#HW, cec32x_HW_30_HW6_arithmetic_instructions, 140 points total.

HW 6. Arithmetic Instructions

Prob 6-1. (60 points total, 10 points each) We have learned in the class that using the barrel shifter, we can implement multiplication/division of a number for some special cases with addition, subtraction, and reversed subtraction instructions. Write the assembly code to perform the following calculations using this approach assuming **signed** integers `A` and `B` are saved in `r0` and `r1`, respectively.

- `A = 16 * B`
- `A = 17 * B`
- `A = 15 * B`
- `A = B / 16`
- `A = 17 * B / 16`
- `A = 15 * B / 16`

Prob 6-2. (80 points total, 10 points each) Consider the following mixed C and assembly programming problem with code snippets given below:

- The code snippets in the C file:

```c
// Functions defined in a .s file which have do be declared before being used
extern void example12p1(uint32_t A0, uint32_t A1, uint32_t A2, uint32_t *pR);

// You can run this code side by side with LO13_2asm to compare the results.
// To see the results in the memory, use the unsigned hexadecimal format.
// For the best view of the results, change the width of the memory window
// so that memory address of its second line starts from ox2000 0020.

uint32_t A[8] = {0x01010111, 0x10010010, 5, 4, 5, 6, 7, 8};
uint32_t Ra[8]; // Results from assembly program
uint32_t Rc[8]; // Results from C program

int main(void){
    printf("Address of Ra: %p\n", Ra);
    printf("Address of Rc: %p\n", Rc);

    example12p1(A[0], A[1], A[2], Ra);

    while(1);
}
```

- Code snippets in the s file:

```
    EXPORT example12p1

    ALIGN
```

```
; r0 = A0, r1 = A1, r2 = A2, r3 = &Ra[0] = Ra
example12p1  PROC
        PUSH {r4, r5, lr}
        AND  r4, r0, r1                 ; Ra[0] = A[0] & A[1]
        STR  r4, [r3], #4
        ORR  r4, r0, r1                 ; Ra[1] = A[0] | A[1]
        STR  r4, [r3], #4
        ORN  r4, r0, r1                 ; Ra[2] = A[0] | ~A[1]
        STR  r4, [r3], #4
        EOR  r4, r0, r1                 ; Ra[3] = A[0] ^ A[1]
        STR  r4, [r3], #4
        BIC  r4, r0, r2                 ; Ra[4] = A[0] & ~A[2]
        STR  r4, [r3], #4
        MVN  r5, r0                     ; Ra[5] = ~A[0]
        STR  r5, [r3], #4
        MOV  r4, r5                     ; Ra[6] = Ra[5]
        BFC  r4, #2, #4                 ; Ra[6] &= ~(15U << 2)
        STR  r4, [r3], #4
        MOV  r4, r5                     ; Ra[7] = Ra[5]
        BFC  r4, #0, #4                 ; Ra[7] &= ~(15U)
        AND  r5, r2, #15                ; r5 = r2 & 15U
        ORR  r4, r5                     ; r4 = r4 | r5
        STR  r4, [r3], #4
        POP  {r4, r5, pc}
        ENDP

        END                             ; End of the entire file
```

Determine the values of `Ra[0]` to `Ra[7]` in hexadecimal.