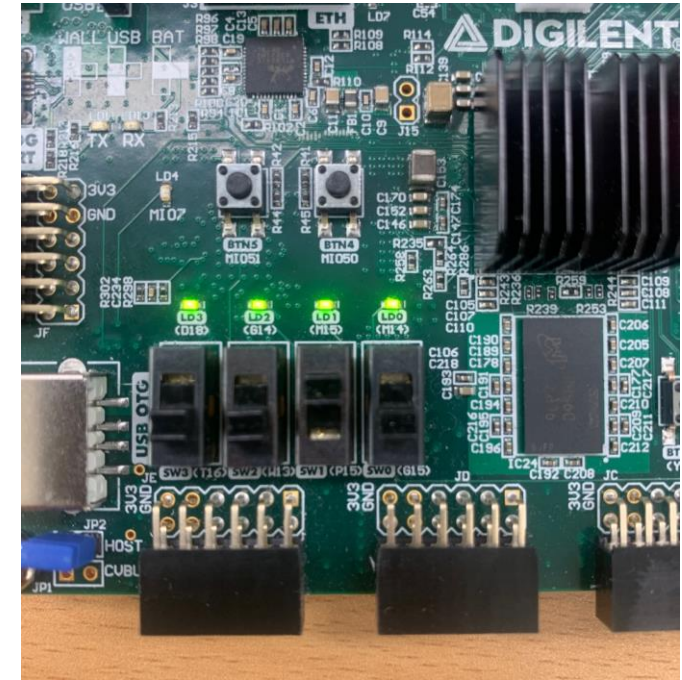
< timer 동작 >

```
// load timer with the new switch settings
XScuTimer_LoadTimer(TimerInstancePtr, ONE_TENTH*dip_check);
count = 0;
}
if(XScuTimer_IsExpired(TimerInstancePtr)) {
    // clear status bit
    XScuTimer_ClearInterruptStatus(TimerInstancePtr);
    // output the count to LED and increment the count
    LED_IP_mWriteReg(XPAR_LED_IP_0_S_AXI_BASEADDR, 0, count);
    count++;
}
```

< led 동작 >

# Verification & programming

- FPGA에 코드 프로그래밍
  - **FPGA Program** 클릭: FPGA에 bitstream 업로드
  - 생성한 fpga\_ps\_230327 프로젝트 우클릭
  - **Build project**
  - **Sdk 터미널 연결**
    - Port: 장치관리자에서 FPGA와 연결된 포트 확인
  - 생성한 fpga\_ps\_230327 프로젝트 우클릭
  - Run as – **launch on hardware (GDB)**
- 동작 확인
  - 터미널 콘솔창에서 결과 확인 가능



DIP Switch Status 2, 0  
DIP Switch Status 0, 1  
DIP Switch Status 1, 3  
DIP Switch Status 3, 2  
DIP Switch Status 2, 0




