Head.S 분석자료

16기 A조

Head.s @0 stext

X0	DTB (PHY ADDRERSS)		
X1	0000000_0000000	X16	
X2	0000000_0000000	X17	
Х3	0000000_0000000	X18	
X4		X19	
X5		X20	
X6		X21	
X7		X22	
X8		X23	
Х9		X24	
X10		X25	
X11		X26	
X12		X27	
X13		X28	
X14		X29	FP
X15		X30	LR

Head.s @1 stext

bl preserve_boot_args

		1		
X0	DTB (PHY ADDRERSS)			
X1	0000000_0000000	X16		
X2	0000000_0000000	X17		
Х3	0000000_0000000	X18		
X4		X19		
X5		X20		
Х6		X21		
X7		X22		
X8		X23		
X9		X24		
X10		X25		
X11		X26		
X12		X27		
X13		X28		
X14		X29		FP
X15		X30	stext+8	LR

Head.s @2 preserve_boot_args

mov x21, x0

DTB (PHY ADDRERSS)		
00000000_00000000	X16	
00000000_00000000	X17	
00000000_00000000	X18	
	X19	
	X20	
	X21	DTB (PHY ADDRERSS)
	X22	
	X23	
	X24	
	X25	
	X26	
	X27	
	X28	
	X29	
	X30	stext+8
	00000000_00000000	00000000_00000000 X16 00000000_00000000 X18 00000000_00000000 X18 X20 X21 X22 X23 X24 X25 X26 X27 X28 X29

FP

Head.s @3 preserve_boot_args

adr_l x0, boot_args
u64 __cacheline_aligned boot_args[4];

		_	
X0	&boot_args		
X1	0000000_0000000	X16	
X2	00000000_00000000	X17	
Х3	00000000_00000000	X18	
X4		X19	
X5		X20	
X6		X21	DTB (PHY ADDRERSS)
X7		X22	
X8		X23	
X9		X24	
X10		X25	
X11		X26	
X12		X27	
X13		X28	
X14		X29	
X15		X30	stext+8

FP

Head.s @4 preserve_boot_args

```
stp x21, x1, [x0]

stp x2, x3, [x0, #16]

u64 __cacheline_aligned boot_args[4];

boot_args[0] = x21; // DTB

boot_args[1] = x1; // 0

boot_args[2] = x2; // 0

boot_args[3] = x3; // 0
```

X0	&boot_args			
X1	00000000_00000000	X16		
X2	00000000_00000000	X17		
Х3	00000000_00000000	X18		
X4		X19		
X5		X20		
X6		X21	DTB (PHY ADDRERSS)	
X7		X22		
X8		X23		
X9		X24		
X10		X25		
X11		X26		
X12		X27		
X13		X28		
X14		X29		FP
X15		X30	stext+8	LR

Head.s @5 preserve_boot_args

dmb sy

DMB

Data Memory Barrier.

SY

Full system barrier operation. This is the default and can be omitted.

boot_arg[]를 저장한 명령어가
Chche clear 명령어에 영향을
받지 않도록 Barrier를 사용한다.
(메모리 오더링 이슈는 1 core에서도 발생한다)

X0	&boot_args		
X1	00000000_00000000	X16	
X2	00000000_00000000	X17	
Х3	0000000_0000000	X18	
X4		X19	
X5		X20	
X6		X21	DTB (PHY ADDRERSS)
X7		X22	
X8		X23	
Х9		X24	
X10		X25	
X11		X26	
X12		X27	
X13		X28	
X14		X29	
X15		X30	stext+8

FP LR Head.s @6 preserve_boot_args

mov x1, #0x20

 $4 \times 8 = 32 = 0 \times 20$

X0	&boot_args		
X1	00000000_00000020	X16	
X2	00000000_00000000	X17	
Х3	00000000_00000000	X18	
X4		X19	
X5		X20	
X6		X21	DTB (PHY ADDRERSS)
Х7		X22	
X8		X23	
Х9		X24	
X10		X25	
X11		X26	
X12		X27	
X13		X28	
X14		X29	
X15		X30	stext+8

FP

Head.s @7 preserve_boot_args

b __inval_dcache_area

__inval_dcache_area(void *addr, size_t len);

X0 - addr

X1 – len

Ensure that any D-cache lines are invalidated.

