마이크로프로세서

CCode.h, CCode.cpp에서는 binary code인 file.bin에서 16자리의 숫자를 받는다.

CRegister.h, CRegister.cpp에서는 16bit의 레지스터를 구현한다.

CMemory.h, CMemory.cpp에서는 256개의 공간을 가지는 Memory를 구현한다.

CDecode.h, CDecode.cpp에서는 16자리의 binary code를 해독한다.

CExecute.h, CExecute.cpp에서는 Decoding한 binary code의 op code에 있는 instruction을 실행시킨다. calculating instruction들은 register에 존재해야 하며 MOV 계통의 경우 Memory->Register, Register->Memory, Register에 직접 쓰기 등을 수행 할 수 있다.

Makefile에서 몇 행까지의 binary code를 받도록 할 것인지 설정하여 tpu실행 파일을 컴파일/생성한다.

register 개수가 16개로 한정되어있어 a행렬과 b행렬 모두 입력할 수 없었기 때문에 14~16레지스터를 이용하여 2번째 행렬을 3개씩 메모리에 저장 및 연산까지 진행하였다. 아래는 file.bin의 소스 + 주석이다.

0011000000000001 //R0 #1

0011000100000010 //R1 #2

0011001000000011 //R2 #3

0011001100000100 //R3 #4

0011010000000101 //R4 #5

0011010100000110 //R5 #6

0011011000000111 //R6 #7

0011011100001000 //R7 #8

0011100000001001 //R8 #9

0001000000000000 //M[0] <- R[0]

0001000100000001 //M[1] <- R[1]

0001001000000010 //M[2] R[2]

0001001100000011 //M[3] R[3]

0001010000000100 //M[4] R[4]

0001010100000101 //M[5] R[5]

0001011000000110 //M[6] R[6]

0001011100000111 //M[7] R[7]

0001100000001000 //M[8] R[8]

0011111100001010 //R15 #10

0011111000001101 //R14 #13

0011110100010000 //R13 #16

0001111100001001 //M[9] R15

0001001100001100 //M[12] R14

0001110100001111 //M[15] R13

0110111100000000 //MUL R15 R0

0110111000010000 //MUL R14 R1

0110110100100000 //MUL R13 R2

0100111111100000 //ADD R15 R14

0100111111010000 //ADD R15 R13

0001111100010010 //M[18] R15

0011111100001011 //R[15] #11

0011111000001110 //R[14] #14

0011110100010001 //R[13] #17

0001111100001010 //M[10] R15

0001111000001101 //M[13] R14

0001110100010000 //M[16] R[13]

0110111100000000 //MUL R15 R0

0110111000010000 //MUL R14 R1

0110110100100000 //MUL R13 R2

0100111111100000 // ADD R15 R14

0100111111010000 //ADD R15 R13

0001111100010011 //M[19] R15

0011111100001100 //R15 #12

0011111000001111 //R[14] #15

0011110100010010 //R[13] #18

0001111100001011 //M[11] R15

0001111000001110 //M[14] R14

0001110100010001 // M[17] R[13]

0110111100000000 //MUL R15 R0

0110111000010000 //MUL R14 R1

0110110100100000 //MUL R13 R2

0100111111100000 //ADD R15 R14

0100111111010000 //ADD R15 R13

0001111100010100 //M[20] R15

1째줄 완료

0011111100001010 //R15 #10

0011111000001101 //R14 #13

0011110100010000 //R13 #16

0110111100110000 //MUL R15 R3

0110111001000000 //MUL R14 R4

0110110101010000 //MUL R13 R5

0100111111100000 //ADD R15 R14

0100111111010000 //ADD R15 R13

0001111100010101 //M[21] R15

0011111100001011 //R[15] #11

0011111000001110 //R[14] #14

0011110100010001 //R[13] #17

0110111100110000 //MUL R15 R3

0110111001000000 //MUL R14 R4

0110110101010000 //MUL R13 R5

0100111111100000 //ADD R15 R14

0100111111000000 //ADD R15 R13

0001111100010110 //M[22] R15

0011111100001100 //R15 #12

0011111000001111 //R[14] #15

0011110100010010 //R[13] #18

0110111100110000 //MUL R15 R3

0110111001000000 //MUL R14 R4

0110110101010000 //MUL R13 R5

0100111111100000 //ADD R15 R14

0100111111010000 //ADD R15 R13

0001111100010111 //M[23] R15

2째줄 완료

0011111100001010 //R15 #10

0011111000001101 //R14 #13

0011110100010000 //R13 #16

0110111101100000 //MUL R15 R6

0110111001110000 //MUL R14 R7

0110110110000000 //MUL R13 R8

0100111111100000 //ADD R15 R14

0100111111010000 //ADD R15 R13

0001111100011000 //M[24] R15

0011111100001011 //R[15] #11

0011111000001110 //R[14] #14

0011110100010001 //R[13] #17

0110111101100000 //MUL R15 R6

0110111001110000 //MUL R14 R7

0110110110000000 //MUL R13 R8

0100111111100000 //ADD R15 R14

0100111111010000 //ADD R15 R13

0001111100011001 //M[25] R15

0011111100001100 //R[15] #12

0011111000001111 //R[14] #15

0011110100010010 //R[13] #18

0110111101100000 //MUL R15 R6

0110111001110000 //MUL R14 R7

0110110110000000 //MUL R13 R8

0100111111100000 //ADD R15 R14

0100111111010000 //ADD R15 R13

0001111100011010 //M[26] R15

3째줄 완료 끝.