# Launch Debugger and debug

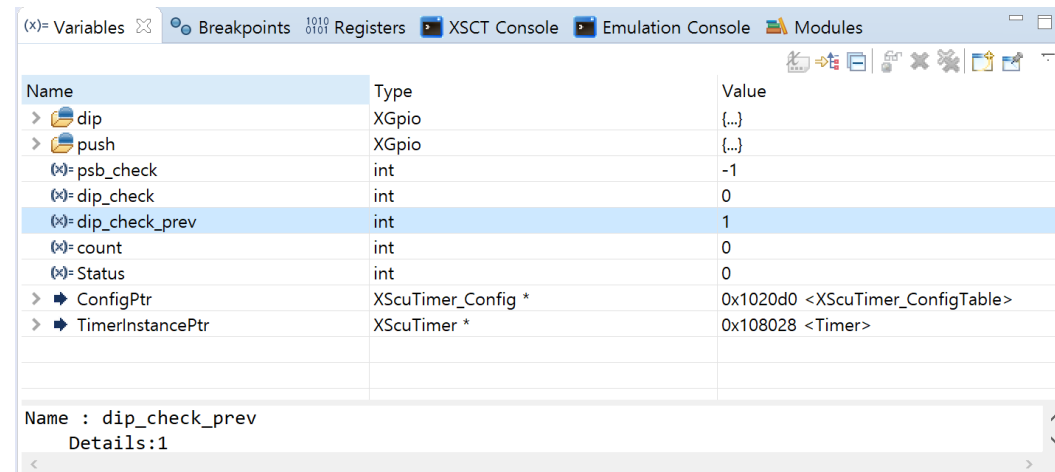
- Debugger 사용
  - Run - Debug as ... - launch on hardware (GDB)  
(➔ Elf 파일 생성)
  - Show line numbers: line 표시
  - Add breakpoint: 특정 line에서 코드 컴파일 일시정지
- Resume 버튼: 활성화된 breakpoint **이전까지** 컴파일
  - 1번째 눌렀을 때: count > 0 (ex. 1055712)
- Step over 버튼 (F6): 코드 한 줄 컴파일
  - 1번째 눌렀을 때: count = 0
- Resume – step over 한번 더 반복하며 변화 관찰

```
30 | count = 0;
31
32 // Initialize the timer
33 ConfigPtr = XScuTimer_LookupConfig (XPAR_PS7_S
34     Status = XScuTimer_CfgInitialize (TimerIns
35     if(Status != XST_SUCCESS){
36         xil_printf("Timer init() failed\r\n");
37         return XST_FAILURE;
38     }
39 // Read dip switch values
40 dip_check_prev = XGpio_DiscreteRead(&dip, 1);
```

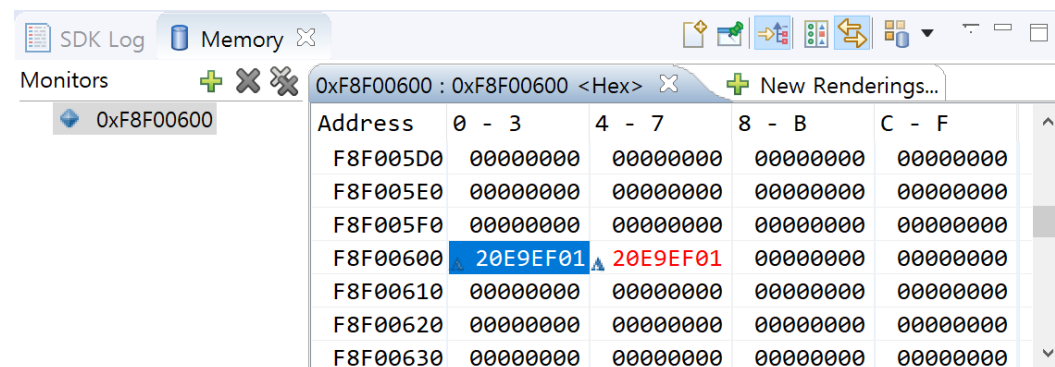
< breakpoint 지점 >

# Launch Debugger and debug

- 사용되는 Memory 모니터링
  - Window – show view – memory
  - 다음 버튼 클릭 
  - Private counter load register 주소 입력
    - ➔ 0xF8F00600
    - 어디서 확인이 가능할까?
    - ➔ PS timer base 주소, timer 오프셋 필요
    - ➔ Timer Base 주소: xparameters.h 파일에서 확인 가능!  
(xpar\_ps7\_scutimer\_0\_baseaddr 검색)
    - ➔ Timer 오프셋: xscutimer.h – xscutimer\_hw.h –  
xscutimer\_load\_offset 확인
- Step over 클릭
  - ➔ Timer 레지스터 load 시, 주소값 변화
    - Switch = 1 ➔ 0x20E9EF01