X0	&boot_args		
X1	00000000_00000020	X16	
X2	0000000_0000000	X17	
Х3	0000000_00000000	X18	
X4		X19	
X5		X20	
X6		X21	DTB (PHY ADDRERSS)
Х7		X22	
X8		X23	
Х9		X24	
X10		X25	
X11		X26	
X12		X27	
X13		X28	
X14		X29	
X15		X30	stext+8

FP

Head.s @8 stext

bl el2_setup

X0	&boot_args		
X1	00000000_00000020	X16	
X2	0000000_0000000	X17	
Х3	00000000_00000000	X18	
X4		X19	
X5		X20	
X6		X21	DTB (PHY ADDRERSS)
X7		X22	
X8		X23	
X9		X24	
X10		X25	
X11		X26	
X12		X27	
X13		X28	
X14		X29	
X15		X30	stext+16

FP

Head.s @9 el2_setup

SPsel, #1 msr

X0	&boot_args			
X1	0000000_00000020	X16		
X2	00000000_00000000	X17		
Х3	00000000_00000000	X18		
X4		X19		
X5		X20		
X6		X21	DTB (PHY ADDRERSS)	
X7		X22		
X8		X23		
Х9		X24		
X10		X25		
X11		X26		
X12		X27		
X13		X28		
X14		X29		FP
X15		X30	stext+16	LR

SPSEL

 $00000000_00000001$

Head.s @10 el2_setup

mrs x0, CurrentEL

System이 EL1 모드로 부팅했다고 가정함

The CurrentEL bit assignments are:



Bits [31:4]

Reserved, RESO.

EL, bits [3:2]

Current exception level. Possible values of this field are:

00 EL0 01 EL1 10 EL2 11 EL3

Resets to an IMPLEMENTATION DEFINED value.

Bits [1:0]

Reserved, RES0.

X0	00000000_00000004			
X1	00000000_00000020	X16		
X2	0000000_0000000	X17		
Х3	0000000_0000000	X18		
X4		X19		
X5		X20		
X6		X21	DTB (PHY ADDRERSS)	1
Х7		X22		
X8		X23		
Х9		X24]
X10		X25]
X11		X26		1
X12		X27		
X13		X28		
X14		X29		FF
X15		X30	stext+16	LR

SPSEL 000000000_00000001

Head.s @11 el2_setup

cmp x0, #CurrentEL_EL2 (=8)

X0	00000000_00000004		
X1	0000000_00000020	X16	
X2	0000000_00000000	X17	
ХЗ	0000000_00000000	X18	
X4		X19	
X5		X20	
X6		X21	DTB (PHY ADDRERSS)
Х7		X22	
Х8		X23	
Х9		X24	
X10		X25	
X11		X26	
X12		X27	
X13		X28	
X14		X29	
X15		X30	stext+16

FP

LR

/* Current Exception Level values, as contained in CurrentEL *
#define CurrentEL_EL1→ → (1 << 2)

#define CurrentEL_EL2 (2 << 2)

SPSEL

00000000_00000001

Head.s @12 el2_setup

b.eq 1f

같지 않으므로 skip

X0	00000000_00000004			
X1	0000000_00000020	X16		
X2	0000000_0000000	X17		
ХЗ	0000000_0000000	X18		
X4		X19		
X5		X20		
X6		X21	DTB (PHY ADDRERSS)	
Х7		X22		
Х8		X23		
Х9		X24		
X10		X25		
X11		X26		
X12		X27		
X13		X28		
X14		X29		FP
X15		X30	stext+16	LR

SPSEL

00000000_00000001

Head.s @13 el2_setup

mov_q x0, (SCTLR_EL1_RES1 | ENDIAN_SET_EL1)
(SCTLR_EL1_RES1 | ENDIAN_SET_EL1) -> 0x30500800

X0	0000000_30500800		
X1	00000000_00000020	X16	
X2	00000000_00000000	X17	
Х3	0000000_0000000	X18	
X4		X19	
X5		X20	
X6		X21	DTB (PHY ADDRERSS)
X7		X22	
X8		X23	
Х9		X24	
X10		X25	
X11		X26	
X12		X27	
X13		X28	
X14		X29	
X15		X30	stext+16

