Xavier Kuehn

1. Translate the following C statements into assembly language equivalents using instructions.

```
// assume x, y, z are in R0, R1, R2 respectively

z = (x < y) ? 6 : x;
     CMP  R0,R1
     BGE  L1
     MOV  R2,6
L1:  MOV  R2,R0

x = 0;
for(y = 1; y < 1000; y *= 2) x += y;
     MOV  R0,0
     MOV  R1,1
     MOV  R2,1000
L1:  CMP  R1,R2
     BGE  L2
     ADD  R0,R0,R1
     LSL  R1,R1,1
L2:  <loop done, something here>

if (x > 10) if (x < 20) y = 1; else z = 0;
     CMP  R0,10
     BLE  ELSE
     CMP  R0,20
     BGE  ELSE
     MOV  R1,1
ELSE:MOV  R2,0
```

2. Translate the following C statements into assembly language equivalents using IT blocks.

```
// assume a and b are in R0 and R1 respectively

uint16_t a, b;
if (a > 0 && a < 100) b = b / 2;
// use DeMorgan's > (a <= 0 || a >= 100)
```

```
     CMP       R0,0
     BLS       L1
     LSR       R1,R1,1
L1:  CMP       R0,100
     BHS       ELSE
     LSR       R1,R1,1
ELSE: <something>
```

```
uint16_t a, b;
if (a > 100 || a < 50) a += b;
// don't need DeMorgan's, again use two IT blocks
     CMP       R0,100
     BLS       L1
     ADD       R0,R0,R1
L1:  CMP       R0,50
     BHS       ELSE
     ADD       R0,R0,R1
ELSE: <something>
```

3. Use an IT block to convert the following C statements into assembly language.

```
// assume a, b, c are in R1.R0, R3.R2, R5.R4 respectively
```

```
if (a > b) c = b + 2;
     CMP       R1,R3
     ITE       LE
     BLE       ELSE
     CMPGT     R0,R2
     BLE       ELSE
     ADDGT     R4,R3,2
ELSE: <something>
```

```
if (a == b) c = 0; else c = a - b;
     CMP       R1,R3
     ITE       NE
     BNE       ELSE
     CMPEQ     R0,R2
     BEQ       L1
     BNE       ELSE
L1:  MOV       R4,0
     MOV       R5,0
```

```
ELSE:SUB       R4,R1,R3
```

4. Translate the following C statements into assembly language equivalents.

```
// assume ch, x, y, and z are in R0,R1,R2, and R3 respectively

if (x < y && y < z) z = 6; else z = x;
     CMP       R1,R2
     BGE       ELSE
     CMP       R2,R3
     BLE       ELSE
     MOV       R3,6
ELSE:LDR       R3,R1

if (-10 < x && x > 10) goto L1;
     CMP       -10,R1
     BGE       L1
     CMP       R1,10
     BGE       L1
     <something>
L1:  <something>

if (x < 10 || x > 20) y = 0; else y = 1;
     CMP       R1,10
     BGE       L1
     MOV       R2,0
L1:  CMP       R1,20
     BLE       ELSE
     MOV       R2,0
ELSE:MOV       R2,1

if ('a' <= ch && ch <= 'z') ch = ch - 'a' + 'A'
(a=97,A=65,z=122)
     CMP       97,R0
     BGT       L1
     SUB       R4,R4,97
     ADD       R0,R4,65
L1:  CMP       R0,122
     BGT       L2
     SUB       R4,R4,97
     ADD       R0,R4,65
```

```
L2: <something>

x = y / 5;
SDIV R0,R1,5

// assume u32 and s32 are in R0 and R1
if (u32 > 10) s32 -= 1; else s32 += 1;
     CMP  R0,10
     BLS  ELSE
     SUB  R1,R1,1
ELSE:ADD  R1,R1,1

if (-10 < s32 && s32 < 10) s32 = 0;
if (-10 >= s32 || s32 >= 10) goto else;
     CMP  -10,R1
     BGE  L1
     MOV  R1,0
L1:  CMP  R1,10
     BGE  ELSE
     MOV  R1,0
ELSE:<something>

// assume u32, min, max are in R0,R1,R2 respectively
if (u32 < min || u32 > max) u32 = 0;
     CMP  R0,R1
     BHS  L1
     MOV  R0,0
L1:  CMP  R0,R2
     BLS  L2
     MOV  R0,0
L2:  <something>
```