SHEET 2

Q1. What are the differences between the following three instructions?

LDRB R0, [R1] LDRH R0, [R1] LDR R0, [R1]

Q2. Translate the below C code for counting the number of occurrence of ones in R0 into ARM assembly code assuming that the initial value of r0=0x00AA, using the registers indicated by the variable names.

```
r3 = 1;
r1 = 0;
while (r3 != 0) {
    if ((r0 & r3) != 0) {
        r1 = r1 + 1;
    }
    r3 = r3 + r3;
}
```

Q3. Explain what an ARM processor accomplishes in terms of accessing and changing its registers when it executes a BEQ instruction.

Q4. Translate the below C code into ARM assembly code, using the registers indicated by the variable names. The C code presumes that r0 holds the address of the first entry of an array of integer values, and r1 indicates how many elements the array holds; the code removes all adjacent duplicates from the array.

```
r3 = 1;
for (r2 = 1; r2 < r1; r2++) {
    if (r0[r2] != r0[r2 - 1]) {
        r0[r3] = r0[r2];
        r3 += 1;
    }
}
r1 = r3;
```

Q5. Translate the below C fragment into an equivalent ARM assembly language program, using registers corresponding to the variable names. Assume r0 and r1 hold signed values.

```
r2 = 0;
while (r1 != 0) {
    if ((r1 & 1) != 0) {
        r2 += r0;
    }
    r0 <<= 1;
    r1 >>= 1;
}
while (1); // halting loop
```

Q6. For the below ARM assembly code, trace the values that will be placed into the registers R4, R5, and R6. By tracing, you are expected to write the values of the mentioned registers after the execution of each instruction.

```
        MOV R4, #7
        MOV R5, #4
        MOV R6, #4
again   MOV R7, R4
        ADD R4, R5, R4
        MOV R5, R7
        SUBS R6, R6, #1
        BNE again
```