FP

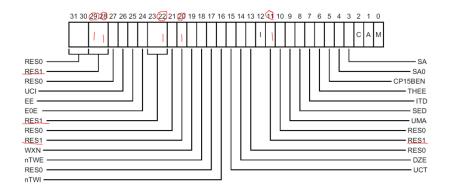
SPSEL

00000000_00000001

Head.s @14 el2_setup

msr sctlr_el1, x0

Reserved 된 값을 1로 채움 왜 하는지는 정확히 모름



X0	0000000_30500800			
X1	00000000_00000020	X16		
X2	0000000_0000000	X17		
ХЗ	0000000_0000000	X18		
X4		X19		
X5		X20		
X6		X21	DTB (PHY ADDRERSS)	
Х7		X22		
Х8		X23		
Х9		X24		
X10		X25		
X11		X26		
X12		X27		
X13		X28		
X14		X29		FP
X15		X30	stext+16	LR

SPSEL 00000000_00000001

Head.s @15 el2_setup

mov w0, #BOOT_CPU_MODE_EL1 (=0xe11)

#define BOOT_CPU_MODE_EL1 \rightarrow (0xe11) #define BOOT_CPU_MODE_EL2 \rightarrow (0xe12)

X0	00000000_00000E11			
X1	00000000_00000020	X16		
X2	0000000_0000000	X17		
Х3	0000000_0000000	X18		
X4		X19		
X5		X20		
X6		X21	DTB (PHY ADDRERSS)	
Х7		X22		
X8		X23		
Х9		X24		
X10		X25		
X11		X26		
X12		X27		
X13		X28		
X14		X29		
X15		X30	stext+16	۱ [

SPSEL 000000000_00000001

Head.s @16 el2_setup

ret

stext로 return 함

X0	00000000_00000E11			
X1	00000000_00000020	X16		
X2	0000000_0000000	X17		
Х3	0000000_0000000	X18		
X4		X19		
X5		X20		
X6		X21	DTB (PHY ADDRERSS)	
X7		X22		
X8		X23		
X9		X24		
X10		X25		
X11		X26		
X12		X27		
X13		X28		
X14		X29		F
X15		X30	stext+16	Įι

SPSEL 000000000_00000001

SCTLR_EL1
00000000_30500800

FP

Head.s @17 stext

adrp x23, __PHYS_OFFSET

KERNEL_START = ffff000010080000

 $TEXT_OFFSET = 0x80000$

__PHYS_OFFSET = ffff000010000000

#define __PHYS_OFFSET (KERNEL_START - TEXT_OFFSET)

X0	00000000_00000E11			
X1	00000000_00000020	X16		
X2	0000000_0000000	X17		
ХЗ	0000000_0000000	X18		
X4		X19		
X5		X20		
X6		X21	DTB (PHY ADDRERSS)	
Х7		X22		
X8		X23	FFFF0000_10000000	
Х9		X24		
X10		X25		
X11		X26		
X12		X27		
X13		X28		
X14		X29		F
X15		X30	stext+16	<u>ا</u> د

SPSEL 000000000_00000001

Head.s @18 stext

and x23, x23, MIN_KIMG_ALIGN – 1

```
/*

* arm64 requires the kernel image to placed

* TEXT_OFFSET bytes beyond a 2 MB aligned base

*/

#define MIN_KIMG_ALIGN→ SZ_2M
```

"deline be_in		0x00100000
#define SZ_2M→		0x00200000
#define SZ_4M→		0x00400000
#define SZ_8M→		0x00800000
#define SZ_16M→		0x01000000
#define SZ 32M⇒		0x02000000

		_		
X0	00000000_00000E11			
X1	00000000_00000020	X16		
X2	0000000_0000000	X17		
Х3	0000000_0000000	X18		
X4		X19		
X5		X20		
X6		X21	DTB (PHY ADDRERSS)	
X7		X22		
X8		X23	00000000_00000000	
X9		X24		
X10		X25		
X11		X26		
X12		X27		1
X13		X28		1
X14		X29		FP
X15		X30	stext+16	LR
	X1 X2 X3 X4 X5 X6 X7 X8 X9 X10 X11 X12 X13	X1 00000000_00000000 X2 00000000_00000000 X3 00000000_00000000 X4 X5 X6 X7 X8 X9 X10 X11 X12 X13 X14 X14	X1 00000000_00000000 X16 X2 00000000_00000000 X17 X3 00000000_00000000 X18 X4 X19 X5 X20 X6 X21 X7 X22 X8 X23 X9 X24 X10 X25 X11 X26 X12 X27 X13 X28 X14 X29	X1 00000000_00000000 X16 X2 00000000_00000000 X17 X3 00000000_00000000 X18 X4 X19 X5 X20 X6 X21 DTB (PHY ADDRERSS) X7 X22 X8 X23 00000000_0000000 X9 X24 X10 X25 X11 X26 X12 X27 X13 X28 X14 X29

