CS 366: Programming Assignment 5

Division

Due: 30 Nov 2005, 11:59 AM

Division using a table As part of this assignment you are expected to write micro-code for division instruction that uses a table in the background to perform the operation.

Instruction	Meaning
DIV ri, rj, rk	ri ← rj / rk

The division is done using table lookup. The table to be used for this assignment has 64 rows, which means the number range of numbers possible for rj are 0 through 63 (64 in all). The table has 2 columns for the divisors — one for 2 and the other for 4. This implies that the possible values for rk are 2 and 4.

Number	2	4
000000	000000	000000
000001	000000	000000
000010	000001	000000
000011	000001	000000
000100	000010	000001
000101	000010	000001
000110	000011	000001
000111	000011	000001
001000	000100	000010
...	...	...
111111	011111	001111

The table has to be stored in memory and its starting address stored in one of the MythSim registers. The division operation uses this starting address and the number rj to get to the row and the value for rk (the column) to get the quotient.

Feel free to write any other basic instruction (like branch, add etc) that would be necessary to complete the multiplier program described below.

Parity Generator As part of this assignment also write a program that generates odd-parity bits for a set of numbers using the following algorithm:

X := starting address **for** data;
C := 0;
N := 01000000; /* *binary* */

```
do {
  M := read 8−bit integer from X;
  for (i=0; i < 7; i++) {
    if (M & N) C++;
    N := N / 2;
  }
  M := M + M; /* M := M * 2 */
  if !(C & 1)
    M := M + 1;
  write M back into location X;
  X := X + 1;
} while (M != 1);
```

The memory organization for this assignment might look as shown below:

Sample code (contents of .mem file)	
0:	instruction
1:	...
	...
X-2:	HALT
X-1:	HALT
X :	00101010
X+1 :	00010010
X+2 :	...
X+N :	00000000

The starting address for the data is X. The input numbers are 7-bit. The leading bit is a zero. The output being written back into the same location as the input will be 8-bit with the parity occupying bit-0 and the number in the first 7-bits. (This bit shift is done in the step M := M + M, which is equivalent to multiplying by 2). The loop halts on input 0.

Turnin The command for the electronic turnin is:

`turnin -c cs366 -p myth_div` ⟨*files*⟩

The hard-copy should contain

- a listing of your .ucode and .mem file (with some comments)
- a table with all the opcodes (you wrote) in the .ucode file and having a description for each opcode similar in style to the table given above
- the status of your implementation (what works and what doesn't)

This is due in class.