CMPT 433: Embedded Systems

Bitwise Operations Worksheet (Answers)

Given the following predefined values where LEDs are active high and buttons are active low:

```
VALUE

LED0_BIT
LED1_BIT
LED2_BIT
LED_MASK = (1 << LED0_BIT) | (1 << LED3_BIT) | (1 << LED2_BIT)

BTN0_BIT
BTN1_BIT
BTN_MASK = (1 << BTN0_BIT) | (1 << BTN1_BIT)

SPD_BIT_BEGIN
SPD_MASK
```

Complete the following calculations.

```
_Bool isLed0On = (VALUE & (1 << LED0_BIT)) != 0;

_Bool isAnyLEDOn = ((VALUE & LED_MASK) != 0);

_Bool areAllLEDsOn = (VALUE & LED_MASK) == LED_MASK;

// If (VALUE & BTN_MASK) == BTN_MASK,
// then because buttons are active low, no buttons are pressed.
_Bool isAnyButtonPressed = (VALUE & BTN_MASK) != BTN_MASK;

_Bool areAllButtonsPressed = (VALUE & BTN_MASK) == 0;
```

```
void turnOnLed0() {
        VALUE |= (1 << LED0_BIT);
}

void turnOnAllLeds() {
        VALUE |= LED_MASK;
}

void turnOffLed() {
        // ~(1 << LED_BIT) sets the LED0 bit to 0, and all
        // other bits to 1.
        // ANDing it changes the LED0 bit to 0, and does not
        // change other bits.
        VALUE &= ~(1 << LED_BIT);
}

void turnOffLeds1And2() {
        VALUE &= ~(1 << LED1_BIT | 1 << LED2_BIT);
}

void turnOffAllLeds() {
        VALUE &= ~LED_MASK;
}

void turnOffAllLedsExcept2() {
        // Remove the LED2 bit from the inverted LED mask
        VALUE &= (~LED_MASK | (1 << LED2_BIT));
}

void toggleLed0() {
        VALUE ^= (1 << LED0_BIT);
}

void toggleAllLeds() {
        VALUE ^= LED_MASK;
}

// Assume ints are in the correct format and do not need
// to be converted to/from binary.
int getSpeed() {
        // Mask the correct bits
        // Bitshift the value so that the speed is at the least
        // significant (rightmost) bit
        return (VALUE & SPD_MASK) >> SPD_BIT;
}

int setSpeed(int speed) {
        // Shift speed to the correct location
        // Remove excess bits outside the speed bits
        int newSpeedBits = (speed << SPD_BIT) & SPD_MASK;
        // Create the value with cleared speed bits
        // Add the new speed bit
        VALUE = (VALUE & ~SPD_MASK) | newSpeedBits;
}
```