Xilinx Zynq FPGA, TI DSP MCU 기반의

프로그래밍 및 회로 설계 전문가

강사 이상훈

(Innova Lee)

Gcccompil3r@gmail.com

학생 김민호

minking12@naver.com

1. mov

```
int main()
{

   register unsigned int r0 asm("r0")=0;
   register unsigned int r1 asm("r1")=0;
   register unsigned int r2 asm("r2")=0;
   register unsigned int r3 asm("r3")=0;
   register unsigned int r4 asm("r4")=0;
   register unsigned int r5 asm("r5")=0;

   asm volatile("mov r0,#0xff,8");//r0 에 0xff 00 00 00

   printf("r0=0x%x\n",r0);

   return 0;
}
```

2. add

```
int main()
{

   register unsigned int r0 asm("r0")=0;
   register unsigned int r1 asm("r1")=0;
   register unsigned int r2 asm("r2")=0;
   register unsigned int r3 asm("r3")=0;
   register unsigned int r4 asm("r4")=0;
   register unsigned int r5 asm("r5")=0;

   asm volatile("mov r0,#0xff,8");
   asm volatile("mov r1,#0xf");
   asm volatile("add r2,r1,r0");//r1 + r2 값을 r2 에 저장

   printf("r2=0x%x\n",r2);
   return 0;
}
```

3. lsl

```
int main()
{

   register unsigned int r0 asm("r0")=0;
   register unsigned int r1 asm("r1")=0;
   register unsigned int r2 asm("r2")=0;
   register unsigned int r3 asm("r3")=0;
   register unsigned int r4 asm("r4")=0;
   register unsigned int r5 asm("r5")=0;

   asm volatile("mov r1,#7");
   asm volatile("mov r2,#3");
   asm volatile("add r0,r1,r2,lsl #7");//3*2^7 연산후에 7 을 합하고 r0 에 저장.

   printf("r0=0x%x\n",r0);

   return 0;
}
```

4.lsl 2

```
int main()
{

   register unsigned int r0 asm("r0")=0;
   register unsigned int r1 asm("r1")=0;
   register unsigned int r2 asm("r2")=0;
   register unsigned int r3 asm("r3")=0;
   register unsigned int r4 asm("r4")=0;
   register unsigned int r5 asm("r5")=0;

   asm volatile("mov r1,#7");
   asm volatile("mov r2,#3");
   asm volatile("mov r3,#2");
   asm volatile("add r0,r1,r2,lsl r3");//3*2^2 한후에 7 더하고 저장

   printf("r0=0x%x\n",r0);

   return 0;
}
```

5. lsl3

```
#include<stdio.h>

int main()
{

   register unsigned int r0 asm("r0")=0;
   register unsigned int r1 asm("r1")=0;
   register unsigned int r2 asm("r2")=0;
   register unsigned int r3 asm("r3")=0;
   register unsigned int r4 asm("r4")=0;
   register unsigned int r5 asm("r5")=0;

   asm volatile("mov r1,#2");
   asm volatile("add r0,r1,r1,lsl #2");//2 에 lsl 2 해주고 2 를 더한후 저장

   printf("r0=0x%x\n",r0);

   return 0;
}
```

6. asr

```
int main()
{

   register unsigned int r0 asm("r0")=0;
   register unsigned int r1 asm("r1")=0;
   register unsigned int r2 asm("r2")=0;
   register unsigned int r3 asm("r3")=0;
   register unsigned int r4 asm("r4")=0;
   register unsigned int r5 asm("r5")=0;

   asm volatile("mov r1,#32");
   asm volatile("add r0,r1,asr #2");//r1/4 -> save to r0

   printf("r0=0x%x\n",r0);

   return 0;
}
```

7. msr ?
8. mul

```
    asm volatile("mov r2,#3");
    asm volatile("mov r3,#7");
    asm volatile("mul r1,r2,r3");//r2*r3 -> save to r1

    printf("r1=%d\n",r1);

    return 0;
}
```

9. mla

```
    asm volatile("mov r2,#3");
    asm volatile("mov r3,#7");
    asm volatile("mov r4,#33");
    asm volatile("mla r1,r2,r3,r4");//a*b+c r2*r3+r4
```

10. umull

```
   sm volatile("mov r2,#0x44,8");
    asm volatile("mov r3,#0x200");
    asm volatile("umull r0,r1,r2,r3");//상위비트만 나오니까 수정이 필요하다 .
    //asm volatile("umull r4,r5,r8,r9");//수정후
    printf("r1r0=0x%x%x\n",r1,r0);//수정전
    //printf("r1r0=0x%x %08x\n",r5,r4)//수정후

    return 0;
```

