NAME: _____

SANTA CLARA UNIVERSITY
Department of Computer Engineering

COEN 020 Midterm Exam Fall 2016

(Closed book & notes; No electronic devices)

Time Allowed: 1 hour

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:_____

Your Points:	
Max Points:	124
Your Score:	%

Points this page: _____ (16 max)

COEN 020 Midterm Exam Fall 2016

1. [1 pt ea] What is the range of each of the following data types?

   a. uint8_t: 0 to 255 (2^8-1)

   b. int8_t: -128 (-2^7) to +127 (+2^7-1)

2. [3 pts ea] Convert the following <u>unsigned</u> integer representations:

   a. 4.5_6 → $?_{10}$ $4x6^0 + 5x6^{-1} = 4 + 5/6 = 4.8333..._{10}$

   b. 27_{10} → $?_5$ $1x5^2 + 0x5^1 + 2x5^0 = 102_5$

      27÷5 → Q=5, R=2

      5÷5 → Q=1, R=0

      1÷5 → Q=0, R=1

   c. 0.875_{10} → $?_4$ $= 0.32_4$

      4x.875 → 3.5

      4x.5 → 2.0

   d. 101.101_2 → $?_{16}$ $= 0101 . 1010 = 5.A_{16}$

Hex	Binary	Hex	Binary
0	0000	8	1000
1	0001	9	1001
2	0010	A	1010
3	0011	B	1011
4	0100	C	1100
5	0101	D	1101
6	0110	E	1110
7	0111	F	1111

3. [5 pts] Convert 1011.0110_2 from signed 2's complement to decimal.

   $1011.0110 = -0100.1010 = -4.625_{10}$

4. [5 pts] Convert -50_{10} to an 8-bit 2's complement representation.

   $+50_{10} = 32 + 16 + 2 = 2^5 + 2^4 + 2^1 \rightarrow 00110010_2$

   $-50_{10} = -00110010_2 = 11001110_2$

5. [4 pts total] Complete the binary addition, showing each of the carry and sum bits:

Carries:	0	1	1	1	0
A:		0	1	0	1
B:		0	0	1	1
A+B:		1	0	0	0

6. [2 pts ea] Consider the following addition:

Carries:	**0**	**1**	?	?	?
A:		?	?	?	?
B:		?	?	?	?
A+B:		?	?	?	?

   a. Did an overflow occur if the operands were unsigned? NO

   b. Did an overflow occur if the operands were 2's complement? YES

Points this page: _____ (40 max)

7. [10 pts ea] Translate the following C into ARM assembly:

C Function	*ARM Assembly*

`uint64_t Promote(uint8_t x)`
`  {`
`  return (uint64_t) x ;`
`  }`
	`Promote:`
	`  LDR  R1,=0`
	`  BX  LR`

`int32_t One(int32_t x)`
`  {`
`  int32_t Two(void) ;`
 	
`  return x + Two() ;`
`  }`
	`One:`
	`  PUSH  {R4,LR}`
	`  MOV  R4,R0`
	`  BL  Two`
	`  ADD  R0,R0,R4`
	`  POP  {R4,LR}`
	`  BX  LR`

`int16_t Get(int16_t a[], int32_t k)`
`  {`
`  return a[k+1] ;`
`  }`
	`Get:`
	`  ADD  R1,R1,1`
	`  LDRSH R0,[R0,R1,LSL 1]`
	`  BX  LR`

`int64_t *PlusK(int64_t *p, int32_t k)`
`  {`
`  return p + k ;`
`  }`
	`PlusK:`
	`  ADD  R0,R0,R1,LSL 3`
	`  BX  LR`

Points this page: _____ (30 max)

8. [10 pts ea] Translate the following C into ARM assembly:

C Function	*ARM Assembly*

`int32_t IsLT(int64_t a, int64_t b)`
`  {`
`  return (a < b) ? 1 : 0 ;`
`  }`
	`// Use an IT block`
	
	`IsLT:`
	`  SUBS  R0,R0,R2`
	`  SBCS  R1,R1,R3`
	`  ITE  LT`
	`  LDRLT  R0,=1`
	`  LDRGE  R0,=0`
	`  BX  LR`

`int32_t Rem(int32_t a, int32_t b)`
`  {`
`  return a % b ;`
`  }`
	`Rem:`
	`  SDIV  R2,R0,R1`
	`  MLS  R0,R1,R2,R0`
	`  BX  LR`

`int32_t GetAndClear(int32_t *p)`
`  {`
`  int32_t temp ; // use a register`
 	
`  temp = *p ;`
`  *p = 0 ;`
`  return temp ;`
`  }`
	`GetAndClear:`
	`  LDR  R1,[R0]`
	`  LDR  R2,=0`
	`  STR  R2,[R0]`
	`  MOV  R0,R1`
	`  BX  LR`

Points this page: _____ (20 max)

9. [10 pts ea] Translate the following C into ARM assembly:

C Function	*ARM Assembly*

`int32_t Valid(int32_t score)`
`  {`
`  if (0 <= score && score <= 100)`
`  return 1 ;`
`  else`
`  return 0 ;`
`  }`
	`// Do NOT use IT block`
	
	`Valid:`
	`  CMP  R0,0`
	`  BLT  Else`
	`  CMP  R0,100`
	`  BGT  Else`
	`Then: LDR  R0,=1`
	`  B  EndIf`
	`Else:  LDR  R0,=0`
	`EndIf: BX  LR`

`int32_t Note(uint32_t score)`
`  {`
`  if (score < 60 || score > 100)`
`  return 1 ;`
`  else`
`  return 0 ;`
`  }`
	`// Do NOT use IT block`
	
	`Note:`
	`  CMP  R0,60`
	`  BLO  Then`
	`  CMP  R0,100`
	`  BLS  Else`
	`Then: LDR  R0,=1`
	`  B  EndIf`
	`Else:  LDR  R0,=0`
	`EndIf: BX  LR`