Project Assignment # 1

Solution

1. (a) Amdahl's Law, f = 20%, s = 2

   $$\text{speedup} = \frac{1}{1 - f + \frac{f}{s}} = \frac{1}{1 - .2 + \frac{.2}{2}} \approx 1.11$$

   (b) Amdahl's Law, $f_1 = 20\%, s_1 = 2, f_2 = 10\%, s_2 = .8$

   $$\text{speedup} = \frac{1}{1 - f_1 - f_2 + \frac{f_1}{s_1} + \frac{f_2}{s_2}} = \frac{1}{1 - .2 - .1 + \frac{.2}{2} + \frac{.1}{.8}} \approx 1.08$$

   (c) $t_{fp} = \frac{.2}{2} = .1$
   $t_{dc} = \frac{.1}{.8} = .125$
   $t_{total} = 1 - .2 - .1 + .1 + .125 = .925$
   % time in floating point $= \frac{t_{fp}}{t_{total}} = \frac{.1}{.925} = 10.8\%$
   % time in data cache $= \frac{t_{dc}}{t_{total}} = \frac{.125}{.925} = 13.5\%$

2. (a) $CPI_{M1} = 1 \times 0.55 + 2 \times 0.25 + 4 \times 0.20 = 1.85$
   $CPI_{M2} = 2 \times 0.55 + 3 \times 0.25 + 4 \times 0.20 = 2.65$

   (b) $MIPS_{M1} = \frac{CLK_{M1}}{CPI_{M1}} \times 10^{-6} = \frac{100M}{1.85} \times 10^{-6} = 54.05$
   $MIPS_{M2} = \frac{CLK_{M2}}{CPI_{M2}} \times 10^{-6} = \frac{150M}{2.65} \times 10^{-6} = 56.60$

   (c) M1 has a smaller MIPS rating. To reduce the average number of cycles of instruction B or C on Machine M1:

   i. B in 1 cycle: $MIPS_{M1}{}' = \frac{100M}{1\times0.55+1\times0.25+4\times0.20} \times 10^{-6} = 62.50$
   ii. C in 3 cycles: $MIPS_{M1}{}' = \frac{100M}{1\times0.55+2\times0.25+3\times0.20} \times 10^{-6} = 60.60$
   iii. C in 2 cycles: $MIPS_{M1}{}' = \frac{100M}{1\times0.55+2\times0.25+2\times0.20} \times 10^{-6} = 68.97$
   iv. C in 1 cycle: $MIPS_{M1}{}' = \frac{100M}{1\times0.55+2\times0.25+1\times0.20} \times 10^{-6} = 80.00$

   Changing the CPI of B from 2 to 1, or the CPI of C from 4 to 3, 2, or 1 on M1 can achieve a higher MIPS rating for M1 than M2.

3. Loads-stores: $\frac{26.5\%+20.1\%+10.3\%+5.1\%}{2} = 31\%$
   Conditional Branches: $\frac{9.3\%+11.0\%+.4\%+1.1\%}{2} = 10.9\%$
   Jumps, calls, returns: $\frac{0.8\%+0.8\%+1.6\%+0.4\%+1.6\%+0.4\%}{2} = 2.8\%$
   ALU instructions: $1 - 31\% - 10.9\% - 2.8\% = 55.3\%$
   CPI: $1.0\times55.3\%+1.6\times31\%+(2.3\times60\%+1.6\times40\%)\times10.9\%+1.1\times2.8\% \approx 1.30$

4. The length of the instructions are 12 bits. There are 32 registers so the length of the register field will be 5 bits for each register operand. We use addr[11] to addr[0] to represent the 12 bits of the address as shown in the tables below.

   (a) In the first case, we need to support 3 two-address instructions. These can be encoded as follows:

	addr[11:10]	**addr[9:5]**	**addr[4:0]**
3 two-addr inst.	'00' - '10'	'00000' - '11111'	'00000' - '11111'
other inst.	'11'	'00000' - '11111'	'00000' - '11111'

Hence, for the one-address and two-address instructions must be mapped to the remaining 10 bits with the upper two bits encoded as '11'. The one-address instructions are then encoded with the addr[9:5] field using '00000' - '11101' for the 30 instruction types, leaving the addr[4:0] field for the register operand. This leaves the patterns with '11' followed by '11110' in the upper seven bits and '00000' - '11111' in the lower five bits to encode 32 of the zero-address instructions. The remaining zero-address instructions can be encoded using '11' followed by '11111' in the upper seven bits and '00000' to '10100' in the lower five bits to encode the other 20 zero-address instructions.

	addr[11:10]	**addr[9:5]**	**addr[4:0]**
3 two-addr inst.	'00' - '10'	'00000' - '11111'	'00000' - '11111'
30 one-addr inst.	'11'	'00000' - '11101'	'00000' - '11111'
52 zero-addr inst.	'11'	'11110'	'00000' - '11111'
	'11'	'11111'	'00000' - '10100'

Hence, it is possible to have the above instruction encodings.

(b)

	addr[11:10]	**addr[9:5]**	**addr[4:0]**
2 two-addr inst.	'00' - '01'	'00000' - '11111'	'00000' - '11111'
32 one-addr inst.	'10'	'00000' - '11111'	'00000' - '11111'
29 one-addr inst.	'11'	'00000' - '11111'	'00000' - '11100'
x zero-addr inst.	'11'	'00000' - '11111'	'11101' - '11111'

Because x includes 96 patterns (there are 32 possible values for addr[9:5], and 3 non-overlapping values in addr[4:0]), we can support up to 96 zero-address instructions.

5. Assuming there are s million instructions in a program:

   (a) T: $\frac{\alpha \cdot s}{n \cdot x} + \frac{(1-\alpha) \cdot s}{x}$
   MIPS: $\frac{s}{T} = \frac{nx}{\alpha + (1-\alpha) \cdot n}$

   (b) T_{new}: $\frac{\alpha \cdot s + \beta \cdot s}{n \cdot x} + \frac{(1-\alpha) \cdot s}{x}$
   Speedup: $\frac{T}{T_{new}} = \frac{\frac{\alpha \cdot s}{n \cdot x} + \frac{(1-\alpha) \cdot s}{x}}{\frac{\alpha \cdot s + \beta \cdot s}{n \cdot x} + \frac{(1-\alpha) \cdot s}{x}} = \frac{\alpha + (1-\alpha) \cdot n}{\alpha + (1-\alpha) \cdot n + \beta}$

   Note: changing β to $(\beta + 1)^{n-1}$ or $(n-1) \cdot \beta$ is also okay.