CPE 325: Embedded Systems Laboratory
Laboratory Assignment #5

Assignment [100 pts]

1. Write an assembly program that evaluates the equation $Y = mX + C$ for a given integer array, *X*. Assume that *m* and *C* are integer constants. This program is responsible for initializing the inputs – array *X*, *m*, *C*, calling the two subroutines – SW_linear, HW_linear, and allocating the space to store the results calculated by the subroutines. This program is also responsible for passing the required input parameters, array *X*, *m*, *C*, and address of the memory location reserved to store the result to the subroutines using the program stack.
2. SW_linear uses Shift-and-Add multiplication algorithm and HW_linear uses Hardware Multiplier to multiply *m* and *X* while evaluating the equation.
3. Determine the number of clock cycles required to evaluate the equation by each subroutine. How many elements can be evaluated using SW_linear and HW_linear in a sec? Explain which one is more efficient and why?
4. **Bonus (Each technique 8 pts – Total 16pts)** Initialize two 16-bit signed integer arrays, array1 and array2, containing 8 elements each. Create subroutines to calculate the dot product of the two arrays using software and hardware. To calculate dot product using software, use Shift-and-add multiplication algorithm from part 1 and for the hardware use hardware multiplier with accumulation. The dot product from two subroutines should be stored in the memory. The parameters (address of the arrays, length of the array, and address to store the final result) should be passed to the subroutines using either registers or the program stack.

Topics for Theory

1. Subroutines
2. Passing parameters

Deliverables

1. Lab report with screenshots of final outputs
2. Source files (.asm files)

Notes:

1. Do not hard code numbers or arrays in the sub routines. You have to pass the required parameters by using the specified method.
2. Do not allocate space to store the result in subroutines. It should be done using the **.data** section and passed to the subroutines the way described above.
3. Try different inputs before you conclude which method is efficient. In your explanation, include the inputs as well.
4. Assume that any of the results does not exceed 16-bits.