SPSEL 000000000000001

Head.s @19 stext

bl set_cpu_boot_mode_flag

X0	00000000_00000E11		
X1	00000000_00000020	X16	
X2	0000000_00000000	X17	
ХЗ	0000000_0000000	X18	
X4		X19	
X5		X20	
X6		X21	DTB (PHY ADDRERSS)
Х7		X22	
Х8		X23	00000000_00000000
X9		X24	
X10		X25	
X11		X26	
X12		X27	
X13		X28	
X14		X29	
X15		X30	stext+40

SPSEL 000000000_00000001

SCTLR_EL1
00000000_30500800

FP

Head.s @20 set_cpu_boot_mode_flag

adr_l x1, __boot_cpu_mode

	X0	00000000_00000E11			
l	X1	&boot_cpu_mode	X16		
l	X2	0000000_0000000	X17		
l	Х3	0000000_0000000	X18		
l	X4		X19		
l	X5		X20		
l	X6		X21	DTB (PHY ADDRERSS)	
l	X7		X22		
l	X8		X23	00000000_00000000	
l	X9		X24		
l	X10		X25		
l	X11		X26		
	X12		X27		
	X13		X28		
	X14		X29		F
	X15		X30	stext+40] L

SPSEL 000000000_00000001

Head.s @21 set_cpu_boot_mode_flag

cmp w0, #BOOT_CPU_MODE_EL2

#define BOOT_CPU_MODE_EL1 \rightarrow (0xe11) #define BOOT_CPU_MODE_EL2 \rightarrow (0xe12)

		_		
X0	00000000_00000E11			
X1	&boot_cpu_mode	X16		
X2	00000000_00000000	X17		
Х3	00000000_00000000	X18		
X4		X19		
X5		X20		
X6		X21	DTB (PHY ADDRERSS)	
Х7		X22		
X8		X23	00000000_00000000	
Х9		X24		
X10		X25		
X11		X26		
X12		X27		
X13		X28		1
X14		X29		FP
X15		X30	stext+40	LR

SPSEL 000000000_00000001

Head.s @22 set_cpu_boot_mode_flag

b.ne 1f

0xE11 != 0xE12 -> b 1f

X0	00000000_00000E11			
X1	&boot_cpu_mode	X16		
X2	0000000_0000000	X17		
ХЗ	0000000_0000000	X18		
X4		X19		
X5		X20		
X6		X21	DTB (PHY ADDRERSS)	
Х7		X22		
Х8		X23	00000000_00000000	
Х9		X24		
X10		X25		
X11		X26		
X12		X27		
X13		X28		
X14		X29		FI
X15		X30	stext+40] u

SPSEL 00000000_00000001

SCTLR_EL1 0000000_30500800

```
str w0, [x1]

*x1 = w0;
_boot_cpu_mode = 0x00000E11
```

Head.s @23 set_cpu_boot_mode_flag

		_		
Х0	00000000_00000E11			
X1	&boot_cpu_mode	X16		
X2	00000000_00000000	X17		
Х3	00000000_00000000	X18		
X4		X19		
X5		X20		
Х6		X21	DTB (PHY ADDRERSS)	
X7		X22		
X8		X23	00000000_00000000	
Х9		X24		
X10)	X25		
X11	1	X26		
X12	2	X27		
X13	3	X28		
X14	1	X29		F
X15	5	X30	stext+40] L

SPSEL 000000000_00000001

Head.s @24 set_cpu_boot_mode_flag

dmb sy dc ivac, x1

__boot_cpu_mode에 값을 저장하고 코어내의 boot_cpu_mode 데이터 캐시 라인(PoC 관점) 을 무효화 시 킨다.

 $DC\ IVAC\ {\footnotesize Invalidate\ by\ Virtual\ Address,\ to\ Point\ of\ Coherency}$

