Microcontrolled based systems

Homework

Brodt Daniil

HF14R8

“I, Brodt Daniil, declare that this is my and only my own solution” - 2021.12.08

Task description:

Counting the number of "0" bits in a 128 bit pattern being in the internal memory.

Input: Start address of the pattern (pointer)

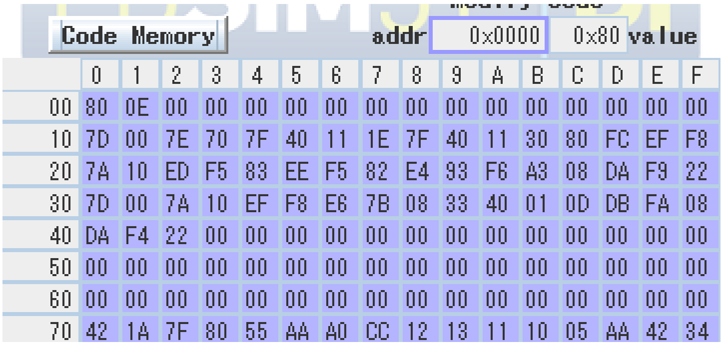
Output: The number of "0" bits in 1 register

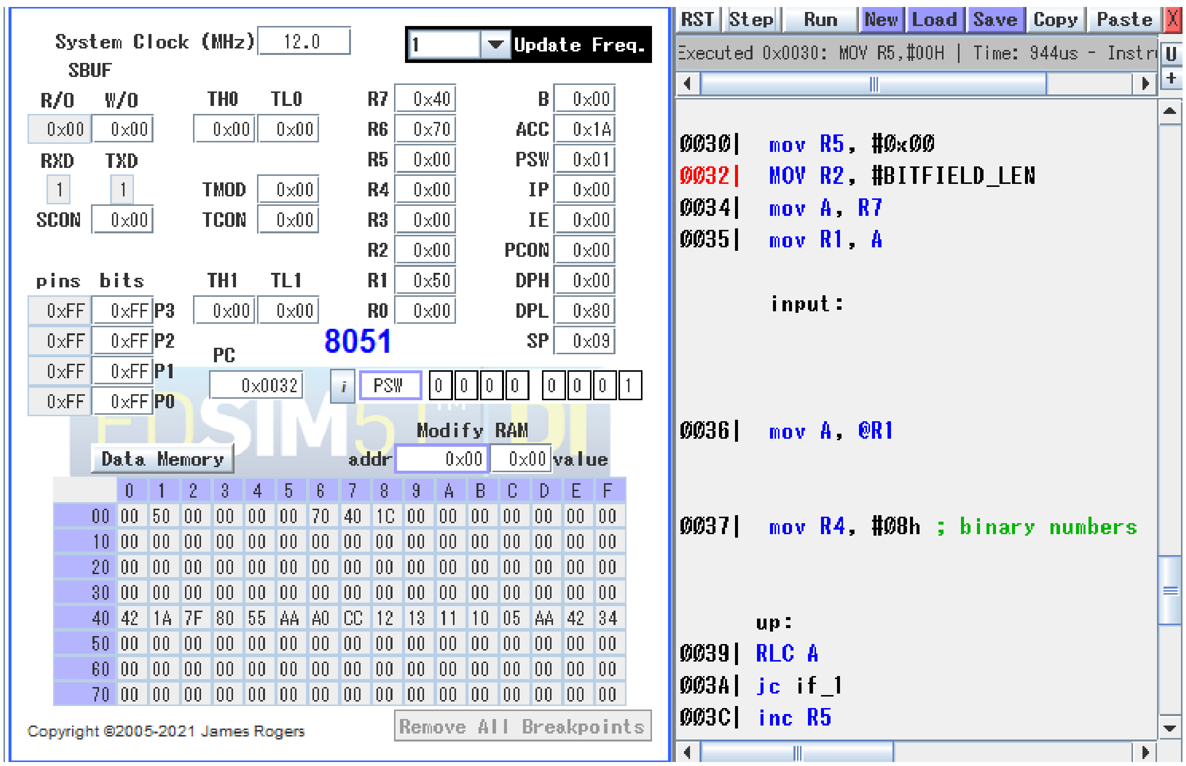
First we need to calculate the number of zeroes in given data manually: 0x42, 0x1A, 0x7F, 0x80, 0x55, 0xAA, 0xA0, 0xCC, 0x12, 0x13, 0x11, 0x10, 0x05, 0xAA, 0x42, 0x34. There are 82 zeroes in these 16 hexadecimal numbers, while the number we want to compare in assembly will be 52H since 82 DEC will be 52 in HEX.