11. umlal

```
   asm volatile("mov r0,#0xf");
    asm volatile("mov r1,#0x1");
    asm volatile("mov r2,#0x44,8");
    asm volatile("mov r3,#0x200");
    asm volatile("umlal r0,r1,r2,r3");//r2*r3 상위 r1 에 덧셈 하고 , 하위 r0 덧셈 한다.

    printf("r1r0=0x%x%x\n",r1,r0);

    return 0;
```

12. ldr

```
#include<stdio.h>

unsigned int arr[5]={1,2,3,4,5};

void show_reg(unsigned int reg)
{
   int i;

   for(i=31;i>=0;)
        printf("%d",(reg>>i--)&1);
   printf("\n");
}

int main()
{

   register unsigned int r0 asm("r0")=0;
   register unsigned int *r1 asm("r1")=NULL;
   register unsigned int *r2 asm("r2")=NULL;
   register unsigned int r3 asm("r3")=0;
   register unsigned int r4 asm("r4")=0;
   register unsigned int r5 asm("r5")=0;

   r1=arr;

   asm volatile("mov r2,#0x8");//8 바이트 만큼 이동.
   asm volatile("ldr r0,[r1,r2]");//메모리 -> 레지스터 r0 은 3

   printf("r0=%u\n",r0);

   return 0;
}
```

13.ldr2

```
#include<stdio.h>

unsigned int arr[5]={1,2,3,4,5};

void show_reg(unsigned int reg)
{
   int i;

   for(i=31;i>=0;)
        printf("%d",(reg>>i--)&1);
   printf("\n");
}

int main()
{

   register unsigned int r0 asm("r0")=0;
   register unsigned int *r1 asm("r1")=NULL;
   register unsigned int *r2 asm("r2")=NULL;
   register unsigned int r3 asm("r3")=0;
   register unsigned int r4 asm("r4")=0;
   register unsigned int r5 asm("r5")=0;

   r1=arr;

   asm volatile("ldr r0,[r1,#0x4]");//r0 2.

   printf("r0=%u\n",r0);

   return 0;
}
```

14. ldreqb

```
#include<stdio.h>

char test[]="HelloARM";

void show_reg(unsigned int reg)
{
   int i;

   for(i=31;i>=0;)
        printf("%d",(reg>>i--)&1);
   printf("\n");
}

int main()
{

   register unsigned int r0 asm("r0")=0;
   register char *r1 asm("r1")=NULL;
   register unsigned int *r2 asm("r2")=NULL;
   register unsigned int r3 asm("r3")=0;
   register unsigned int r4 asm("r4")=0;
   register unsigned int r5 asm("r5")=0;

   r1=test;

   asm volatile("ldreqb r0,[r1,#0x5]");//A

   printf("r0=%c\n",r0);

   return 0;
}
```

15.strb

```
#include<stdio.h>

char test[]="HelloARM";

void show_reg(unsigned int reg)
{
   int i;

   for(i=31;i>=0;)
        printf("%d",(reg>>i--)&1);
   printf("\n");
}

int main()
{

   register unsigned int r0 asm("r0")=0;
   register char *r1 asm("r1")=NULL;
   register unsigned int *r2 asm("r2")=NULL;
   register unsigned int r3 asm("r3")=0;
   register unsigned int r4 asm("r4")=0;
   register unsigned int r5 asm("r5")=0;

   r1=&test[5];    //r1=test

   asm volatile("mov r0,#61");
   asm volatile("strb r0,[r1]"); //레지스터에서 메모리로 간다.   //strb r0,[r1,#5] Hello=RM

   printf("test=%s\n",test);

   return 0;
}
```

16.ldr !

```
#include<stdio.h>

char test[]="HelloARM";

void show_reg(unsigned int reg)
{
   int i;

   for(i=31;i>=0;)
        printf("%d",(reg>>i--)&1);
   printf("\n");
}

int main()
{

   register unsigned int r0 asm("r0")=0;
   register char *r1 asm("r1")=NULL;
   register unsigned int *r2 asm("r2")=NULL;
   register unsigned int r3 asm("r3")=0;
   register unsigned int r4 asm("r4")=0;
   register unsigned int r5 asm("r5")=0;

   r1=test;

   asm volatile("mov r2,#0x5");
   asm volatile("ldr r0,[r1,r2]!");//r0 결과값 저장 , r1 주소 , r2 이동

   printf("test=%s,r1=%s\n",test,r1);//ARM  !는 이동한곳까지 값을 갱신 한다.

   return 0;
}
```

17. ldr4

```
#include<stdio.h>

unsigned int arr[5]={1,2,3,4,5};

void show_reg(unsigned int reg)
{
   int i;

   for(i=31;i>=0;)
        printf("%d",(reg>>i--)&1);
   printf("\n");
}

int main()
{

   register unsigned int r0 asm("r0")=0;
   register unsigned int *r1 asm("r1")=NULL;
   register unsigned int *r2 asm("r2")=NULL;
   register unsigned int r3 asm("r3")=0;
   register unsigned int r4 asm("r4")=0;
   register unsigned int r5 asm("r5")=0;

   r1=arr;

   asm volatile("mov r2,#0x4");
   asm volatile("ldr r0,[r1],r2");//r1 시작주소 1 이 r0 에 , r2 4 바이트만큼 이동한것이 2
   printf("r0=%u,r1=%u\n",r0,*r1);

   return 0;
}
```