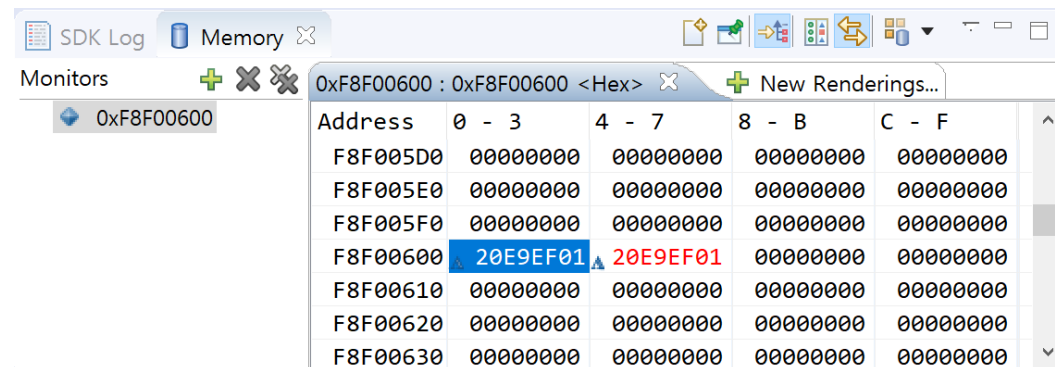
< variables >



< timer 메모리 모니터링 >

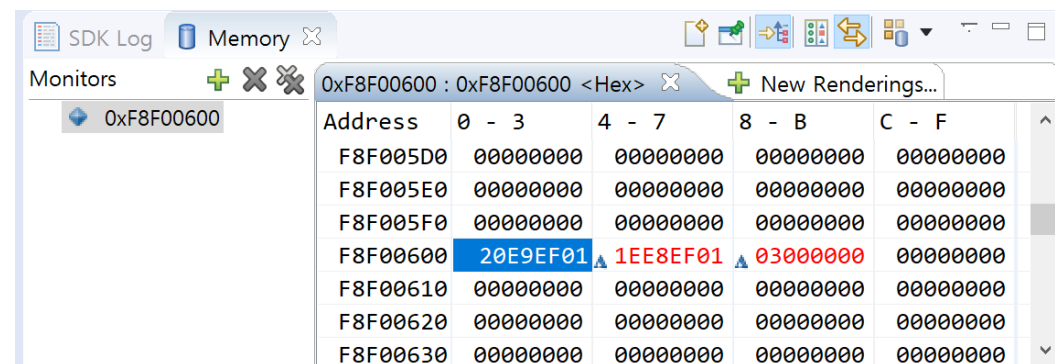
# Launch Debugger and debug

- LED 모니터링
  - Resume 클릭
    - LED 변화 X
    - Timer counter 시작 → Counter 레지스터 값 변화 O
  - Step over 클릭
    - Count = 0 → LED 꺼짐
  - Resume 클릭
    - LED 변화 O
  - Push 버튼 클릭
    - 4번째 breakpoint에서 프로그램 정지
    - 스위치에 따른 led 출력 변화
    - Timer/control 레지스터 변화(빨간색)
- 종료: terminate 클릭
- Sdk 종료
- 보드 전원 off



Memory window showing the initial state of memory addresses F8F005D0 to F8F00630. The value at address F8F00600 is 20E9EF01.

Address	0 - 3	4 - 7	8 - B	C - F
F8F005D0	00000000	00000000	00000000	00000000
F8F005E0	00000000	00000000	00000000	00000000
F8F005F0	00000000	00000000	00000000	00000000
F8F00600	20E9EF01	20E9EF01	00000000	00000000
F8F00610	00000000	00000000	00000000	00000000
F8F00620	00000000	00000000	00000000	00000000
F8F00630	00000000	00000000	00000000	00000000

