NAME: _____

SANTA CLARA UNIVERSITY
Department of Computer Engineering

COEN 020 Final Exam Spring 2017

(Closed Book & Notes; Honor Code Applies)

NOTE: This final exam consists of two parts with a break in between. You will only be allowed to leave the room during the break. You may take your break as soon as you have completed Part 1. You will be given Part 2 of the exam when you return from your break.

Total Time Allowed: 3 hours

SCU's Academic Integrity Pledge

"I am committed to being a person of integrity. I pledge, as a member of the Santa Clara University community, to abide by and uphold the standards of academic integrity contained in the Student Conduct Code."

Signature:_____

	Points	Maximum
Part 1:		95
Part 2:		67
Total:		162

Overall: _____ %

NAME: _____

SANTA CLARA UNIVERSITY
Department of Computer Engineering

COEN 020 Final Exam (Part 1) Spring 2017

1. [5 pts] Convert -25_{10} from decimal to 8-bit two's complement.

   $25_{10} = 16 + 8 + 1 \rightarrow 00011001_2$

   $-25_{10} = \mathbf{11100111_2}$

2. [5 pts] Convert 1011.0101_2 from two's complement to decimal.

   $1011.0101_2 = -8 + 2 + 1 + .25 + .0625 = \mathbf{-4.6875_{10}}$

3. [5 pts] Convert $7F.35_{16}$ to octal (base 8).

   $7F.35_{16}$ $= 0111\ \ 1111\ .\ 0011\ \ 0101$
   $= 01\ 111\ 111\ .\ 001\ 101\ 01$
   $= 001\ 111\ 111\ .\ 001\ 101\ 010$
   $= \mathbf{1\ \ 7\ \ 7\ .\ 1\ \ 5\ \ 2_8}$

4. [5 pts ea] Translate each of the following into ARM assembly:

`void f1(int32_t, int64_t, int64_t *) ;`
`int32_t  a32 ;`
`int64_t  b64, c64 ;`
`...`
`f1(a32, b64, &c64) ;`
	`LDR  R0,a32`
	`LDRD  R1,R2,b64`
	`ADR  R3,c64`
	`BL  f1`

`int64_t f2(void)`
`  {`
`  return (int64_t) -1 ;`
`  }`
	`f2:  LDR  R0,=-1`
	`  MOV  R1,R0`
	`  BX  LR`

`int32_t f3(int32_t a32)`
`  {`
`  int32_t f4(int32_t) ;`
 	
`  return a32 + f4(10) ;`
`  }`
	`F3:  PUSH  {R4,LR}`
	`  MOV  R4,R0`
	`  LDR  R0,=10`
	`  BL  f4`
	`  ADD  R0,R0,R4`
	`  POP  {R4,PC}`

NAME: _____

SANTA CLARA UNIVERSITY
Department of Computer Engineering

COEN 020 Final Exam (Part 1) Spring 2017

5. [5 pts ea] Translate each of the following into ARM assembly:

`int32_t *p32, k32 ;`
`...`
`p32 = p32 + k32 ;`
	`LDR R0,p32`
	`LDR R1,k32`
	`ADD R0,R0,R1,LSL 2`
	`STR R0,p32`

`int64_t a64[10] ;`
`int32_t k32 ;`
`...`
`a64[k32+1] = 0 ;`
	`LDR R0,=0`
	`ADR R1,a64`
	`LDR R2,k32`
	`ADD R2,R2,1`
	`STRD R0,R0,[R1,R2,LSL 3]`

`int16_t *p16, a16[10] ;`
`int32_t k32 ;`
`...`
`k32 = p16 – a16 ;`
	`LDR R0,p16`
	`ADR R1,a16`
	`SUB R0,R0,R1`
	`ASR R0,R0,1`
	`STR R0,k32`

6. [5 pts] Assume s32 is a 32-bit signed integer and s64 is a 64-bit signed integer. Circle the code that correctly adds s32 to s64:

LDRD R0,R1,s64
LDR R2,s32
ADDS R0,R0,R2
ADC R1,R1,0
STRD R0,R1,s64
	LDRD R0,R1,s64
	LDR R2,s32
	ADDS R0,R0,R2
	ADC R1,R1,R2,ASR 31
	STRD R0,R1,s64
		LDRD R0,R1,s64
		LDR R2,s32
		ADDS R0,R0,R2
		ADC R1,R1,-1,ASR 31
		STRD R0,R1,s64

7. [5 pts] True or False? If an ADDS instruction is used to add two 32-bit unsigned integers, the overflow flag (V) will be 1 if an overflow occurred.

Answer: False (Unsigned overflow is indicated by C=1)

8. [5 pts] Which of the following signed double-length products would be an overflow condition if we were only going to keep the single-length product?

FFFF $8BCE_{16}$	0000 $7F45_{16}$	FFFF $0ABC_{16}$

NAME: _____

SANTA CLARA UNIVERSITY
Department of Computer Engineering

COEN 020 Final Exam (Part 1) Spring 2017

9. [5 pts ea] Indicate whether or not it's possible to translate the C code into a single IT block and with no conditional branch instructions.

`int64_t a64 ;`
 	
`if (a64 == 0) foo() ; else bar() ;`
	YES____
	
	NO ____

`int32_t a32 ;`
 	
`if (a32 < 20 || a32 > 50) doit() ;`
	YES____
	
	NO ____

`int32_t a32, b32, c32 ;`
 	
`if (a32 != 0) a32 += b32 + c32 ;`
	YES____
	
	NO ____

10. [5 pts ea] For each row below enter an "X" in a column if the C code is equivalent to the assembly code.

`uint32_t u32 ;`	`u32 |= (1 << 5) ;`	`u32 &= ~(1 << 5) ;`	`u32 ^= (1 << 5) ;`

`LDR R0,u32`
`BFC R0,5,1`
`STR R0,u32`
		X	

`LDR R0,u32`
`EOR R0,R0,1<<5`
`STR R0,u32`
			X

`LDR R0,u32`
`AND R0,R0,1<<5`
`STR R0,u32`			

11. [5 pts] Circle the code that does NOT implement x=y/16, where x and y are int32_t:

```
LDR   R0,y
ADD   R0,R0,15
ASR   R0,R0,4
STR   R0,x
```

```
LDR   R0,y
CMP   R0,0
IT    LT
ADDLT R0,R0,15
ASR   R0,R0,4
STR   R0,x
```

```
LDR   R0,y
ASR   R1,R0,31
AND   R1,R1,15
ADD   R0,R0,R1
ASR   R0,R0,4
STR   R0,x
```