Implemented algorithm: we need two subroutines to solve the given task

CODE2IRAM: requires 3 inputs and 0 outputs, and utilizes few registers, accumulator and DPTR

1) Move base address in IRAM to ACC, and then move ACC to R0 since we are going to use it for indirect addressing
2) Move the BITFIELD_LEN to R2 to use it for loop condition check (16 cells)
3) Move R5 and R6 to DPH and DPL accordingly since we are going to use DPTR
4) Then in the LOOP1 we move the HEX numbers from code memory to internal memory. The loop proceeds as such : clear ACC (to avoid jumping extra cell because of the stored address from the previous time), move data in DPTR to ACC, move ACC to base address in IRAM via @R0. Then we increment the value of R0 to move to the next HEX number and DJNZ to check R2 value to know whether we are done with all the HEX numbers or not



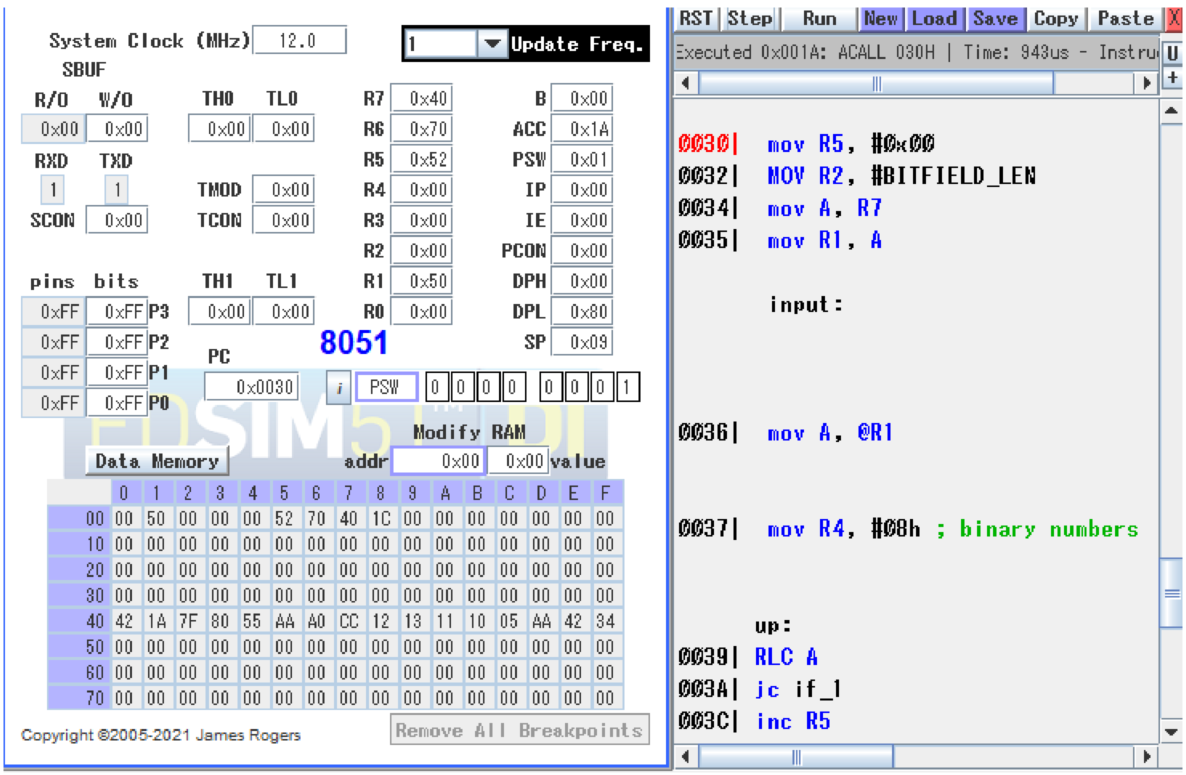


We can see from above that it perfectly transferred all the data from code memory to internal memory

COUNT_0: 1 input and 1 output, and utilized 4 registers and 1 acc

1) We reset the value of R5 to 0, same procedure for R2 as in the previous subroutine. Then we move the base address of IRAM to A and then from A to R1 to use for indirect addressing.

2) Enter the LOOP2 and we move the value from R1 to A to work with a pointer in the loop. Since we work on bytes, we also need to check the loop on the value 8 in R3 for DJNZ
3) Then we enter LOOP3 that analyzes each bit with a help of rotation and carry (RLC) and increments our output R5 in case of 0. After that it decrements R3 and jumps back
4) After the LOOP3 we increment R0 to point to the next cell and loop back to LOOP2 to process other HEX numbers.



At the end of the counting we get 52 in HEX which means it counted 82 zeroes in all given HEX numbers which proves that the algorithm works. Then it starts counting again in the infinite loop.

Flow-chart with arrows that are in the opposite direction