NAME: _____

SANTA CLARA UNIVERSITY
Department of Computer Engineering

COEN 020 Final Exam (Part 2) Fall 2016

11. [5 pts ea] Convert each C function call into ARM assembly language.

C code	ARM Assembly

`int8_t  x8 ;`
`int64_t y64 ;`
`void f1(int8_t, int64_t) ;`
 	
`f1(x8, y64) ;`
	`LDRSB  R0,x8`
	`LDRD  R1,R2,y64`
	`BL  f1`

`uint64_t z64, f2(void) ;`
 	
`z64 = f2() ;`
	`BL  f2`
	`STRD  R0,R1,z64`

`int32_t count ;`
`float  real ;`
`void f3(int32_t, float) ;`
 	
`f3(count, real) ;`
	`LDR  R0,count`
	`VLDR  S0,real`
	`BL  f3`

`float result ;`
`float f4(void) ;`
 	
`result = f4() ;`
	`BL  f4`
	`ADR  R0,result`
	`VSTR  S0,[R0]`

NAME: _____

SANTA CLARA UNIVERSITY
Department of Computer Engineering

COEN 020 Final Exam (Part 2) Fall 2016

12. [5 pts ea] Convert each C function definition into ARM assembly language.

C code	ARM Assembly

`int64_t f5(int32_t s32)`
`  {`
`  return (int64_t) s32 ;`
`  }`
	`f5:  ASR  R1,R0,31`
	`  BX  LR`

`float f6(int8_t s8)`
`  {`
`  return (float) s8 ;`
`  }`
	`f6:  VMOV  S0,R0`
	`  VCVT.F32.S32  S0,S0`
	`  BX  LR`

`uint64_t f7(uint64_t u64)`
`  {`
`  return u64 + 1 ;`
`  }`
	`f7:  ADDS  R0,R0,1`
	`  ADC  R1,R1,0`
	`  BX  LR`

`uint64_t f8(uint64_t u64)`
`  {`
`  return u64 << 4 ;`
`  }`
	`f8:  LSL  R1,R1,4`
	`  ORR  R1,R1,R0,LSR 28`
	`  LSL  R0,R0,4`
	`  BX  LR`

NAME: _____

SANTA CLARA UNIVERSITY
Department of Computer Engineering

COEN 020 Final Exam (Part 2) Fall 2016

13. [5 pts ea] Convert each C function definition into ARM assembly language.

```
int32_t f9(int32_t x, int32_t a, int32_t b)
   {
   if (x < a) return a ;
   if (x > b) return b ;
   return x ;
   }
```

Do NOT use an IT block

```
f9:   CMP   R0,R1
      BGE   L1
      MOV   R0,R1
      BX    LR
L1:   CMP   R0,R2
      BLE   L2
      MOV   R0,R2
L2:   BX    LR
```

```
int32_t f10(uint32_t score)
   {
   if (score >= 60 && score <= 100)
      return 1 ;
   else
      return 0 ;
   }
```

Do NOT use an IT block

```
f10:  CMP   R0,60
      BLO   Else
      CMP   R0,100
      BHI   Else
Then: MOV   R0,1
      BX    LR
Else: MOV   R0,0
      BX    LR
```

NAME: _____

SANTA CLARA UNIVERSITY
Department of Computer Engineering

COEN 020 Final Exam (Part 2) Fall 2016

14. [5 pts ea] Convert each C function definition into ARM assembly language.

```
int32_t f11(int64_t a64, int64_t b64)
   {
   return (a64 < b64) ? 1 : 0 ;
   }
```

Use an IT block

```
f11:    SUBS    R0,R0,R2
        SBC     R1,R1,R3
        ITE     LT
        MOVLT   R0,1
        MOVGE   R0,0
        BX      LR
```

```
int32_t f12(int32_t a32)
   {
   int32_t f13(void) ;

   return f13() + a32 ;
   }
```

```
f12:    PUSH    {R4,LR}
        MOV     R4,R0
        BL      f13
        ADD     R0,R0,R4
        POP     {R4,LR}
        BX      LR
```

```
int32_t f14(int32_t *p32, int32_k)
   {
   return *(p32 + k) ;
   }
```

```
f14:    LDR   R0,[R0,R1,LSL 2]
        BX    LR
```

NAME: _____

SANTA CLARA UNIVERSITY
Department of Computer Engineering

COEN 020 Final Exam (Part 2) Fall 2016

15. [5 Pts] Write a function in ARM assembly language that calculates the single-length (32-bit) product of two 32-bit unsigned operands. Unlike regular multiplication, the function should return the maximum 32-bit unsigned value (0xFFFFFFFF) if the single-length product overflows. The function prototype should be:

```
uint32_t Product(uint32_t a, uint32_t b) ;

Product:      UMULL   R0,R1,R0,R1
              CBZ     R1,NoOverflow
              LDR     R0,=0xFFFFFFFF
NoOverflow:   BX      LR
```

16. [5 pts] Write a C inline function that returns the minimum of two integers. Use an extended asm statement inside the inline function so that the compiler is allowed to choose all of the registers. (Suggestion: Use an IT block for simplicity.)

```
static inline int32_t Minimum(int32_t a, int32_t b)
     {
     int32_t min ;

     asm(
          "CMP    %[a],%[b]      \n\t"
          "ITE    LT             \n\t"
          "MOVLT  %[min],%[a]    \n\t"
          "MOVGE  %[min],%[b]        "
          : [min] "=r" (min)
          : [a]   "r"  (a),
            [b]   "r"  (b)
          : "cc"
     ) ;

     return min ;
     }
```