18. stmia

```
#include<stdio.h>

int main()
{

   int i;
   unsigned int test_arr[5]={0};
   register unsigned int *r0 asm("r0")=0;
   register unsigned int r1 asm("r1")=0;
   register unsigned int r2 asm("r2")=0;
   register unsigned int r3 asm("r3")=0;
   register unsigned int r4 asm("r4")=0;
   register unsigned int r5 asm("r5")=0;

   r0=test_arr;

   asm volatile("mov r1,#0x3");//3
   asm volatile("mov r2,r1,lsl #2");//12
   asm volatile("mov r4,#0x2") ;//2
   asm volatile("add r3,r1,r2,lsl r4");//51
   asm volatile("stmia r0,{r1,r2,r3}");//ia increament after (선증가후 삽입) 3 12 2 51

   for(i=0;i<5;i++)
        printf("test_arr[%d]=%d\n",i,test_arr[i]);

   return 0;
}
```

19. stmia2

```
#include<stdio.h>

int main()
{

   int i;
   unsigned int test_arr[5]={0};
   register unsigned int *r0 asm("r0")=0;
   register unsigned int r1 asm("r1")=0;
   register unsigned int r2 asm("r2")=0;
   register unsigned int r3 asm("r3")=0;
   register unsigned int r4 asm("r4")=0;
   register unsigned int r5 asm("r5")=0;

   r0=test_arr;

   asm volatile("mov r1,#0x3");//3
   asm volatile("mov r2,r1,lsl #2");//12
   asm volatile("mov r4,#0x2");//2
   asm volatile("add r3,r1,r2,lsl r4");//51
   asm volatile("stmia r0!,{r1,r2,r3}");//!는 이동한 값을 갱신  r0 위치가 r3 뒤다 ..
   asm volatile("str r4,[r0]");//레지스터 -> 메모리 r4 에 2 가 들어간다.

   for(i=0;i<5;i++)
        printf("test_arr[%d]=%d\n",i,test_arr[i]);

   return 0;
}
```

20. stmia3

```
#include<stdio.h>

int main()
{

   int i;
   unsigned int test_arr[5]={0};
   register unsigned int *r0 asm("r0")=0;
   register unsigned int r1 asm("r1")=0;
   register unsigned int r2 asm("r2")=0;
   register unsigned int r3 asm("r3")=0;
   register unsigned int r4 asm("r4")=0;
   register unsigned int r5 asm("r5")=0;

   r0=test_arr;

   asm volatile("mov r1,#0x3\n"
                "mov r2,r1,lsl #2\n"
                "mov r4,#0x2\n"
                "add r3,r1,r2,lsl r4\n"
                "stmia r0!,{r1,r2,r3}\n"
                "str r4,[r0]");
   for(i=0;i<5;i++)
        printf("test_arr[%d]=%d\n",i,test_arr[i]);

   return 0;
}
```

21. ldmla

```
#include<stdio.h>

int main()
{

   int i;
   unsigned int test_arr[7]={0};
   register unsigned int *r0 asm("r0")=0;
   register unsigned int r1 asm("r1")=0;
   register unsigned int r2 asm("r2")=0;
   register unsigned int r3 asm("r3")=0;
   register unsigned int r4 asm("r4")=0;
   register unsigned int r5 asm("r5")=0;
   register unsigned int r6 asm("r6")=0;

   r0=test_arr;


   asm volatile("mov r1,#0x3\n"//3
            "mov r2,r1,lsl #2\n"//12
            "mov r4,#0x2\n"//2
            "add r3,r1,r2,lsl r4\n"//51
            "stmia r0!,{r1,r2,r3}\n"//r0 위치가 r3 뒤 . ! 갱신이니까 ..
            "str r4,[r0]\n"//r0 위치에 r4 값을 넣는다. (메모리 -> 레지스터)
            "mov r5,#128\n"
            "mov r6,r5,lsr #3\n"//r6 : 16
            "stmia r0,{r4,r5,r6}\n"//2,128,16,0,0
            "sub r0,r0,#12\n"//r0 4byte*3 12 전으로 돌아감 .
            "ldmia r0,{r4,r5,r6}");//메모리 -> 레지스터

            for(i=0;i<7;i++)
                 printf("test_arr[%d]=%d\n",i,test_arr[i]);

            printf("r4=%u,r5=%u,r6=%u\n",r4,r5,r6);


   return 0;
}
```