

2.

- (a) LDR R0, =0
STRB R0, u8
- (b) LDR R0, =0
STRH R0, u16
- (c) LDR R0, =0
STR R0, u32
- (d) LDR R0, =0
LDR R1, =0
STRD R0, u64
- (e) LDRB R0, u8
STRH R0, u16
- (f) LDRB R0, u8
STR R0, u32
- (g) LDRH R0, u16
STR R0, u32
- (h) LDR R0, u32
LDR R1, =0
STRD R1, R0, u64
- (i) LDR R0, u32
STRB R0, u8 //does u8 have enough space for a uin32_t
- (j) LDRH R0, u16
STRB R0, u8
- (k) LDR R0, u32
STRH R0, u16

3.

- (a) LDR R0, =-1
STRSB R0, s8
- (b) LDR R0, =-1
STRH R0, s16
- (c) LDR R0, =-1
STR R0, s32
- (d) LDR R0, =-1
LDR R1, =0
STRD R1, R0, s64
- (e) LDRSB R0, s8
STRH R0, s16
- (f) LDRSB R0, s8
STR R0, s32
- (g) LDRH R0, s16
STR R0, s32
- (h) LDR R0, s32

	ASR	R1, R0, 31
	STRD	R1, R0, s64
(i)	LDR	R0, s32
	STRSB	R0, s8
(j)	LDRH	R0, s16
	STRSB	R0, s8
(k)	LDR	R0, s32
	STRH	R0, s16

4.

(a)	ADR	R0, s32	
	STR	R0, ps32	
(b)	LDRSB	R0, [s8]	
	STRSB	R0, ps8	
(c)	ADR	R0, ps8	
	ADD	R0, R0, 1	
	STR	R0, ps8	
(d)	ADR	R0, ps32	
	ADD	R0, R0, 1	
	STR	R0, ps32	
(e)	LDR	R0, =0	
	LDR	R1, s32	
	STR	R0, [R1]	
(f)	LDR	R0, =0	
	LDR	R1, ps8	
	STR	R0, [R1, 1]	
(g)	LDR	R0, =0	
	LDR	R1, ps32	
	STR	R0, [R1, 4]	
(h)	LDR	R0, =0	
	LDR	R1, s32	
	LDR	R2, ps32	
	STR	R0, [R2, R1, LSL, 2]	
(i)	LDR	R0, =0	
	ADR	R1, s32	//R1 = s32[0]
	STRSB	R0, [R1, 1]	

7.

void Swap32(int32_t *p1, int32_t *p2)

Swap32:	//R0 = *p1, R1 = *p2	
	LDR R2, R0	//R2 = R0
	LDR R0, R1	//R0 = R1

LDR R1, R2 //R1 = R2 = R0