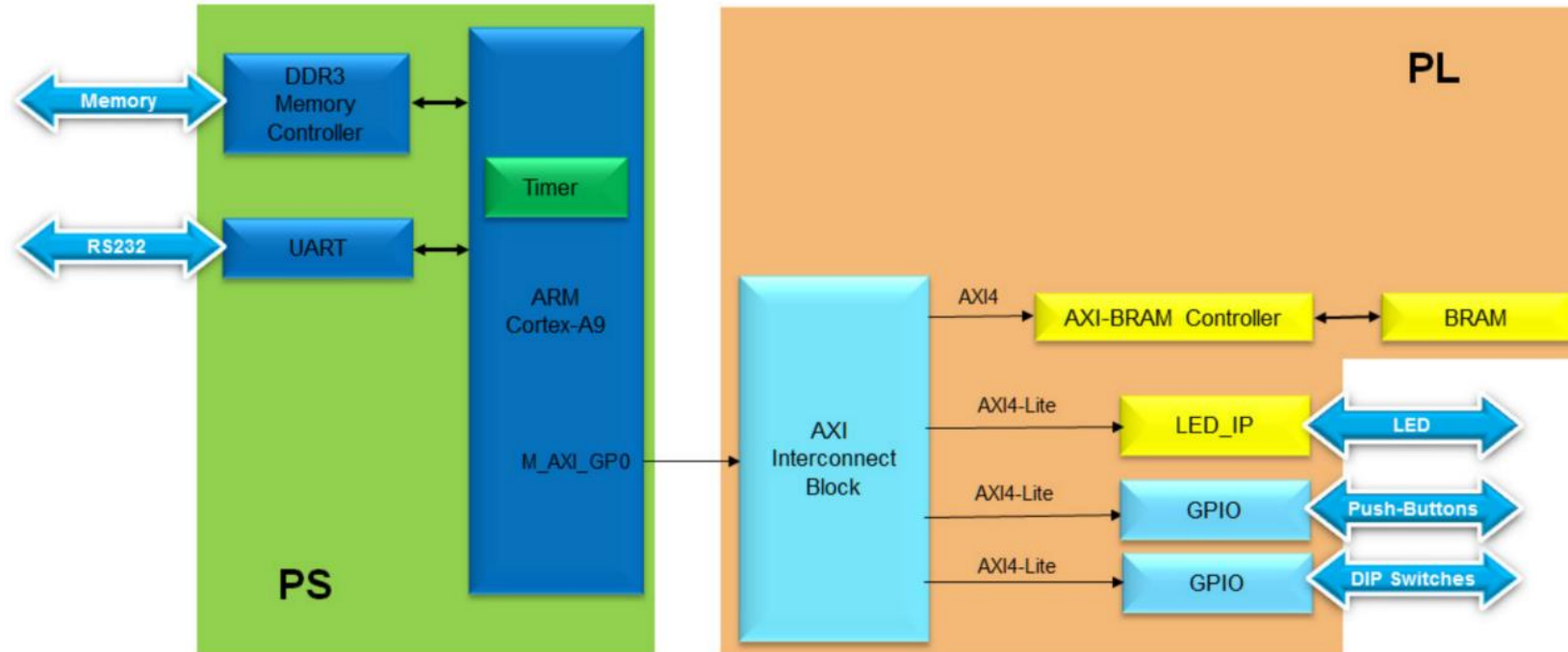



Memory window showing the state of memory addresses F8F005D0 to F8F00630 after a push button click. The value at address F8F00600 has changed to 1EE8EF01, and the value at address F8F00608 has changed to 03000000.

Address	0 - 3	4 - 7	8 - B	C - F
F8F005D0	00000000	00000000	00000000	00000000
F8F005E0	00000000	00000000	00000000	00000000
F8F005F0	00000000	00000000	00000000	00000000
F8F00600	20E9EF01	1EE8EF01	03000000	00000000
F8F00610	00000000	00000000	00000000	00000000
F8F00620	00000000	00000000	00000000	00000000
F8F00630	00000000	00000000	00000000	00000000

< control register[8-B] >

# Conclusion



- 실습 내용 정리:
  - ARM Cortex-A9 timer를 활용한 sdk 코딩
  - 기능 구현을 위해 코드 추가
  - SDK debugger, breakpoint 활용
    - ➔ 코드의 변수, memory 값 확인

# 감사합니다!

- Q&A