00000000_00000E11		
&boot_cpu_mode	X16	
00000000_00000000	X17	
00000000_00000000	X18	
	X19	
	X20	
	X21	DTB (PHY ADDRERSS)
	X22	
	X23	00000000_00000000
	X24	
	X25	
	X26	
	X27	
	X28	
	X29	
	X30	stext+40
	&boot_cpu_mode 00000000_00000000	&boot_cpu_mode X16 00000000_00000000 X17 00000000_00000000 X18 X19 X20 X21 X21 X22 X23 X24 X25 X26 X27 X28 X29

SPSEL 000000000_00000001

SCTLR_EL1
00000000_30500800

FP

Head.s @25 stext

bl __create_page_tables

X0	00000000_00000E11			
X1	&boot_cpu_mode	X16		
X2	00000000_00000000	X17		
Х3	00000000_00000000	X18		
X4		X19		
X5		X20		
X6		X21	DTB (PHY ADDRERSS)	
Х7		X22		
X8		X23	00000000_00000000	
Х9		X24		
X10		X25		
X11		X26		
X12		X27		
X13		X28		
X14		X29		FP
X15		X30	stext+48	LR

SPSEL 00000000000001

Head.s @26 __create_page_tables

mov x28, Ir

X0	00000000_00000E11		
X1	&boot_cpu_mode	X16	
X2	0000000_0000000	X17	
Х3	0000000_0000000	X18	
X4		X19	
X5		X20	
X6		X21	DTB (PHY ADDRERSS)
Х7		X22	
X8		X23	00000000_00000000
Х9		X24	
X10		X25	
X11		X26	
X12		X27	
X13		X28	stext+48
X14		X29	
X15		X30	stext+48

SPSEL 00000000000001

SCTLR_EL1
00000000_30500800

FP LR Head.s @27 __create_page_tables

adrp x0, init_pg_dir adrp x1, init_pg_end

X0	& init_pg_dir		
X1	& init_pg_end	X16	
X2	00000000_00000000	X17	
Х3	00000000_00000000	X18	
X4		X19	
X5		X20	
X6		X21	DTB (PHY ADDRERSS)
X7		X22	
X8		X23	00000000_00000000
Х9		X24	
X10		X25	
X11		X26	
X12		X27	
X13		X28	stext+48
X14		X29	
X15		X30	stext+48

SPSEL 000000000_00000001

SCTLR_EL1
00000000_30500800

FP

```
Head.s @28 __create_page_tables
```

```
sub x1, x1, x0
```

init_pg_dir ~ init_pg_end 까지 크기

```
extern pgd_t init_pg_dir[PTRS_PER_PGD];
everous pgd_t init_pg_end[];
extern pgd_t swapper_pg_dir[PTRS_PER_PGD];
extern pgd_t idmap_pg_dir[PTRS_PER_PGD];
extern pgd_t tramp_pg_dir[PTRS_PER_PGD];
```

```
. = ALIGN(PAGE_SIZE);
init_pg_dir = .;
. += INIT_DIR_SIZE;
init_pg_end = .;
```

X0	& init_pg_dir			
X1	INIT_DIR_SIZE	X16		
X2	00000000_00000000	X17		
Х3	0000000_0000000	X18		
X4		X19		
X5		X20		
X6		X21	DTB (PHY ADDRERSS)	
X7		X22		
X8		X23	00000000_00000000	
Х9		X24		
X10		X25		
X11		X26		
X12		X27		
X13		X28	stext+48	
X14		X29		FP
X15		X30	stext+48	LR

SPSEL 000000000_00000001

Head.s @29 __create_page_tables

bl __inval_dcache_area

Init_pg_dir 부터 INIT_DIR_SIZE 만큼 데이터 캐시를 무효화 시킨다.

X0	& init_pg_dir		
X1	INIT_DIR_SIZE	X16	
X2	00000000_00000000	X17	
Х3	00000000_00000000	X18	
X4		X19	
X5		X20	
X6		X21	DTB (PHY ADDRERSS)
Х7		X22	
X8		X23	00000000_00000000
Х9		X24	
X10		X25	
X11		X26	
X12		X27	
X13		X28	stext+48
X14		X29	
X15		X30	@29 + 8

SPSEL 000000000000001

SCTLR_EL1
00000000_30500800

FP

Head.s @30 __create_page_tables

adrp x0, init_pg_dir adrp x1, init_pg_end

X0	& init_pg_dir			_
X1	& init_pg_end	X16		
X2	00000000_00000000	X17		
Х3	0000000_0000000	X18		
X4		X19		
X5		X20		
X6		X21	DTB (PHY ADDRERSS)	
Х7		X22		
X8		X23	00000000_00000000	
Х9		X24		
X10		X25		
X11		X26		
X12		X27		
X13		X28	stext+48	
X14		X29		FP
X15		X30	@29 + 8	LR

x1, x1, x0

sub

Head.s @31 __create_page_tables

init_pg_dir ~ init_pg_end 까지 크기

```
extern pgd_t init_pg_dir[PTRS_PER_PGD];
every pgd_t init_pg_end[];
extern pgd_t swapper_pg_dir[PTRS_PER_PGD];
extern pgd_t idmap_pg_dir[PTRS_PER_PGD];
extern pgd_t tramp_pg_dir[PTRS_PER_PGD];
```

```
. = ALIGN(PAGE_SIZE);
init_pg_dir = .;
. += INIT_DIR_SIZE;
init_pg_end = .;
```

X0	& init_pg_dir			
X1	INIT_DIR_SIZE	X16		
X2	0000000_0000000	X17		
Х3	0000000_0000000	X18		
X4		X19		
X5		X20		
X6		X21	DTB (PHY ADDRERSS)	
X7		X22		
X8		X23	00000000_00000000	
X9		X24		
X10		X25		
X11		X26		
X12		X27		
X13		X28	stext+48	
X14		X29		FP
X15		X30	@29 + 8	LR

SPSEL 000000000_00000001

Head.s @32 __create_page_tables

stp xzr, xzr, [x0], #16 stp xzr, xzr, [x0], #16 stp xzr, xzr, [x0], #16 stp xzr, xzr, [x0], #16

16 * 4 = 64 bytes memset(init_pg_dir, 0, 64) Init_pg_dir 부터 64byte 씩 0으로 초기화 한다.

x0 += 16

stp xzr, xzr, [x0, #16] preindex $x0 += 16 \\
*(x0) = xzr \\
(x0 + 8) = xzr$ stp xzr, xzr, [x0], #16 postindex $(x0) = xzr \\
*(x0 + 8) = xzr$

X0	& init_pg_dir + 64			
X1	INIT_DIR_SIZE	X16		
X2	00000000_00000000	X17		
Х3	00000000_00000000	X18		
X4		X19		
X5		X20		
X6		X21	DTB (PHY ADDRERSS)	
X7		X22		
X8		X23	00000000_00000000	
X9		X24		
X10		X25		
X11		X26		
X12		X27		
X13		X28	stext+48	
X14		X29		FP
X15		X30	@29 + 8	LR

SPSEL 0000000000000001

Head.s @33 __create_page_tables

subs x1, x1, #64

b.ne 1b

X1 값이 0이 아니면 레이블 1(@32)로 분기 X1 값이 0이면 다음 라인(@34) 실행

X0	& init_pg_dir + 64			
X1	INIT_DIR_SIZE - 64	X16		
X2	00000000_00000000	X17		
Х3	00000000_00000000	X18		
X4		X19		
X5		X20		
X6		X21	DTB (PHY ADDRERSS)	
X7		X22		
X8		X23	00000000_00000000	
Х9		X24		
X10		X25		
X11		X26		
X12		X27		
X13		X28	stext+48	
X14		X29		FP
X15		X30	@29 + 8	LR

SCTLR_EL1
00000000_30500800

SPSEL 00000000000001

x7, SWAPPER_MM_MMUFLAGS mov

SWAPPER_MM_MMUFLAGS

L	X0	& init_pg_dir + 64			_
	X1	INIT_DIR_SIZE - 64	X16		
	X2	00000000_00000000	X17		
	Х3	00000000_00000000	X18		
	X4		X19		
	X5		X20		
	X6		X21	DTB (PHY ADDRERSS)	
	X7	00000000_00000713	X22		
	X8		X23	00000000_00000000	
	X9		X24		
	X10		X25		
	X11		X26		
	X12		X27		
	X13		X28	stext+48	
	X14		X29		FP
Γ	X15		X30	@29 + 8	LR

SPSEL 00000000_00000001